WebSphere® eXtreme Scale Versão 7.1
Guia de Administração

Guia de Administração do WebSphere eXtreme Scale

Esta edição aplica-se à versão 7, release 1, do WebSphere eXtreme Scale e a todos os releases e modificações subsequentes até que seja indicado de outra forma em novas edições.

Índice

Figuras

Tabelas

Sobre o *Guia de Administração*

O conjunto da documentação do WebSphere eXtreme Scale inclui três volumes que fornecem as informações necessárias para utilizar, programar e administrar o produto WebSphere eXtreme Scale.

Biblioteca do WebSphere eXtreme Scale

A biblioteca do WebSphere eXtreme Scale contém os seguintes livros:

- O *Guia de Administração* contém as informações necessárias para os administradores de sistema, incluindo como planejar implementações do aplicativo, planejar capacidade, instalar e configurar o produto, iniciar e parar servidores, monitorar o ambiente e proteger o ambiente.
- O *Guia de Programação* contém informações para desenvolvedores de aplicativos sobre como desenvolver aplicativos para o WebSphere eXtreme Scale utilizando as informações da API incluídas.
- O *Visão Geral do Produto* contém um visualização de alto nível dos conceitos do WebSphere eXtreme Scale, incluindo cenários de caso de uso e tutoriais.

Para fazer download dos manuais, vá para a Página da Biblioteca do WebSphere eXtreme Scale.

Também é possível acessar as mesmas informações nesta biblioteca no centro de informações do WebSphere eXtreme Scale.

Quem Deve Utilizar este Manual

Este manual é destinado principalmente para administradores de sistema, administradores de segurança e operadores de sistema.

Como este Manual Está Estruturado

O manual contém informações sobre os seguintes tópicos principais:

- **Capítulo 1** inclui informações sobre introdução.
- **Capítulo 2** inclui informações sobre o planejamento da implementação do aplicativo.
- **Capítulo 3** inclui informações sobre o planejamento de capacidade.
- **Capítulo 4** inclui informações sobre a instalação do produto.
- **Capítulo 5** inclui informações sobre a customização da plataforma z/OS.
- **Capítulo 6** inclui informações sobre a configuração do produto.
- **Capítulo 7** inclui informações sobre a administração do ambiente.
- **Capítulo 8** inclui informações sobre o monitoramento do ambiente de implementação.
- **Capítulo 9** inclui informações sobre a segurança do ambiente de implementação.
- **Capítulo 10** inclui informações sobre a resolução de problemas do ambiente.
- **Capítulo 11** inclui informações sobre o glossário do produto.

Obtendo Atualizações para este Manual

É possível obter as atualizações para esse manual ao fazer download da versão mais recente da Página da Biblioteca do WebSphere eXtreme Scale.

Como Enviar Seus Comentários

Entre em contato com a equipe de documentação. Você localizou o que precisava? O conteúdo era exato e completo? Envie seus comentários sobre esta documentação por e-mail para wasdoc@us.ibm.com.

Capítulo 1. Introdução ao WebSphere eXtreme Scale

Depois de instalar o WebSphere eXtreme Scale em um ambiente independente, use as seguintes etapas como uma simples introdução para sua capacidade como uma grade de dados em memória.

A instalação independente do WebSphere eXtreme Scale inclui uma amostra que pode ser usada para verificar a sua instalação e ver como uma grade e um cliente simples do eXtreme Scale podem ser usados. A amostra de iniciação está localizada no diretório *installRoot*/ObjectGrid/gettingstarted.

A amostra de iniciação é fornecida para uma introdução rápida à operação básica e de funcionalidade do eXtreme Scale. A amostra consiste de scritps de shell e lote projetados para iniciar uma grade simples com muito pouca customização necessária. Além disso, um programa cliente, incluindo a fonte, é fornecido para executar funções simples de criação, leitura, atualização e exclusão (CRUD) nesta grade básica.

Scripts e suas Funções

Essa amostra fornece os seguintes scripts:

O script env.sh|bat é chamado pelos outros scripts para configurar as variáveis necessárias do ambiente. Normalmente não é necessário alterar esse script.

- UNIX Linux `./env.sh`
- Windows `env.bat`

O script runcat.sh|bat inicia o processo de serviço de catálogo do eXtreme Scale no sistema local.

- UNIX Linux `./runcat.sh`
- Windows `runcat.bat`

O script runcontainer.sh|bat inicia um processo de servidor de contêiner. É possível executar esse script várias vezes com nomes do servidor exclusivos especificados para iniciar qualquer número de contêineres. Essas instâncias podem funcionar juntas para hospedar informações particionadas e redundantes na grade.

- UNIX Linux `./runcontainer.sh unique_server_name`
- Windows `runcontainer.bat unique_server_name`

O script runclient.sh|bat executa o cliente CRUD simples e inicia a operação especificada.

- UNIX Linux `./runclient.sh command value1 value2`
- Windows `runclient.sh command value1 value2`

Para *command*, use uma das seguintes opções:

- Especifique `i` para inserir *value2* na grade com a chave *value1*
- Especifique `u` para atualizar o objeto com chave pelo *value1* para o *value2*
- Especifique `d` para excluir o objeto com chave pelo *value1*

- Especifique `g` para recuperar e exibir o objeto com chave pelo *value1*

Nota: O arquivo *installRoot*`/ObjectGrid/ gettingstarted/src/Client.java` é o programa cliente que demonstra como se conectar a um servidor de catálogo, obter uma instância de ObjectGrid, e usar a API de ObjectMap.

Etapas Básicas

Use as seguintes etapas para iniciar sua primeira grade e executar um cliente para interagir com a grade.

1. Abra uma janela de sessão do terminal ou linha de comandos.
2. Utilize o seguinte comando para navegar para o diretório `gettingstarted`:

   `cd` *installRoot*`/ObjectGrid/gettingstarted`

   Substitua *installRoot* pelo caminho para o diretório-raiz da instalação do eXtreme Scale ou o caminho do arquivo-raiz do trial do eXtreme Scale extraído *installRoot*.
3. Execute o script a seguir para iniciar um processo de serviço de catálogo no host local:
   - UNIX Linux `./runcat.sh`
   - Windows `runcat.bat`

   O processo do serviço de catálogo executa na janela do terminal atual.
4. Abra outra janela de sessão de terminal ou de linha de comandos e execute o seguinte comando para iniciar uma instância do servidor de contêiner:
   - UNIX Linux `./runcontainer.sh server0`
   - Windows `runcontainer.bat server0`

   O servidor de contêiner será executado na janela do terminal atual. Repita as etapas 5 e 6 se desejar iniciar mais instâncias do servidor de contêiner para suportar a replicação.
5. Abra outra janela de sessão de terminal ou linha de comandos para executar comandos do cliente.
   - Incluir dados na grade:
     - UNIX Linux `./runclient.sh i key1 helloWorld`
     - Windows `runclient.bat i key1 helloWorld`
   - Procurar e exibir o valor:
     - UNIX Linux `./runclient.sh g key1`
     - Windows `runclient.bat g key1`
   - Atualizar o valor:
     - UNIX Linux `./runclient.sh u key1 goodbyeWorld`
     - Windows `runclient.bat u key1 goodbyeWorld`
   - Excluir o valor:
     - UNIX Linux `./runclient.sh d key1`
     - Windows `runclient.bat d key1`
6. Use <ctrl+c> para parar o processo de serviço de catálogo e os servidores de contêiner em suas respectivas janelas.

Definindo um ObjectGrid

A amostra usa os arquivos `objectgrid.xml` e `deployment.xml` que estão no diretório *installRoot*`/ObjectGrid/gettingstarted/xml` para iniciar um servidor de contêineres. O arquivo `objectgrid.xml` é o arquivo XML descritor do ObjectGrid e o arquivo `deployment.xml` é o arquivo XML descritor de política de implementação do ObjectGrid. Ambos os arquivos juntos definem uma topologia de ObjectGrid distribuída.

Arquivo XML descritor do ObjectGrid

Um arquivo XML descritor de ObjectGrid é usado para definir a estrutura do ObjectGrid que será usada pelo aplicativo. Ele inclui uma lista de configurações de BackingMap. Esses BackingMaps são o armazenamento de dados atual para dados em cache. O exemplo a seguir é um arquivo `objectgrid.xml` de amostra. As primeiras linhas do arquivo incluem o cabeçalho obrigatório de cada arquivo XML do ObjectGrid. Este arquivo de exemplo define o Grid ObjectGrid com BackingMaps Map1 e Map2.

```
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

    <objectGrids>
        <objectGrid name="Grid">
            <backingMap name="Map1" />
            <backingMap name="Map2" />
        </objectGrid>
    </objectGrids>

</objectGridConfig>
```

Arquivo Descritor XML de Política de Implementação

Um arquivo XML descritor de política de implementação é passado para um servidor de contêineres ObjectGrid durante a inicialização. Uma política de implementação deve ser usada com o arquivo XML de ObjectGrid e deve ser compatível com o XML de ObjectGrid que é usado com ele. Para cada elemento objectgridDeployment na política de implementação, você deve ter um elemento ObjectGrid correspondente no XML do seu ObjectGrid. Os elementos de backingMap que são definidos dentro do elemento objectgridDeployment devem ser consistentes com os backingMaps localizados no XML de ObjectGrid. Cada backingMap deve ser referido dentro de um e apenas um mapSet.

O arquivo XML descritor de política de implementação deve igualar-se com o arquivo XML ObjectGrid correspondente, o arquivo `objectgrid.xml`. No seguinte exemplo, as primeiras linhas do arquivo `deployment.xml` incluem o cabeçalho obrigatório de cada arquivo XML de política de implementação. O arquivo define o elemento objectgridDeployment para o ObjectGrid da Grade que está definido no arquivo `objectgrid.xml`. Ambos os BackingMaps, Map1 e Map2, que são definidos dentro do ObjectGrid da Grade estão incluídos no mapSet que tem os atributos numberOfPartitions, minSyncReplicas e maxSyncReplicas configurados.

```
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy
 ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

    <objectgridDeployment objectgridName="Grid">
        <mapSet name="mapSet" numberOfPartitions="13" minSyncReplicas="0"
maxSyncReplicas="1" >
```

```
            <map ref="Map1"/>
            <map ref="Map2"/>
        </mapSet>
    </objectgridDeployment>

</deploymentPolicy>
```

O atributo numberOfPartitions do elemento mapSet especifica o número de partições para o mapSet. Ele é um atributo opcional e o padrão é 1. O número deve ser apropriado para a capacidade antecipada da grade.

O atributo minSyncReplicas de mapSet especifica o número mínimo de réplicas síncronas para cada partição no mapSet. Ele é um atributo opcional e o padrão é 0. Primária e réplica não são colocadas até que o domínio possa suportar o número mínimo de réplicas síncronas. Para suporte do valor minSyncReplicas, é preciso de mais um contêiner do que o valor de minSyncReplicas. Se o número de réplicas síncronas ficar abaixo do valor de minSyncReplicas, grave transações que não são mais permitidas àquela partição.

O atributo maxSyncReplicas de mapSet especifica o número máximo de réplicas síncronas para cada partição no mapSet. Ele é um atributo opcional e o padrão é 0. Nenhuma outra réplica síncrona é posicionada para uma partição após um domínio alcançar este número de réplicas síncronas para esta partição específica. A inclusão de contêineres que podem suportar esse ObjectGrid pode resultar em um aumento no número de réplicas síncronas se seu valor de maxSyncReplicas ainda não tiver sido atingido. A amostra define o maxSyncReplicas para 1, que significa que o domínio posicionará, no máximo, uma réplica síncrona. Se você iniciar mais de uma instância do servidor de contêineres, haverá somente uma réplica síncrona posicionada em uma das instâncias do servidor de contêineres.

Usando o ObjectGrid

O arquivo `Client.java` no diretório *`installRoot`*`/ObjectGrid/gettingstarted/src/` é o programa cliente que demonstra como se conectar a um servidor de catálogos, obter uma instância de ObjectGrid, e usar a API de ObjectMap.

Da perspectiva de um aplicativo cliente, o uso do WebSphere eXtreme Scale pode ser dividido nas seguintes etapas.

1. Conexão com o serviço de catálogo por meio da obtenção de uma instância de ClientClusterContext.
2. Obtenção de uma instância do ObjectGrid do cliente.
3. Obtenção de uma instância da Sessão.
4. Obtenção de uma instância do ObjectMap.
5. Uso de métodos ObjectMap.

1. **Conectando ao serviço de catálogo por meio da obtenção de uma instância do ClientClusterContext**

   Para se conectar a um servidor de catálogos, use o método connect da API ObjectGridManager. O método connect usado requer apenas o terminal do servidor de catálogos no formato de `hostname:port`, como `localhost:2809`. Se a conexão com o servidor de catálogos for bem-sucedida, o método connect retornará uma instância do ClientClusterContext. A instância do ClientClusterContext é necessária para a obtenção do ObjectGrid a partir da

API do ObjectGridManager. O trecho de código a seguir demonstra como se conectar a um servidor de catálogos e obter uma instância do ClientClusterContext.

```
ClientClusterContext ccc = ObjectGridManagerFactory.getObjectGridManager().connect("localhost:2809", null, null);
```

2. **Obtendo uma Instância do ObjectGrid**

   Para obter uma instância do ObjectGrid, use o método getObjectGrid da API do ObjectGridManager. O método getObjectGrid necessita de ambos, a instância do ClientClusterContext e o nome da instância do ObjectGrid. A instância do ClientClusterContext é obtida durante a conexão com o servidor de catálogos. O nome de ObjectGrid é `Grid` que é especificado no arquivo `objectgrid.xml`. O fragmento de código a seguir demonstra como obter o ObjectGrid chamando o método getObjectGrid da API ObjectGridManager.

   ```
   ObjectGrid grid = ObjectGridManagerFactory.getObjectGridManager().getObjectGrid(ccc, "Grid");
   ```

3. **Obtendo uma Instância da Sessão**

   É possível obter uma Sessão da instância do ObjectGrid obtida. Uma instância da Sessão é necessária para obter a instância do ObjectMap e executar a demarcação da transação. O fragmento de código a seguir demonstra como obter uma instância da Sessão chamando o método getSession da API ObjectGrid.

   ```
   Session sess = grid.getSession();
   ```

4. **Obtendo uma Instância do ObjectMap**

   Após obter uma Sessão, é possível obter uma instância do ObjectMap a partir de uma instância da Sessão chamando o método getMap da API de Sessão. Você deve passar o nome do mapa como parâmetro para o método getMap para obter a instância do ObjectMap. O fragmento de código a seguir demonstra como obter ObjectMap chamando o método getMap da API de Sessão.

   ```
   ObjectMap map1 = sess.getMap("Map1");
   ```

5. **Usando os métodos ObjectMap**

   Após a obtenção de um ObjectMap, é possível usar a API ObjectMap. Lembre-se de que a interface ObjectMap é um mapa transacional e requer demarcação de transação usando os métodos begin e commit da API Session. Se não houver demarcação de transação explícita no aplicativo, as operações do ObjectMap serão executadas com transações de confirmação automática.

   O fragmento de código a seguir demonstra como usar a API ObjectMap com transação de confirmação automática.

   ```
   map1.insert(key1, value1);
   ```

   O fragmento de código a seguir demonstra como usar a API ObjectMap com demarcação de transação explícita.

   ```
   sess.begin();
   map1.insert(key1, value1);
   sess.commit();
   ```

Informações adicionais

Esta amostra demonstra como iniciar o servidor de catálogos e o servidor de contêiner e usar a API do ObjectMap em um ambiente independente. Também é possível usar a API do EntityManager.

Em um ambiente WebSphere Application Server com oWebSphere eXtreme Scale instalado ou ativado, o cenário mais comum é uma topologia conectada à rede. Em uma topologia conectada à rede, o servidor de catálogos é hospedado no processo do gerenciador de implementação do WebSphere Application Server e cada

instância doWebSphere Application Server hospeda um servidor eXtreme Scale automaticamente. Os aplicativos Java™ Platform, Enterprise Edition precisam somente incluir ambos, o arquivo XML descritor do ObjectGrid e o arquivo XML descritor da política de implementação do ObjectGrid, no diretório `META-INF` de qualquer módulo e o ObjectGrid se torna disponível automaticamente. Então, um aplicativo pode se conectar a um servidor de catálogos disponível localmente e obter uma instância do ObjectGrid para uso.

Convenções de Diretório

Este tópico descreve muitos exemplos e a sintaxe da linha de comandos que deve fazer referência a diretórios especiais como *wxs_install_root* e *wxs_home*. Esses diretórios são definidos da seguinte maneira.

wxs_install_root
O diretório *wxs_install_root* é o diretório-raiz no qual arquivos do produto WebSphere eXtreme Scale são instalados. Este pode ser o diretório no qual o arquivo zip de teste é extraído ou o diretório no qual o produto eXtreme Scale é instalado.

- Exemplo ao extrair o teste:

  ```
  /opt/IBM/WebSphere/eXtremeScale
  ```

- Exemplo quando o eXtreme Scale é instalado em um diretório independente:

  ```
  /opt/IBM/eXtremeScale
  ```

- Exemplo quando o eXtreme Scale é integrado ao WebSphere Application Server:

  ```
  /opt/IBM/WebSphere/AppServer
  ```

wxs_home
O diretório *wxs_home* é o diretório-raiz de bibliotecas, amostras e componentes do produto WebSphere eXtreme Scale. É o mesmo que o diretório *wxs_install_root* quando o teste é extraído. Para instalações independentes, esse é o subdiretório de ObjectGrid dentro de *wxs_install_root*. Para instalações integradas ao WebSphere Application Server, esse diretório está no diretório `optionalLibraries/ObjectGrid` dentro de *wxs_install_root*.

- Exemplo ao extrair o teste:

  ```
  /opt/IBM/WebSphere/eXtremeScale
  ```

- Exemplo quando o eXtreme Scale é instalado em um diretório independente:

  ```
  /opt/IBM/eXtremeScale/ObjectGrid
  ```

- Exemplo quando o eXtreme Scale é integrado ao WebSphere Application Server:

  ```
  /opt/IBM/WebSphere/AppServer/optionalLibraries/ObjectGrid
  ```

was_root
O diretório *was_root* é o diretório-raiz de uma instalação do WebSphere Application Server:

```
/opt/IBM/WebSphere/AppServer
```

restservice_home
O diretório *restservice_home* é o diretório no qual as bibliotecas e amostras do serviço de dados REST do eXtreme Scale estão localizadas. Este diretório é denominado *restservice* e é um subdiretório no diretório *wxs_home*.

- Exemplo para implementações independentes:

  ```
  /opt/IBM/WebSphere/eXtremeScale/ObjectGrid/restservice
  ```

- Exemplo para implementações integradas do WebSphere Application Server:

```
/opt/IBM/WebSphere/AppServer/optionalLibraries/ObjectGrid/restservice
```

tomcat_root
O *tomcat_root* é o diretório-raiz da instalação do Apache Tomcat.

```
/opt/tomcat5.5
```

wasce_root
O *wasce_root* é o diretório-raiz da instalação do WebSphere Application Server Community Edition.

```
/opt/IBM/WebSphere/AppServerCE
```

java_home
java_home é o diretório-raiz de uma instalação de Java Runtime Environment (JRE).

```
/opt/IBM/WebSphere/eXtremeScale/java
```

Capítulo 2. Planejamento de Capacidade

Se você tiver um tamanho de conjunto de dados inicial e um tamanho de conjunto de dados projetado, é possível planejar a capacidade necessária para executar o WebSphere eXtreme Scale. Embora esse planejamento ajude a implementar o eXtreme Scale eficientemente para futuras alterações, ele permite maximizar a elasticidade do eXtreme Scale o que não aconteceria em um cenário diferente, como um banco de dados de memória ou outro tipo de banco de dados.

Visão Geral de Conceitos de Escalabilidade

A escalabilidade permite que os dados em uma implementação do WebSphere eXtreme Scale sejam distribuídos em um conjunto de servidores (contêineres), com base em sua opção de configuração.

Dimensionamento de Memória e Cálculo de Contagem de Partições

É possível calcular a quantidade de memória e partições necessárias para sua configuração específica.

O WebSphere eXtreme Scale armazena dados no espaço de endereço das Java Virtual Machines (JVM). Cada JVM fornece espaço no processador para criar, recuperar, atualizar e excluir chamadas para os dados que estão armazenados na JVM. Além disso, cada JVM fornece espaço de memória para entradas de dados e réplicas. Objetos Java variam de tamanho, assim você deve medir para fazer uma estimativa de quanta memória você precisa.

Para dimensionar a memória necessária, carregue os dados do seu aplicativo em uma única JVM. Quando o uso do heap alcança 60%, observe o número de objetos que são utilizados. Este número é a contagem máxima recomendada de objetos para cada uma de suas Java Virtual Machines. Para obter o dimensionamento mais exato, utilize dados realistas e inclua quaisquer índices definidos em seu dimensionamento porque os índices também consomem memória. A melhor forma de dimensionar o uso da memória é executar a saída verbosegc de coleta de lixo pois esta saída fornece os números após a coleta de lixo. É possível consultar o uso do heap em qualquer ponto específico por meio de MBeans ou programaticamente, mas tais consultas fornecem apenas uma captura instantânea atual do heap, o que pode incluir lixo não coletado, sendo que, desta forma, o método não é uma indicação precisa da memória consumida.

Escalando a Configuração

Quantidade de shards por partição (numShardsPerPartition value)

Para calcular a quantidade de shards por partição, ou o valor de numShardsPerPartition, inclua 1 para o shard primário mais a quantidade total de shards de réplica que desejar.

```
numShardsPerPartition = 1 + total_number_of_replicas
```

Quantidade de Java Virtual Machines (valor minNumJVMs)

Para escalar sua configuração, primeiro, decida o número máximo de objetos necessários a serem armazenados no total. Para determinar a quantidade de Java Virtual Machines que precisa, use a seguinte fórmula:

```
minNumJVMS=(numShardsPerPartition * numObjs) / numObjsPerJVM
```

Arredonde este valor para cima para o valor de número inteiro mais próximo.

Quantidade de shards (valor numShards)

No tamanho de crescimento final, devem ser utilizados 10 shards para cada JVM. Conforme descrito antes, cada JVM possui um shard principal e (N-1) shards para as réplicas, ou neste caso, 9 réplicas. Como você já possui uma quantidade de Java Virtual Machines para armazenar os dados, é possível multiplicar a quantidade de Java Virtual Machines por 10 para determinar a quantidade de shards:

```
numShards = minNumJVMs * 10 shards/JVM
```

Número de partições

Se uma partição tiver um shard primário e um shard de réplica, então a partição possui dois shards (principal e de réplica). O número de partições é a contagem de shards dividido por 2, arredondado para o número primo mais próximo. Se a partição possui um primário e duas réplicas, então, o número de partições é a contagem de shards dividida por 3, arredondada para o número primo mais próximo.

```
numPartitions = numShards / numShardsPerPartition
```

Exemplo de Escala

Neste exemplo, o número de entrada inicia em 250 milhões. A cada ano, o número de entradas aumenta em cerca de 14%. Após 7 anos, o número total de entrada será de 500 milhões, portanto, planeje sua capacidade de acordo. Para alta disponibilidade, uma única réplica é necessária. Com uma réplica, o número de entradas duplica, ou seja, é de 1 bilhão de réplicas. Como um teste, 2 milhões de entradas podem ser armazenadas em cada JVM. Utilizando os cálculos neste cenário, a seguinte configuração é necessária:

- 500 Java Virtual Machines para armazenar o número final de entradas.
- 5000 shards, calculados ao multiplicar 500 Java Virtual Machines por 10.
- 2500 partições, ou 2503 como o próximo número primo mais alto, calculados ao obter 5000 shards, divididos por dois para shards primários e de réplica.

Iniciando a Configuração

Com base nos cálculos anteriores, você deveria iniciar com 250 Java Virtual Machines e crescer até 500 Java Virtual Machines ao longo de 5 anos, o que lhe permite gerenciar o crescimento incremental até chegar na quantidade final de entradas.

Nesta configuração, cerca de 200.000 entradas são armazenadas por partição (500 milhões de entradas divididas por 2503 partições). É necessário configurar o parâmetro numberOfBuckets no mapa que contém as entradas para o número primo mais alto e mais próximo, neste exemplo 70887, que mantém a proporção por volta de 3.

Quando o número máximo de Java Virtual Machines é atingido

Quando você atinge seu número máximo de 500 Java Virtual Machines, ainda pode aumentar sua grade. À medida que o número de Java Virtual Machines aumenta além de 500, a contagem de shards começa a diminuir abaixo de 10 para cada JVM, que está abaixo do número recomendado. Os shards começam a ficar maiores, o que pode causar problemas. Você deve repetir o processo de dimensionamento considerando novamente o crescimento futuro e reconfigurar a contagem da partição. Esta prática requer um reinício total da grade ou uma interrupção da sua grade.

Número de Servidores

Atenção: Não utilize paginação em um servidor sob nenhuma circunstância.

Uma única JVM utiliza mais memória do que o tamanho do heap. Por exemplo, 1 GB de heap para uma JVM na verdade utiliza 1.4 GB de memória real. Determine a RAM livre disponível no servidor. Divida a quantidade de RAM pela memória por JVM para obter o número máximo da Java Virtual Machines no servidor.

Dimensionando a CPU por Partição para Transações

Embora a maior funcionalidade do eXtreme Scale seja sua capacidade de efetuar escala elástica, também é importante considerar o dimensionamento e o ajuste do número ideal de CPUs para efetuar scale up.

Os custos do processador incluem:

- Custo de serviços de operações de criação, recuperação, atualização e exclusão de clientes.
- Custo de replicação da outra Java Virtual Machines.
- Custo de invalidação.
- Custo da política de despejo.
- Custo da coleta de lixo.
- Custo da lógica de aplicativo.

Java Virtual Machines por servidor

Utilize dois servidores e inicie a contagem máxima de JVM por servidor. Utilize as contagens de partição calculadas da seção anterior. Em seguida, pré-carregue a Java Virtual Machines com dados suficientes para ajuste apenas nestes dois computadores. Utilize um servidor separado como um cliente. Execute uma simulação de transação realista junto a esta grade de dois servidores.

Para calcular a linha de base, tente saturar o uso do processador. Se não for possível saturar o processador, então, é provável que a rede esteja saturada. Se a rede estiver saturada, inclua mais cartões de rede e execute o round robin da Java Virtual Machines sobre os vários cartões de rede.

Execute os computadores em 60% de uso do processador, meça a taxa de transações de criação, recuperação, atualização e exclusão. Esta medida fornece o rendimento nos dois servidores. Este número duplica com quatro servidores, duplica novamente em 8 servidores e assim por diante. Esta escala assume que a capacidade da rede e a capacidade do cliente também tenham a capacidade de escalar.

Como resultado, o tempo de resposta do eXtreme Scale deve ser estável à medida que o número de servidores é escalado para cima. O rendimento da transação deve escalar linearmente à medida que os computadores são incluídos à grade.

Dimensionando CPUs para Transações Paralelas

As transações de partição única possuem escala de rendimento linear à medida que a grade cresce. Transações paralelas são diferentes de transações de partição única, pois elas acessam um conjunto de servidores (pode ser todos os servidores).

Se uma transação acessar todos os servidores, o rendimento será limitado ao rendimento do cliente que inicia a transação ou ao servidor mais lento que está sendo acessado. Grades maiores propagam mais os dados e fornecem mais espaço do processador, memória, rede e assim por diante. Entretanto, o cliente deve aguardar que o servidor mais lento responda, e o cliente deve consumir os resultados da transação.

Quando uma transação acessa um subconjunto de servidores, M entre N servidores obtêm uma solicitação. O rendimento é então dividido N por M vezes mais rápido do que o rendimento do servidor mais lento. Por exemplo, se você tiver 20 servidores e uma transação que acessa 5 servidores, então, o rendimento é de 4 vezes o rendimento do servidor mais lento na grade.

Quando uma transação paralela é concluída, os resultados são enviados para o encadeamento do cliente que iniciou a transação. Este cliente deve então agregar os resultados únicos encadeados. Este tempo de agregação aumenta à medida que os número de servidores acessados pela transação aumentam. Entretanto, este tempo depende do aplicativo porque é possível que cada servidor retorne um resultado menor à medida que a grade aumenta.

Normalmente, as transações paralelas acessam todos os servidores na grade porque as partições são distribuídas de maneira uniforme pela grade. Neste caso, o rendimento é limitado ao primeiro caso.

Resumo

Com este dimensionamento, são obtidas três métricas como a seguir:

- Número de partições.
- Número de servidores que são necessários para a memória que é necessária.
- Número de servidores que são necessários para o rendimento necessário.

Se você precisar de 10 servidores para requisitos de memória mas estiver obtendo apenas 50% do rendimento necessário devido à saturação do processador, então precisa do dobro de servidores.

Para maior estabilidade, você deve executar seus servidores a um carregamento de processador de 60% e JVMheaps a um carregamento de heap de 60%. Os picos podem então conduzir o uso do processador a 80-90%, mas não execute seus servidores regularmente em níveis superiores a este.

Planejamento de Capacidade e Alta Disponibilidade (Armazenamento em Cache Dinâmico)

A API de Cache Dinâmico está disponível para os aplicativos Java EE que são implementados no WebSphere Application Server. O cache dinâmico pode ser potencializado para dados de negócios em cache, HTML gerado ou para sincronizar os dados em cache na célula usando o data replication service (DRS).

Visão Geral

Todas as instâncias de cache dinâmico criadas com o provedor de cache dinâmico do WebSphere eXtreme Scale são altamente disponíveis por padrão. O nível e custo de memória de alta disponibilidade depende da topologia usada.

Ao usar a topologia integrada, o tamanho do cache é limitado à quantidade de memória livre em um único processo do servidor, e cada processo do servidor armazena uma cópia completa do cache. Enquanto um único processo do servidor continua a executar, o cache sobrevive. Os dados do cache somente serão perdidos se todos os servidores que acessam o cache forem desligados.

Para armazenamento em cache que usa topologia particionada integrada, o tamanho do cache é limitado a um agregado do espaço livre disponível em todos os processos do servidor. Por padrão, o provedor de cache dinâmico do eXtreme Scale usa 1 réplica para cada shard primário, assim cada parte dos dados armazenados em cache é armazenada duas vezes.

Use a seguinte fórmula A para determinar a capacidade de um cache particionado integrado.

Fórmula A

F * C / (1 + R) = M

Em que:

- F = Memória livre por processo de contêiner
- C = número de contêineres
- R = número de réplicas
- M = Tamanho total do cache

Para uma grade de Implementação de Rede do WebSphere que tenha 256 MB de espaço disponível em cada processo, com 4 processos do servidor no total, uma instância de cache por todos esses servidores poderia armazenar até 512 megabytes de dados. Deste modo, o cache pode sobreviver a um dano no servidor sem perder dados. Também, até dois servidores poderiam ser desligados sequencialmente sem perda de dado algum. Assim, para o exemplo anterior, a fórmula é a seguinte:

256mb * 4 contêineres/ (1 primário + 1 réplica) = 512mb.

Caches que usam a topologia remota têm características de dimensionamento similares às dos caches que usam topologia particionada integrada, mas eles são limitados pela quantidade de espaço disponível em todos os processos de contêiner do eXtreme Scale.

Em topologias remotas, é possível aumentar o número de réplicas para fornecer um nível mais alto de disponibilidade ao custo de gasto adicional de memória. Na maioria dos aplicativos com cachê dinâmico isso seria desnecessário, mas é possível editar o arquivo dynacache-remote-deployment.xml para aumentar o número de réplicas.

Use as seguintes fórmulas, B e C, para determinar o efeito da inclusão de mais réplicas na Alta Disponibilidade do cache.

Fórmula B

N = Mínimo(T -1, R)

Em que:
- N = o número de processos que podem travar simultaneamente
- T = o número total de contêineres
- R = o número total de réplicas

Fórmula C

Limite(T/ (1+N)) = m

Em que:
- T = Número total de contêineres
- N = Número total de réplicas
- m = número mínimo de contêineres necessários para suportar os dados em cache.

Para o ajuste de desempenho com o provedor de cache dinâmico, consulte Ajustando o Provedor de Cache Dinâmico.

Dimensionamento do Cache

Antes que um aplicativo que usa o provedor de Cache Dinâmico do WebSphere eXtreme Scale possa ser implementado, os princípios gerais descritos na seção anterior devem ser combinados com os dados ambientais para os sistemas de produção. O primeiro valor a estabelecer é o número total de processos do contêiner e a quantidade de memória disponível em cada processo para conter os dados do cache. Ao usar a topologia integrada, os contêineres de cache serão colocados dentro dos processos do servidor do WebSphere Application, assim, há um contêiner para cada servidor que estiver compartilhando o cache. Determinar o gasto adicional de memória do aplicativo sem o armazenamento em cache ativado e o WebSphere Application Server é a melhor maneira de descobrir quanto espaço está disponível no processo. Isso pode ser feito por meio de análise detalhada dos dados da coleta de lixo. Ao usar a topologia remota, essas informações podem ser localizadas por meio da verificação da saída detalhada da coleta de lixo de um contêiner independente recentemente iniciado que ainda não foi preenchido com dados do cache. A última coisa a lembrar para descobrir quanto espaço por processo está disponível para dados em cache, é reservar algum espaço de heap para a coleta de lixo. O gasto adicional do contêiner, o WebSphere Application Server ou independente, mais o tamanho reservado para o cache não deve ser maior que 70% do heap total.

Assim que essas informações são coletadas, os valores podem ser inseridos na fórmula A, descrita anteriormente, para determinar o tamanho máximo para o cache particionado. Depois que o tamanho máximo é conhecido, a próxima etapa é determinar o número total de entradas do cache que pode ser suportado, o que requer determinar o tamanho médio por entrada do cache. A maneira mais simples de fazer isso é incluir 10% ao tamanho do objeto do cliente. Consulte o Guia de Ajuste para cache dinâmico e serviço de replicação de dados para obter informações mais detalhadas sobre dimensionamento de entradas do cache ao usar o Cache Dinâmico.

Quando a compactação está ativada, ela afeta o tamanho do objeto do cliente, não o gasto adicional do sistema de armazenamento em cache. Use a fórmula a seguir para determinar o tamanho de um objeto em cache ao usar a compactação:

S = O * C + O * 0.10

Em que:

- S = Tamanho médio do objeto em cache
- O = Tamanho médio do objeto do cliente não-compactado
- C = Proporção de compactação expressa como uma fração.

Assim, uma proporção de compactação 2 para 1 é 1/2 = 0.50. Para este valor, quanto menor, melhor. Se o objeto que está sendo armazenado é um POJO normal cheio principalmente tipos primitivos, então assuma uma proporção de compactação de 0.60 para 0.70. Se o objeto em cache é um objeto Servlet, JSP ou WebServices, o método ideal para determinar a taxa de compactação é compactar uma amostra representativa com um utilitário de compactação ZIP. Se isso não for possível, então uma proporção de compactação de 0.2 para 0.35 é comum para esses tipos de dados.

Depois, use estas informações para determinar o número total de entradas do cache que pode ser suportado. Use a seguinte fórmula D:

Fórmula D

T = S / A

Em que:

- T= Número total de entradas do cache
- S = Tamanho total disponível para dados em cache conforme computados usando a fórmula A
- A = Tamanho médio de cada entrada do cache

Finalmente, você deve configurar o tamanho do cache na instância do cache dinâmico para impor este limite. O provedor de cache dinâmico do WebSphere eXtreme Scale difere do provedor de cache dinâmico padrão neste aspecto. Use a fórmula a seguir para determinar o valor para definir o tamanho do cache na instância do cache dinâmico. Use a seguinte fórmula E:

Fórmula E

Cs = Ts / Np

Em que:

- Ts = Tamanho de destino total para o cache

- Cs = Configuração do Tamanho do Cache para definir na instância do cache dinâmico
- Np = Número de partições. O padrão é 47.

Configure o tamanho da instância do cache dinâmico para um valor calculado pela fórmula E em cada servidor que compartilhar a instância do cache.

Capítulo 3. Instalando e Implementando WebSphere eXtreme Scale

O WebSphere eXtreme Scale é uma grade de dados de memória que pode ser usada para particionar, replicar e gerenciar dados e lógica de negócios dinamicamente por meio de vários servidores. Após determinar os propósitos e os requisitos da sua implementação, instale o eXtreme Scale em seu sistema.

Antes de Iniciar

- Estabeleça como o WebSphere eXtreme Scale se enquadra na sua topologia atual. Consulte WebSphere eXtreme Scale visão geral de arquitetura e topologia na *Visão Geral do Produto* para obter mais informações.
- Verifique se seu ambiente atende aos pré-requisitos para instalar o eXtreme Scale. Consulte "Requisitos de Hardware e Software" na página 63 para obter mais informações.

Sobre Esta Tarefa

7.1+ Dois tipos de instalação

- Instalação completa do WebSphere eXtreme Scale: É possível usar esta instalação para instalar o servidor, o cliente ou ambos.
- Instalação do WebSphere eXtreme Scale Client: É possível usar esta instalação para instalar o cliente em plataformas específicas.

Ambientes com Suporte

Não é necessário instalar e implementar o eXtreme Scale em um nível específico do sistema operacional. Cada instalação do Java Platform, Standard Edition (J2SE) e Java Platform, Enterprise Edition (JEE) requer correções ou níveis de sistema operacional diferentes.

É possível instalar e implementar o produto nos ambientes do JEE e do J2SE. Também é possível incluir em pacote configurável o componente do cliente com os aplicativos JEE diretamente sem integrar com o WebSphere Application Server. O WebSphere eXtreme Scale suporta o Java Runtime Environment (JRE) Versão 1.4.2 e superior e o WebSphere Application Server Versão 6.0.2 e superior.

Procedimento

- Instale o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client independente.

  É possível instalar o eXtreme Scale independente em um ambiente que não contenha o WebSphere Application Server ou o WebSphere Application Server Network Deployment. Com a opção independente, você define um novo local de instalação para instalar o servidor eXtreme Scale.

Atenção: Também é possível utilizar um perfil não raiz (não administrador) para o WebSphere eXtreme Scale em um ambiente independente. Um ambiente independente é um ambiente que não está utilizando WebSphere Application Server. Para utilizar um perfil não raiz, você deverá alterar o proprietário do diretório `ObjectGrid` para o perfil não raiz. Em seguida, é possível efetuar login com esse perfil não raiz e operar o eXtreme Scale como faria normalmente para um perfil raiz (administrador).

- Integre o produto com o WebSphere Application Server ou o WebSphere Application Server Network Deployment.

  É possível instalar e integrar o eXtreme Scale com uma instalação existente do WebSphere Application Server ou do WebSphere Application Server Network Deployment. Com a instalação completa, é possível selecionar instalar o cliente e o servidor do eXtreme Scale ou pode instalar apenas o cliente.
- Crie e aumente perfis.

  Crie e aumente os perfis para usar os recursos do eXtreme Scale. Se você estiver executando o WebSphere Application Server Versão 6.1 ou Versão 7.0, é possível usar o plug-in Profile Management Tool ou o comando manageprofiles. Se você estiver executando o WebSphere Application Server Versão 6.0.2, é necessário usar o comando wasprofile para criar e aumentar os perfis.
- Aplique a manutenção.

  Use o IBM® Update Installer Versão 7.0.0.4 ou superior para aplicar manutenção no seu ambiente.

Migrando para o WebSphere eXtreme Scale Versão 7.1

Com o instalador do WebSphere eXtreme Scale, não é possível fazer upgrade ou modificar uma instalação anterior. É necessário desinstalar a versão anterior antes de instalar a nova. Não é necessário migrar seus arquivos de configuração porque eles são compatíveis com versões anteriores. No entanto, se você tiver alterado qualquer um dos arquivos de script que são enviados com o produto, será necessário reaplicar essas mudanças para os arquivos de script atualizados.

Antes de Iniciar

Verifique se seus sistemas atendem aos requisitos mínimos para as versões do produto que você pretende migrar e instalar. Consulte “Requisitos de Hardware e Software” na página 63 para obter mais informações.

Sobre Esta Tarefa

Mescle quaisquer arquivos de script de produto modificados com os novos arquivos de script do produto no diretório `/bin` para manter as alterações.

Dica: Se você não modificou os arquivos de script que estão instalados com o produto, não será necessário concluir as seguintes etapas de migração. Em vez disso, é possível fazer upgrade para a Versão 7.1 desinstalando a versão anterior e instalando a nova versão no mesmo diretório.

Procedimento

1. Pare todos os processos que usam o eXtreme Scale.
   - Leia sobre como parar servidores independentes para parar todos os processos que estão em execução no ambiente independente do eXtreme Scale.

- Leia sobre utilitários de linha de comandos para parar todos os processos que estão em execução em seu ambiente do WebSphere Application Server ou WebSphere Application Server Network Deployment.

2. Salve quaisquer scripts modificados a partir do diretório de instalação atual em um diretório temporário.
3. Desinstale o produto.
4. Instale o eXtreme Scale Versão 7.1. Consulte Capítulo 3, “Instalando e Implementando WebSphere eXtreme Scale”, na página 17 para obter informações adicionais.
5. Mescle suas alterações a partir dos arquivos no diretório temporário para os novos arquivos de script do produto no diretório `/bin`.
6. Inicie todos os processos do eXtreme Scale para começar a usar o produto. Consulte “Terminologia Administrativa” na página 317 para obter mais informações.

Instalando WebSphere eXtreme Scale ou WebSphere eXtreme Scale Client Independente

É possível instalar o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client independente em um ambiente que não contenha o WebSphere Application Server ou o WebSphere Application Server Network Deployment.

Antes de Iniciar

- Verifique se o diretório de instalação de destino está vazio ou não existe.

  Importante: Se uma versão anterior do WebSphere eXtreme Scale ou componente ObjectGrid existir no diretório que você especificar para instalar a Versão 7.1, o produto não será instalado. Por exemplo, é possível ter uma pasta `<wxs_install_root>/ObjectGrid` previamente existente. É possível selecionar um diretório de instalação diferente ou cancelar a instalação. Em seguida, remova a instalação anterior e execute o assistente novamente.

- **7.1+** Um IBM Runtime Environment é instalado como parte da instalação independente na pasta *<wxs_install_home>*`/java`.

Sobre Esta Tarefa

Ao instalar o produto como independente, você instala o cliente e o servidor do WebSphere eXtreme Scale de forma independente. Com a instalação do WebSphere eXtreme Scale Client no modo independente, você está instalando um cliente para acessar os dados nas grades de dados. Os processos do servidor e do cliente, portanto, acessam todos os recursos necessários localmente. Também é possível integrar o WebSphere eXtreme Scale em aplicativos Java Platform, Standard Edition (J2SE) existentes utilizando scripts e arquivos Java archive (JAR).

Tabela 1. Arquivos de Tempo de Execução para Instalação Completa do WebSphere eXtreme Scale. O WebSphere eXtreme Scale conta com processos do ObjectGrid e APIs relacionadas. A seguinte tabela lista os arquivos JAR incluídos na instalação.

Nome do arquivo	Ambiente	Local da instalação	Descrição
`wxsdynacache.jar`	Cliente e Provedor	`dynacache/lib`	O arquivo `wxsdynacache.jar` contém as classes necessárias a serem usadas com o provedor de cache dinâmico. O arquivo é incluído automaticamente no ambiente de tempo de execução do servidor quando você utiliza os scripts fornecidos.

Tabela 1. Arquivos de Tempo de Execução para Instalação Completa do WebSphere eXtreme Scale (continuação). O WebSphere eXtreme Scale conta com processos do ObjectGrid e APIs relacionadas. A seguinte tabela lista os arquivos JAR incluídos na instalação.

Nome do arquivo	Ambiente	Local da instalação	Descrição
wxshyperic.jar	Funções de Utilitário	hyperic/lib	O plug-in de detecção do servidor WebSphere eXtreme Scale para o agente de monitoramento SpringSource Hyperic.
objectgrid.jar	Local, cliente e servidor	lib	O arquivo objectgrid.jar é utilizado pelo ambiente de tempo de execução do servidor do J2SE Versão 1.4.2 e posterior. O arquivo é incluído automaticamente no ambiente de tempo de execução do servidor quando você utiliza os scripts fornecidos.
ogagent.jar	Local, cliente e servidor	lib	O arquivo ogagent.jar contém as classes do tempo de execução que são necessárias para executar o agente de instrumentação Java que é usado com a API do EntityManager.
ogclient.jar	Local e cliente	lib	O arquivo ogclient.jar contém apenas os ambientes de tempo de execução local e do cliente. É possível utilizar este arquivo com o J2SE Versão 1.4.2 e posterior.
ogspring.jar	Local, cliente e servidor	lib	O arquivo ogspring.jar contém classes de suporte para a integração de estrutura SpringSource Spring.
wsogclient.jar	Local e cliente	lib	O arquivo wsogclient.jar é instalado quando você utiliza um ambiente contendo o WebSphere Application Server Versão 6.0.2 e posterior. Esse arquivo contém apenas os ambientes de tempo de execução local e do cliente.
wxssizeagent.jar	Local, cliente e servidor	lib	O arquivo wxssizeagent.jar é usado para fornecer informações mais precisas de dimensionamento de entrada de cache ao usar o Java Runtime Environment (JRE) Versão 1.5 ou mais recente.
ibmcfw.jar ibmext.jar ibmorb.jar ibmorbapi.jar	Cliente e Provedor	lib/endorsed	Este conjunto de arquivos inclui o tempo de execução Object Request Broker (ORB) que é usado para executar aplicativos em processos Java SE.
restservice.ear	Cliente	restservice/lib	O arquivo restservice.ear contém o archive corporativo de aplicativo de serviço de dados eXtreme Scale REST para ambientes WebSphere Application Server.
restservice.war	Cliente	restservice/lib	O arquivo restservice.war contém o arquivamento Web de serviço de dados eXtreme Scale REST para servidores de aplicativos adquiridos de outro fornecedor.
xsadmin.jar	Funções de Utilitário	samples	O arquivo xsadmin.jar contém o utilitário de amostra de administração eXtreme Scale.
sessionobjectgrid.jar	Cliente e Provedor	session/lib	O arquivo sessionobjectgrid.jar contém o tempo de execução de gerenciamento de sessões eXtreme Scale HTTP.
splicerlistener.jar	Funções de Utilitário	session/lib	O arquivo splicerlistener.jar contém o utilitário splicer para o listener de sessão HTTP do eXtreme Scale Versão 7.1.
xsgbean.jar	Servidor	wasce/lib	O arquivo xsgbean.jar contém o GBean para integrar servidores eXtreme Scale nos servidores de aplicativos WebSphere Application Server Community Edition.
splicer.jar	Funções de Utilitário	legacy/session/lib	O utilitário splicer para o filtro do gerenciador de sessões HTTP do WebSphere eXtreme Scale Versão 7.0.

Tabela 2. Arquivos de Tempo de Execução para o WebSphere eXtreme Scale Client. O WebSphere eXtreme Scale Client conta com processos do ObjectGrid e APIs relacionadas. A seguinte tabela lista os arquivos JAR incluídos na instalação.

Nome do arquivo	Ambiente	Local da instalação	Descrição
`wxsdynacache.jar`	Cliente e Provedor	`dynacache/lib`	O arquivo `wxsdynacache.jar` contém as classes necessárias a serem usadas com o provedor de cache dinâmico. O arquivo é incluído automaticamente no ambiente de tempo de execução do servidor quando você utiliza os scripts fornecidos.
`wxshyperic.jar`	Funções de Utilitário	`hyperic/lib`	O plug-in de detecção do servidor WebSphere eXtreme Scale para o agente de monitoramento SpringSource Hyperic.
`ogagent.jar`	Local, cliente e servidor	`lib`	O arquivo `ogagent.jar` contém as classes do tempo de execução que são necessárias para executar o agente de instrumentação Java que é usado com a API do EntityManager.
`ogclient.jar`	Local e cliente	`lib`	O arquivo `ogclient.jar` contém apenas os ambientes de tempo de execução local e do cliente. É possível utilizar este arquivo com o J2SE Versão 1.4.2 e posterior.
`ogspring.jar`	Local, cliente e servidor	`lib`	O arquivo `ogspring.jar` contém classes de suporte para a integração de estrutura SpringSource Spring.
`wsogclient.jar`	Local e cliente	`lib`	O arquivo `wsogclient.jar` é instalado quando você utiliza um ambiente contendo o WebSphere Application Server Versão 6.0.2 e posterior. Esse arquivo contém apenas os ambientes de tempo de execução local e do cliente.
`wxssizeagent.jar`	Local, cliente e servidor	`lib`	O arquivo `wxssizeagent.jar` é usado para fornecer informações mais precisas de dimensionamento de entrada de cache ao usar o Java Runtime Environment (JRE) Versão 1.5 ou mais recente.

`ibmcfw.jar`
`ibmext.jar`
`ibmorb.jar`
`ibmorbapi.jar`
	Cliente e Provedor	`lib/endorsed`	Este conjunto de arquivos inclui o tempo de execução Object Request Broker (ORB) que é usado para executar aplicativos em processos Java SE.

`restservice.ear`	Cliente	`restservice/lib`	O arquivo `restservice.ear` contém o archive corporativo de aplicativo de serviço de dados eXtreme Scale REST para ambientes WebSphere Application Server.
`restservice.war`	Cliente	`restservice/lib`	O arquivo `restservice.war` contém o arquivamento Web de serviço de dados eXtreme Scale REST para servidores de aplicativos adquiridos de outro fornecedor.
`xsadmin.jar`	Funções de Utilitário	`samples`	O arquivo `xsadmin.jar` contém o utilitário de amostra de administração eXtreme Scale.
`sessionobjectgrid.jar`	Cliente e Provedor	`session/lib`	O arquivo `sessionobjectgrid.jar` contém o tempo de execução de gerenciamento de sessões eXtreme Scale HTTP.
`splicerlistener.jar`	Funções de Utilitário	`session/lib`	O arquivo `splicerlistener.jar` contém o utilitário splicer para o listener de sessão HTTP do eXtreme Scale Versão 7.1.
`splicer.jar`	Funções de Utilitário	`legacy/session/lib`	O utilitário splicer para o filtro do gerenciador de sessões HTTP do WebSphere eXtreme Scale Versão 7.0.

Procedimento

1. Utilize o assistente para instalar.
   - Execute o seguinte script para iniciar o assistente para a instalação completa do WebSphere eXtreme Scale:
     - Linux UNIX *dvd_root*`/install`

– Windows *dvd_root*\install.bat

- Execute o seguinte script para iniciar o assistente para a instalação do WebSphere eXtreme Scale Client:
  – Linux UNIX root/*WXS_Client*/install
  – Windows root*WXS_Client*\install.bat

2. Siga os prompts no assistente e clique em **Concluir**.

   Restrição: O painel de recursos opcionais lista os recursos dos quais você pode selecionar para instalar. Porém, os recursos não poderão ser incluídos incrementalmente no ambiente do produto depois que o produto for instalado. Se você escolher não instalar um recurso com a instalação inicial do produto, deverá desinstalar e reinstalar o produto para incluir o recurso.

Resultados

Windows Se estiver instalando o WebSphere eXtreme Scale Client no Windows®, será possível ver o seguinte texto nos resultados da instalação:

```
Sucesso: A instalação do seguinte produto foi bem-sucedida:
WebSphere eXtreme Scale Client.  Algumas etapas de configuração têm erros.
Para obter mais informações, consulte o seguinte arquivo de log:
<WebSphere Application Server install root>\logs\wxs_client\install\log.txt"
Revise o log de instalação (log.txt) e o log de aumento do gerenciador de implementação
.
```

Se vir uma falha com o arquivo `iscdeploy.sh`, é possível ignorar o erro. Este erro não causa nenhum problema.

O que Fazer Depois

Leia sobre a configuração do eXtreme Scale para configurar os processos do aplicativo cliente e os processos do servidor.

Integrando o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client com o WebSphere Application Server

É possível instalar o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client em um ambiente no qual o WebSphere Application Server or WebSphere Application Server Network Deployment esteja instalado. É possível utilizar os recursos existentes do WebSphere Application Server or WebSphere Application Server Network Deployment para aprimorar os aplicativos do eXtreme Scale.

Antes de Iniciar

- Instale WebSphere Application Server ou WebSphere Application Server Network Deployment. Consulte Instalando o Ambiente de Serviço do Aplicativo para obter mais informações.
- Com base na versão instalada, Versão 6.0.x, Versão 6.1 ou Versão 7.0, aplique o fix pack mais recente para o WebSphere Application Server ou o WebSphere Application Server Network Deployment para atualizar o nível do produto. Consulte os Fix Packs mais Recentes para o WebSphere Application Server para obter mais informações.
- Verifique se o diretório de instalação de destino não contém uma instalação existente do WebSphere eXtreme Scale ou do WebSphere eXtreme Scale Client.

- Pare todos os processos em execução no ambiente do WebSphere Application Server ou do WebSphere Application Server Network Deployment. Consulte Usando ferramentas de linha de comandos de script wsadmin para obter mais informações sobre os comandos stopManager, stopNode e stopServer.

  CUIDADO:
  Certifique-se de que quaisquer processos em execução sejam interrompidos. Se os processos em execução não forem interrompidos, a instalação continuará, criando resultados imprevisíveis e deixando a instalação em um estado indeterminado em algumas plataformas.

Importante: Quando você instala o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client, ele deve estar no mesmo diretório no qual você instalou o WebSphere Application Server. Por exemplo, se tiver instalado o WebSphere Application Server em `C:\<was_root>`, você deverá também escolher `C:<was_root>` como o diretório de destino para sua instalação do WebSphere eXtreme Scale ou do WebSphere eXtreme Scale Client.

Sobre Esta Tarefa

Integre o eXtreme Scale com o WebSphere Application Server ou WebSphere Application Server Network Deployment para aplicar os recursos do eXtreme Scale nos aplicativos do Java Platform, Enterprise Edition. Os aplicativos Java EE hospedam grades de dados e acessam as grades de dados usando uma conexão do cliente.

Tabela 3. Arquivos de Tempo de Execução para o WebSphere eXtreme Scale. A seguinte tabela lista os arquivos Java archive (JAR) que são incluídos na instalação.

Nome do arquivo	Ambiente	Local da instalação	Descrição
`wxsdynacache.jar`	Cliente e Provedor	`lib`	O arquivo `wxsdynacache.jar` contém as classes necessárias a serem usadas com o provedor de cache dinâmico.
`wsobjectgrid.jar`	Local e cliente	`lib`	O `wsobjectgrid.jar` contém os tempos de execução local, do cliente e do servidor do eXtreme Scale.
`ogagent.jar`	Local, cliente e servidor	`lib`	O arquivo `ogagent.jar` contém as classes do tempo de execução que são necessárias para executar o agente de instrumentação Java que é usado com a API do EntityManager.
`ogsip.jar`	Servidor	`lib`	O arquivo `ogsip.jar` contém o tempo de execução de gerenciamento de sessões SIP (Session Initiation Protocol) do eXtreme Scale que é compatível com o WebSphere Application Server Versão 6.1.x.
`sessionobjectgrid.jar`	Cliente e servidor	`lib`	O arquivo `sessionobjectgrid.jar` contém o tempo de execução de gerenciamento de sessões eXtreme Scale HTTP.
`sessionobjectgridsip.jar`	Servidor	`lib`	O arquivo `sessionobjectgridsip.jar` contém o tempo de execução de gerenciamento de sessões SIP do eXtreme Scale que é compatível com o WebSphere Application Server Versão 7.x.
`wsogclient.jar`	Local e cliente	`lib`	O arquivo `wsogclient.jar` é instalado quando você utiliza um ambiente contendo o WebSphere Application Server Versão 6.0.2 e posterior. Esse arquivo contém apenas os ambientes de tempo de execução local e do cliente.
`wxssizeagent.jar`	Local, cliente e servidor	`lib`	O arquivo `wxssizeagent.jar` é usado para fornecer informações mais precisas de dimensionamento de entrada de cache ao usar o Java Runtime Environment (JRE) Versão 1.5 ou mais recente.
`oghibernate-cache.jar`	Cliente e Provedor	`optionalLibraries/ObjectGrid`	O arquivo `oghibernate-cache.jar` contém o plug-in de cache de nível 2 do eXtreme Scale para JBoss Hibernate.
`ogspring.jar`	Local, cliente e servidor	`optionalLibraries/ObjectGrid`	O arquivo `ogspring.jar` contém classes de suporte para a integração de estrutura SpringSource Spring.

Tabela 3. Arquivos de Tempo de Execução para o WebSphere eXtreme Scale (continuação). A seguinte tabela lista os arquivos Java archive (JAR) que são incluídos na instalação.

Nome do arquivo	Ambiente	Local da instalação	Descrição
xsadmin.jar	Funções de Utilitário	optionalLibraries/ObjectGrid	O arquivo xsadmin.jar contém o utilitário de amostra de administração eXtreme Scale.
ibmcfw.jar ibmext.jar ibmorb.jar ibmorbapi.jar	Cliente e Provedor	optionalLibraries/ObjectGrid/endorsed	Este conjunto de arquivos inclui o tempo de execução Object Request Broker (ORB) que é usado para executar aplicativos em processos Java SE.
wxshyperic.jar	Funções de Utilitário	optionalLibraries/ObjectGrid/hyperic/lib	O plug-in de detecção do servidor WebSphere eXtreme Scale para o agente de monitoramento SpringSource Hyperic.
restservice.ear	Cliente	optionalLibraries/ObjectGrid/restservice/lib	O arquivo restservice.ear contém o archive corporativo de aplicativo de serviço de dados eXtreme Scale REST para ambientes WebSphere Application Server.
restservice.war	Cliente	optionalLibraries/ObjectGrid/restservice/lib	O arquivo restservice.war contém o arquivamento Web de serviço de dados eXtreme Scale REST para servidores de aplicativos adquiridos de outro fornecedor.
splicerlistener.jar	Funções de Utilitário	optionalLibraries/ObjectGrid/session/lib	O arquivo splicerlistener.jar contém o utilitário splicer para o filtro do gerenciador de sessões HTTP do eXtreme Scale.
splicer.jar	Funções de Utilitário	optionalLibraries/ObjectGrid/legacy/session/lib	O splicer.jar contém o utilitário splicer Versão 7.0 para o filtro do gerenciador de sessões HTTP do eXtreme Scale.

Tabela 4. Arquivos de Tempo de Execução para o WebSphere eXtreme Scale Client. A seguinte tabela lista os arquivos Java archive (JAR) que são incluídos na instalação.

Nome do arquivo	Ambiente	Local da instalação	Descrição
wxsdynacache.jar	Cliente e Provedor	lib	O arquivo wxsdynacache.jar contém as classes necessárias a serem usadas com o provedor de cache dinâmico.
ogagent.jar	Local, cliente e servidor	lib	O arquivo ogagent.jar contém as classes do tempo de execução que são necessárias para executar o agente de instrumentação Java que é usado com a API do EntityManager.
ogsip.jar	Servidor	lib	O arquivo ogsip.jar contém o tempo de execução de gerenciamento de sessões SIP (Session Initiation Protocol) do eXtreme Scale que é compatível com o WebSphere Application Server Versão 6.1.x.
sessionobjectgrid.jar	Cliente e servidor	lib	O arquivo sessionobjectgrid.jar contém o tempo de execução de gerenciamento de sessões eXtreme Scale HTTP.
sessionobjectgridsip.jar	Servidor	lib	O arquivo sessionobjectgridsip.jar contém o tempo de execução de gerenciamento de sessões SIP do eXtreme Scale que é compatível com o WebSphere Application Server Versão 7.x.
wsogclient.jar	Local e cliente	lib	O arquivo wsogclient.jar é instalado quando você utiliza um ambiente contendo o WebSphere Application Server Versão 6.0.2 e posterior. Esse arquivo contém apenas os ambientes de tempo de execução local e do cliente.
wxssizeagent.jar	Local, cliente e servidor	lib	O arquivo wxssizeagent.jar é usado para fornecer informações mais precisas de dimensionamento de entrada de cache ao usar o Java Runtime Environment (JRE) Versão 1.5 ou mais recente.
oghibernate-cache.jar	Cliente e Provedor	optionalLibraries/ObjectGrid	O arquivo oghibernate-cache.jar contém o plug-in de cache de nível 2 do eXtreme Scale para JBoss Hibernate.
ogspring.jar	Local, cliente e servidor	optionalLibraries/ObjectGrid	O arquivo ogspring.jar contém classes de suporte para a integração de estrutura SpringSource Spring.

Tabela 4. Arquivos de Tempo de Execução para o WebSphere eXtreme Scale Client (continuação). A seguinte tabela lista os arquivos Java archive (JAR) que são incluídos na instalação.

Nome do arquivo	Ambiente	Local da instalação	Descrição
`xsadmin.jar`	Funções de Utilitário	`optionalLibraries/ObjectGrid`	O arquivo `xsadmin.jar` contém o utilitário de amostra de administração eXtreme Scale.

`ibmcfw.jar`
`ibmext.jar`
`ibmorb.jar`
`ibmorbapi.jar`
	Cliente e Provedor	`optionalLibraries/ObjectGrid/endorsed`	Este conjunto de arquivos inclui o tempo de execução Object Request Broker (ORB) que é usado para executar aplicativos em processos Java SE.

`wxshyperic.jar`	Funções de Utilitário	`optionalLibraries/ObjectGrid/hyperic/lib`	O plug-in de detecção do servidor WebSphere eXtreme Scale para o agente de monitoramento SpringSource Hyperic.
`restservice.ear`	Cliente	`optionalLibraries/ObjectGrid/restservice/lib`	O arquivo `restservice.ear` contém o archive corporativo de aplicativo de serviço de dados eXtreme Scale REST para ambientes WebSphere Application Server.
`restservice.war`	Cliente	`optionalLibraries/ObjectGrid/restservice/lib`	O arquivo `restservice.war` contém o arquivamento Web de serviço de dados eXtreme Scale REST para servidores de aplicativos adquiridos de outro fornecedor.
`splicerlistener.jar`	Funções de Utilitário	`optionalLibraries/ObjectGrid/session/lib`	O arquivo `splicerlistener.jar` contém o utilitário splicer para o filtro do gerenciador de sessões HTTP do eXtreme Scale.
`splicer.jar`	Funções de Utilitário	`optionalLibraries/ObjectGrid/legacy/session/lib`	O `splicer.jar` contém o utilitário splicer Versão 7.0 para o filtro do gerenciador de sessões HTTP do eXtreme Scale.

Procedimento

1. Use o assistente para concluir a instalação.
   - Execute o seguinte script para iniciar o assistente para a instalação completa do WebSphere eXtreme Scale:
     - Linux UNIX *dvd_root*`/install`
     - Windows *dvd_root*`\install.bat`
   - Execute o seguinte script para iniciar o assistente para a instalação do WebSphere eXtreme Scale Client:
     - Linux UNIX `root/`*`WXS_Client`*`/install`
     - Windows `root\`*`WXS_Client`*`\install.bat`
2. Siga os prompts no assistente.

   O painel de recursos opcionais lista os recursos a partir dos quais é possível escolher instalar. Porém, os recursos não poderão ser incluídos incrementalmente no ambiente do produto depois que o produto for instalado. Se você escolher não instalar um recurso com a instalação inicial do produto, deverá desinstalar e reinstalar o produto para incluir o recurso.

   O painel Aumento de Perfil lista os perfis existentes que podem ser selecionados para alterar com os recursos do eXtreme Scale. Porém, se você selecionar perfis existentes que já estão em uso, um painel de aviso será exibido. Para continuar com a instalação, pare os servidores que estão configurados nos perfis ou clique em **Voltar** para remover os perfis da sua seleção.

Resultados

Windows Se estiver instalando o WebSphere eXtreme Scale Client no Windows, será possível ver o seguinte texto nos resultados da instalação:

```
Sucesso: A instalação do seguinte produto foi bem-sucedida:
WebSphere eXtreme Scale Client.  Algumas etapas de configuração têm erros.
Para obter mais informações, consulte o seguinte arquivo de log:
<WebSphere Application Server install root>\logs\wxs_client\install\log.txt"
Revise o log de instalação (log.txt) e o log de aumento do gerenciador de implementação
.
```

Se vir uma falha com o arquivo `iscdeploy.sh`, é possível ignorar o erro. Este erro não causa nenhum problema.

O que Fazer Depois

Se você estiver executando o WebSphere Application Server Versão 6.1 ou Versão 7.0, é possível usar o plug-in Profile Management Tool ou o comando manageprofiles. Se você estiver executando o WebSphere Application Server Versão 6.0.2, é necessário usar o comando wasprofile para criar e alterar os perfis.

Implemente o aplicativo, inicie o serviço de catálogo e inicie os contêineres no seu ambiente do WebSphere Application Server. Consulte “Administrando o WebSphere eXtreme Scale com o WebSphere Application Server” na página 342 para obter mais informações.

Uso do Plug-in Installation Factory para Criação e Instalação de Pacotes Customizados

Utilize o plug-in do IBM Installation Factory para WebSphere eXtreme Scale para criar um pacote de instalação customizado (CIP) ou um pacote de instalação integrado (IIP). Um CIP contém um único pacote de instalação do produto e vários recursos opcionais. Um IIP combina um ou mais pacotes de instalação em um único fluxo de trabalho de instalação projetado.

Antes de Iniciar

Antes de poder criar e instalar pacotes customizados para o eXtreme Scale, primeiro é necessário fazer o download dos seguintes produtos:

- IBM Installation Factory para WebSphere Application Server
- Plug-in doIBM Installation Factory lug-in para o WebSphere eXtreme Scale

Sobre Esta Tarefa

Utilizando o Installation Factory, é possível criar um CIP ao combinar um único componente de produto com pacotes de manutenção, scripts de customização e outros arquivos. Ao criar um IIP, você agrega componentes individuais ou pacotes de instalação em um único pacote de instalação.

Arquivo de Definição da Construção

Um arquivo de definição de construção é um documento XML que especifica como construir e instalar um pacote de instalação customizado (CIP) ou um pacote de instalação integrado (IIP). O IBM Installation Factory para WebSphere eXtreme Scale lê os detalhes do pacote do arquivo de definição de construção para gerar um CIP ou um IIP.

Para poder criar um CIP ou um IIP, é necessário criar um arquivo de definição de construção para cada pacote customizado. O arquivo de definição de construção descreve quais componentes do produto ou pacotes de instalação devem ser instalados, o local do CIP ou do IIP, os pacotes de manutenção a serem incluídos, os scripts de instalação e outros arquivos escolhidos. Também é possível especificar no arquivo de definição de construção do IIP a ordem na qual o Installation Factory instalará cada pacote de instalação.

O assistente de Definição de Construção demonstra o processo de criação de um arquivo de definição de construção. Também é possível utilizar o assistente para modificar um arquivo de definição de construção existente. Cada painel do assistente de Definição de Construção solicita informações sobre um pacote customizado, como a identificação do pacote, o local de instalação da definição de construção e o local de instalação do pacote customizado. Todas essas informações são salvas no novo arquivo de definição de construção, ou modificadas e salvas em um arquivo de definição de construção existente. Para obter informações adicionais, consulte os painéis do Assistente de Definição de Construção do CIP e os painéis do Assistente de Definição de Construção do IIP.

Para criar apenas o arquivo de definição de construção, é possível utilizar a ferramenta da interface da linha de comandos para gerar o pacote customizado fora da GUI. Consulte “Instalação Silenciosa de um CIP ou IIP” na página 33 para obter informações adicionais.

Criando um Arquivo de Definição de Construção e Gerando um CIP

O plug-in do IBM Installation Factory para WebSphere eXtreme Scale gera um pacote de instalação customizado (CIP) de acordo com os detalhes especificados no arquivo de definição de construção. A definição de construção especifica o pacote do produto a ser instalado, o local do CIP, os pacotes de manutenção a serem incluídos na instalação, os arquivos de script de instalação e quaisquer arquivos adicionais a serem incluídos no CIP.

Sobre Esta Tarefa

É possível usar o assistente de definição de Construção para criar um arquivo de definições de construção e gerar um CIP.

Procedimento

1. Execute o seguinte script a partir do diretório *IF_HOME*/bin para iniciar o Installation Factory:
   - UNIX Linux `ifgui.sh`
   - Windows `ifgui.bat`

   Clique no ícone **Nova Definição de Construção**.
2. Selecione o produto a ser incluído no arquivo de definições de construção e clique em **Concluir** para iniciar o assistente de definição de Construção.
3. Siga os prompts no assistente.

   No painel Scripts de Instalação e Desinstalação, clique em **Incluir Scripts...** para preencher a tabela com todos os scripts de instalação customizados. Digite o local dos arquivos de script e limpe a caixa de opção para continuar se uma mensagem de erro for exibida. A operação é interrompida por padrão. Clique em **OK** para retornar ao painel.

Resultados

Você criou e customizou o arquivo de definição de construção e gerou o CIP se tiver optado por trabalhar no modo conectado.

Se o assistente de definição de Construção não fornecer a opção de gerar o CIP a partir do arquivo de definições de construção, você ainda pode gerá-lo por meio da execução do script `ifcli.sh|bat` no diretório *IF_HOME*`/bin`.

O que Fazer Depois

Instale o CIP. Consulte "Instalando um CIP" para obter mais informações.

Instalando um CIP:

Simplifique o processo de instalação do produto instalando um pacote de instalação customizado (CIP). Um CIP é uma imagem de instalação de produto único que pode incluir um ou mais pacotes de manutenção, scripts de configuração e outros arquivos.

Antes de Iniciar

Para poder instalar um CIP, é necessário criar um arquivo de definição de construção para especificar quais opções serão incluídas no CIP. Consulte o "Criando um Arquivo de Definição de Construção e Gerando um CIP" na página 27 para obter informações adicionais.

Sobre Esta Tarefa

Um CIP combina e instala um único componente do produto com pacotes de manutenção, scripts de customização e outros arquivos.

Procedimento

1. Pare todos os processos que estão sendo executados na estação de trabalho que você está preparando para instalação. Para parar o gerenciador de implementação, execute o seguinte script:
   - Linux UNIX *profile_root*`/bin/stopManager.sh`
   - Windows *profile_root*`\bin\stopManager.bat`

   Para parar os nós, execute o seguinte script:
   - Linux UNIX *profile_root*`/bin/stopNode.sh`
   - Windows *profile_root*`\bin\stopNode.bat`
2. Execute o seguinte script para iniciar a instalação:
   - Linux UNIX *CIP_home*`/bin/install`
   - Windows *CIP_home*`\bin\install.bat`
3. Siga os prompts no assistente para concluir a instalação.

   O painel de recursos opcionais lista os recursos a partir dos quais é possível escolher instalar. Porém, os recursos não poderão ser incluídos incrementalmente no ambiente do produto depois que o produto for instalado. Se você escolher não instalar um recurso com a instalação inicial do produto, deverá desinstalar e reinstalar o produto para incluir o recurso.

O painel Aumento de Perfil lista os perfis existentes que podem ser selecionados para alterar com os recursos do eXtreme Scale. Porém, se você selecionar perfis existentes que já estão em uso, um painel de aviso será exibido. Para continuar com a instalação, pare os servidores que estão configurados nos perfis ou clique em **Voltar** para remover os perfis da sua seleção.

Resultados

Você instalou com êxito o CIP.

O que Fazer Depois

Se você estiver executando o WebSphere Application Server Versão 6.1 ou Versão 7.0, é possível usar o plug-in da Profile Management Tool ou o comando manageprofiles para criar e alterar perfis. Se você estiver executando o WebSphere Application Server Versão 6.0.2, é necessário usar o comando wasprofile para criar e aumentar os perfis. Consulte “Criando e Alterando Perfis para o WebSphere eXtreme Scale” na página 42 para obter informações adicionais.

Se você alterar perfis do eXtreme Scale durante o processo de instalação, poderá implementar os aplicativos, iniciar um serviço de catálogo e iniciar os contêineres no seu ambiente do WebSphere Application Server. Consulte “Administrando o WebSphere eXtreme Scale com o WebSphere Application Server” na página 342 para obter mais informações.

Instalando um CIP para Aplicar Manutenção a uma Instalação do Produto Existente:

É possível aplicar pacotes de manutenção a uma instalação existente do produto instalando um CIP (Pacote de Instalação Customizada). O processo de aplicar a manutenção a uma instalação existente com uma CIP é normalmente referida como *instalação slip*.

Antes de Iniciar

Crie um arquivo de definições de construção para especificar quais opções incluir no CIP. Consulte o “Criando um Arquivo de Definição de Construção e Gerando um CIP” na página 27 para obter informações adicionais.

Sobre Esta Tarefa

Ao aplicar a manutenção com um CIP que contém um pacote de atualizações, um fix pack ou ambos, todos os relatórios de análise do programa autorizado instalados anteriormente (APAR) são desinstalados pelo assistente. Se o CIP estiver no mesmo nível do produto, as APARs anteriormente instaladas permanecem somente se elas forem empacotadas no CIP. Para aplicar com sucesso a manutenção em uma instalação existente, você deve incluir os recursos instalados no CIP.

Procedimento

1. Pare todos os processos que estão sendo executados na estação de trabalho que você está preparando para instalação. Para parar o gerenciador de implementação, execute o seguinte comando:
   - Linux UNIX *profile_root*/bin/stopManager.sh

- Windows *profile_root*\bin\stopManager.bat

Para parar os nós, execute o seguinte script:

- Linux UNIX *profile_root*\bin\stopNode.sh
- Windows *profile_root*\bin\stopNode.bat

2. Execute o seguinte script para iniciar a instalação:
   - Linux UNIX *CIP_home*/bin/install
   - Windows *CIP_home*\bin\install.bat
3. Siga os prompts no assistente para concluir a instalação.

   O resumo da visualização da instalação relaciona a versão do produto resultante e todos os recursos aplicáveis e correções temporárias. Em seguida, o assistente aplica com êxito a manutenção e atualiza os recursos do produto.

Resultados

Os arquivos binários do produto são copiados para o diretório *was_home*/properties/version/nif/backup. É possível usar o IBM Update Installer para desinstalar a atualização e restaurar sua estação de trabalho. Consulte o "Desinstalando Atualizações de CIP de uma Instalação Existente do Produto" para obter informações adicionais.

Desinstalando Atualizações de CIP de uma Instalação Existente do Produto:

É possível remover atualizações de CIP de uma instalação de produto existente sem remover o produto todo. Use o IBM Update Installer Versão 7.0.0.4 para desinstalar quaisquer atualizações do CIP. Esta tarefa também é chamada de uma *desinstalação slip*.

Antes de Iniciar

Você precisa ter pelo menos uma cópia existente do produto instalada no sistema.

Procedimento

1. Faça Download da Versão 7.0.0.4 do Update Installer a partir do seguinte site FTP:

   ftp://ftp.software.ibm.com/software/websphere/cw/process_server/FEP/UPDI/7004
2. Instale o Update Installer. Consulte Instalando o Update Installer para WebSphere Software no Centro de Informações do WebSphere Application Server para obter informações adicionais.
3. Desinstale todos os fix packs, pacotes de atualizações ou correções temporárias que você incluiu em seu ambiente após a instalação do CIP.
4. Desinstale todas as correções temporárias que tiver incluído na instalação slip. Este processo é igual a desinstalar um fix pack simples ou pacote de atualizações. Porém, a manutenção que foi incluída no CIP agora está incluída em uma única operação.
5. Desinstale o CIP utilizando o Update Installer. Os níveis de manutenção retornam ao estado pré-atualização e o CIP é denotado pelo identificador de CIP que é incluído como prefixo no nome do arquivo. O exemplo a seguir mostra como um CIP pode ser exibido de forma diferente de outros pacotes de manutenção regulares no painel de seleção do pacote de manutenção:

   CIP

```
com.ibm.ws.cip.7000.wxs.primary.ext.pak
```

Resultados

Você removeu com êxito as atualizações de CIP de uma instalação existente do produto.

Criando um Arquivo de Definição de Construção e Gerando um IIP

O plug-in IBM Installation Factory para WebSphere eXtreme Scale gera um IIP baseado nas propriedades fornecidas pelo arquivo de definições de construção. O arquivo de definições de construção contém informações como quais pacotes de instalação incluir no IIP, a ordem em que o Installation Factory instala cada pacote e o local do IIP.

Sobre Esta Tarefa

É possível usar o assistente de definição de Construção para criar um arquivo de definições de construção e gerar um IIP.

Procedimento

1. Execute o seguinte script a partir do diretório *IF_HOME*/bin para iniciar o Installation Factory:
   - UNIX Linux `ifgui.sh`
   - Windows `ifgui.bat`
2. Clique no ícone **Criar Novo Pacote de Instalação Integrado** para iniciar o assistente de Definição de Construção.
3. Siga os prompts no assistente.
   a. No painel Construir o IIP, selecione um pacote de instalação suportado na lista, e clique em **Incluir Instalador** para incluir o pacote de instalação no IIP. Um painel com o nome do pacote, identificador do pacote e as propriedades do pacote é exibido. Para visualizar informações específicas sobre o pacote selecionado, clique em **Visualizar Informações do Pacote de Instalação**. Clique em **Modificar** para digitar o caminho de diretório para o pacote de instalação de cada sistema operacional. Se estiver incluindo um pacote de instalação para WebSphere Extended Deployment, selecione a caixa de opção, que fornece a opção para usar o mesmo pacote para todos os sistemas operacionais suportados. Clique em **OK** e retorne ao painel Construir o IIP. Uma chamada é criada por padrão.
      - Para modificar o caminho do diretório para um pacote de instalação, selecione o pacote a partir dos Pacotes de Instalação usados na lista do IIP e clique em **Modificar**.
      - Para modificar uma chamada, selecione a chamada e clique em **Modificar**. Especifique o local de instalação padrão da chamada em cada sistema operacional. Especifique o local para o arquivo de resposta se selecionar uma instalação silenciosa como o modo de instalação padrão.
      - Clique em **Incluir Chamada** para incluir uma contribuição de chamada no pacote de instalação. É exibido um painel a partir do qual é possível especificar propriedades para a chamada.
      - Clique em **Remover** para remover os pacotes de instalação ou as chamadas.
4. Revise o resumo das seleções, selecione a opção **Salvar arquivo de definição de construção e gerar pacote de instalação integrado** e clique em **Concluir**.

Alternativamente, é possível salvar o arquivo de definições de construção sem gerar o IIP. Com essa opção, você realmente gera o IIP fora do assistente executando o script `ifcli.bat | ifcli.sh` a partir do diretório *IF_home*/bin/.

Resultados

Você criou e customizou o arquivo de definições de construção para um IIP.

O que Fazer Depois

Instale o IIP.

Instalando um IIP:

Utilize o plug-in do IBM Installation Factory para WebSphere eXtreme Scale para instalar um pacote de instalação integrado (IIP). Um IIP combina um ou mais pacotes de instalação em um único fluxo de trabalho que você projeta.

Antes de Iniciar

Para poder instalar um CIP, é necessário criar um arquivo de definição de construção para especificar quais opções serão incluídas no CIP. Consulte o "Criando um Arquivo de Definição de Construção e Gerando um IIP" na página 31 para obter informações adicionais.

Sobre Esta Tarefa

Um IIP pode incluir um ou mais pacotes de instalação geralmente disponíveis, um ou mais CIPs e outros arquivos e diretórios opcionais. Ao instalar o IIP, você agrega vários pacotes de instalação, ou *contribuições*, em um único pacote, e depois instala as contribuições em uma ordem específica para concluir a instalação de ponta a ponta.

Procedimento

1. Execute o seguinte script para iniciar o assistente:
   - Linux UNIX *IIP_home*/bin/install
   - Windows *IIP_home*\bin\install.bat
2. Clique em **Sobre** no painel Boas-vindas para visualizar os detalhes do IIP, como o identificador do pacote, os sistemas operacionais suportados e os pacotes de instalação incluídos.

   Optional: Para modificar as opções de instalação de cada pacote, clique em **Modificar**.

   Optional: Dois botões **Visualizar Log** são exibidos no painel do assistente. Para visualizar o log de cada pacote, clique no botão **Visualizar Log** exibido ao lado da tabela que lista os pacotes de instalação. Para visualizar os detalhes gerais do log do IIP, clique no botão **Visualizar Log** exibido ao lado das informações de status.
3. Selecione os pacotes de instalação a serem executados e clique em **Instalar**. É exibida uma lista de todas as contribuições na ordem de chamada que o IIP contém. Para designar que chamadas de contribuição não devem ser executadas durante a instalação, desmarque a caixa de opção localizada próxima ao campo **Nome da instalação**.

Resultados

Você instalou com êxito um IIP.

Modificando um Arquivo de Definição de Construção Existente de um IIP:

É possível editar ou incluir nas propriedades de um IIP para customizar ainda mais a instalação.

Sobre Esta Tarefa

Para alterar as propriedades de um IIP, modifique o arquivo de definição de construção existente.

Procedimento

1. Execute o seguinte script a partir do diretório *IF_HOME*/bin para iniciar o Installation Factory:
   - UNIX Linux `ifgui.sh`
   - Windows `ifgui.bat`
2. Clique no ícone **Abrir Definição de Construção** e selecione o arquivo de definições de construção que deseja modificar.
3. Selecione as propriedades específicas do IIP que deseja modificar. A lista a seguir contém as possíveis modificações que podem ser feitas:
   - Alterar a seleção de modo atual. No modo conectado, crie a definição de construção que utilizará e, opcionalmente, gere o IIP, a partir da estação de trabalho atual. No modo desconectado, crie o arquivo de definição de construção a ser utilizado em outra estação de trabalho.
   - Incluir ou remover os sistemas operacionais existentes suportados pelo IIP.
   - Editar o identificador e a versão existentes do IIP.
   - Editar o local de destino do arquivo de definição de construção.
   - Editar o local de destino do IIP.
   - Mudar se um assistente de instalação deve ser exibido para o IIP. O assistente fornece informações sobre o IIP e as opções de instalação quando o IIP é executado.
   - Incluir, remover e editar os pacotes de instalação contidos no IIP.

     Importante: Se você tiver incluído um sistema operacional suportado e não tiver atualizado as propriedades do pacote de instalação no IIP, será exibida uma mensagem de aviso informando que as contribuições selecionadas não contêm os pacotes de instalação identificados de todos os sistemas operacionais suportados pelo IIP. Clique em **Sim** para continuar ou em **Não** para editar o pacote de instalação.
4. Revise o resumo das seleções, selecione **Salvar arquivo de definição de construção e gerar pacote de instalação integrado** e clique em **Concluir**.

Instalação Silenciosa de um CIP ou IIP

É possível instalar silenciosamente um pacote de instalação customizado (CIP) ou um pacote de instalação integrado (IIP) para o produto usando um arquivo de resposta completo, que é configurado especificamente para atender suas necessidades, ou parâmetros passados para a linha de comandos.

Antes de Iniciar

Crie o arquivo de definições de construção para o CIP ou IIP. Consulte o “Criando um Arquivo de Definição de Construção e Gerando um CIP” na página 27 para obter informações adicionais.

Sobre Esta Tarefa

Uma instalação silenciosa utiliza o mesmo programa que a versão de interface gráfica com o usuário (GUI) utiliza. Entretanto, no lugar de exibir uma interface de assistente, a instalação silenciosa lê todas as suas respostas de um arquivo que você customiza ou dos parâmetros que você transmite para a linha de comandos. Se estiver instalando silenciosamente um IIP, é possível chamar uma contribuição com uma combinação de opções especificadas diretamente na linha de comandos, bem como opções que você especifica em um arquivo de resposta. No entanto, as opções de contribuição transmitidas à linha de comandos farão com que o instalador do IIP ignore todas as opções especificadas em um arquivo de resposta específico da contribuição. Consulte as opções de instalação do IIP em detalhes para obter informações adicionais.

Nota: É necessário especificar o nome completo do arquivo de resposta. Especificar o caminho relativo faz com que a instalação falhe sem nenhuma indicação de que ocorreu um erro.

Procedimento

1. Opcional: Se você optar por instalar o CIP ou IIP usando um arquivo de resposta, primeiro customize o arquivo.
   a. Copie o arquivo de resposta `wxssetup.response.txt` do DVD do produto para a unidade de disco.
   b. Abra e edite o arquivo de resposta no editor de texto de sua escolha. O arquivo inclui comentários para ajudá-lo no processo de configuração e deve incluir estes parâmetros:
      - O contrato de licença
      - O local da instalação do produto

      Dica: O instalador usa o local que você seleciona para sua instalação para determinar onde sua instância do WebSphere Application Server está instalada. Se você instalar em um nó com múltiplas instâncias do WebSphere Application Server, defina claramente seu local.
   c. Execute o script a seguir para iniciar seu arquivo de resposta customizado.
      - Linux UNIX `install -options /absolute_path/response_file.txt -silent`
      - Windows `install.bat -options C:\drive_path\response_file.txt -silent`
2. Opcional: Se você optar por instalar o CIP ou IIP passando determinados parâmetros para a linha de comandos, execute o script a seguir para iniciar a instalação:
   - Linux UNIX `install -silent -OPT silentInstallLicenseAcceptance=true -OPT installLocation=install_location`
   - Windows `install.bat -silent -OPT silentInstallLicenseAcceptance=true -OPT installLocation=install_location`

em que *install_location* é o local de sua instalação do WebSphere Application Server existente.

3. Revise os registros de erros resultantes ou uma falha na instalação.

Resultados

Você instalou silenciosamente o CIP ou IIP.

O que Fazer Depois

Se você estiver executando o WebSphere Application Server Versão 6.1 ou Versão 7.0, é possível usar o plug-in ferramenta Gerenciamento de Perfil ou o comando manageprofiles para criar e alterar perfis. Se você estiver executando o WebSphere Application Server Versão 6.0.2, é necessário usar o comando wasprofile para criar e alterar os perfis.

Se você alterar perfis do eXtreme Scale durante o processo de instalação, poderá implementar os aplicativos, iniciar um serviço de catálogo e iniciar os contêineres no seu ambiente do WebSphere Application Server. Consulte "Administrando o WebSphere eXtreme Scale com o WebSphere Application Server" na página 342 para obter mais informações.

Arquivo `wxssetup.response.txt`:

É possível usar um arquivo completo de resposta para instalar o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client silenciosamente.

CUIDADO:
Não inclua barras à direita, como / ou \, no final dos caminhos do local de instalação. Esses caminhos são especificados com o atributo installLocation. A inclusão de uma barra no final do local de instalação pode fazer com que a instalação falhe. Por exemplo, o caminho a seguir pode fazer com que a instalação falhe:

```
-OPT installLocation="/usr/IBM/WebSphere/eXtremeScale/"
```

O caminho deve ser especificado como:

```
-OPT installLocation="/usr/IBM/WebSphere/eXtremeScale"
```

Arquivo de resposta para instalação completa do WebSphere eXtreme Scale

```
############################################################################
#
# Arquivo de Opções InstallShield do IBM WebSphere eXtreme Scale
V7.1.0
#
# Nome do assistente: Install
# Origem do assistente: setup.jar
#
# Esse arquivo pode ser utilizado para configurar a Instalação com as opções
especificadas abaixo
# quando o assistente for executado com a opção da linha de comandos
 "-options". Leia a
# documentação de cada configuração para obter informações sobre como alterar esse valor.
# Delimite todos os valores com um par simples de aspas duplas.
#
# Um uso comum de um arquivo de opções é executar o assistente no modo silencioso.
Isso permite
# que o autor do arquivo de opções especifique as configurações do assistente
 sem ter que executar o
```

```
# assistente no modo gráfico ou de console. Para utilizar esse
 arquivo de opções na execução de modo
# silencioso, utilize os seguintes argumentos de linha
de comandos ao executar o assistente:
#
#    -options "D:\installImage\WXS\wxssetup.response" -silent
#
# Observe que o nome do arquivo de resposta qualificado deve ser
utilizado.
#
##############################################################################

##############################################################################
#
# Aceitação de Licença
#
# Valores Válidos:
# true -  Aceita a licença. Instalará o produto.
# false - Recusa a licença. Não ocorrerá a instalação.
#
# Se não ocorrer nenhuma instalação, ela será registrada em um arquivo de log temporário no
# diretório temporário do usuário.
#
# Ao alterar a propriedade silentInstallLicenseAcceptance neste arquivo de resposta para
# "true", você concorda que revisou e concordou com os termos do
# Contrato de Licença do Programa Internacional IBM que acompanha este programa, que
# está localizado em
CD_ROOT\XD\wxs.primary.pak\repository\legal.xs\license.xs.  Se
# você não concordar com esses termos, não altere o valor ou, de
alguma outra forma,
# faça o download, instale, copie, acesse ou utilize o programa e
devolva imediatamente
# o programa e a prova de titularidade à parte de quem você o
adquiriu para
# obter um reembolso do valor pago.
#
-OPT silentInstallLicenseAcceptance="false"


##############################################################################
# Verificação de Pré-requisito Não de Bloqueio
#
# Se desejar desativar a verificação de pré-requisito não de bloqueio, remova o comentário
# da seguinte linha. Isso notificará o instalador para continuar
# a instalação e registrará os avisos mesmo que a verificação de pré-requisito
# tenha falhado.
#
#-OPT disableNonBlockingPrereqChecking="true"


##############################################################################
#
# Local da Instalação
#
# O local de instalação do produto. Especifique um diretório válido no qual o
# produto deve ser instalado. Se o diretório contiver espaços, coloque-o entre
# aspas duplas conforme mostrado no exemplo do Windows abaixo. Observe que espaços no
# local de instalação são suportados apenas nos sistemas operacionais Windows. # O comprimento máxim
#
# Abaixo está uma lista dos locais de instalação padrão para cada sistema operacional
# suportado quando você estiver instalando como um usuário root.  Por padrão, neste arquivo de respos
# é utilizado o local de instalação do Windows.  Se você desejar utilizar o local da instalação
# padrão para outro sistema operacional, remova o comentário da entrada do local da
# instalação padrão apropriada (removendo '#') e comente a linha
# (incluindo '#') na entrada do sistema operacional Windows abaixo.
#
# O local da instalação é utilizado para determinar se o WebSphere eXtreme Scale deve
```

```
# ser instalado como uma implementação independente ou se deve ser integrado a uma
# instalação existente do WebSphere Application Server.
#
# Se o local especificado for uma instalação existente do WebSphere Application Server
# ou WebSphere Network Deployment, o eXtreme Scale será integrado ao
# WebSphere Application Server existente.  Se o local especificado for
# um diretório novo ou vazio, o WebSphere eXtreme Scale será instalado como uma
# implementação independente.
#
# Nota: Se o local da instalação especificado contiver uma instalação anterior do
# WebSphere eXtreme Scale, WebSphere eXtended Deployment DataGrid ou
# ObjectGrid, a instalação falhará.
#
# Local de Instalação Padrão do AIX:
#
#  -OPT installLocation="/usr/IBM/WebSphere/eXtremeScale"
#
# Local de Instalação Padrão do HP-UX, Solaris ou Linux:
#
#  -OPT installLocation="/opt/IBM/WebSphere/eXtremeScale"
#
#
# Local de Instalação Padrão do Windows:
#
-OPT installLocation="C:\Program Files\IBM\WebSphere\eXtremeScale"

#
# Se você estiver instalando como um usuário não raiz no Unix ou um não administrador no
# Windows, os seguintes locais de instalação padrão serão sugeridos. Certifique-se de que
# você tenha permissões de gravação para o local da instalação escolhido.
#
# Local de Instalação Padrão do AIX:
#
#  -OPT installLocation="<user's home>/IBM/WebSphere/eXtremeScale"
#
# Local de Instalação Padrão do HP-UX, Solaris ou Linux:
#
#  -OPT installLocation="<user's home>/IBM/WebSphere/eXtremeScale"
#
# Local de Instalação Padrão do Windows:
#
# -OPT installLocation="C:\IBM\WebSphere\eXtremeScale"


###############################################################################
#  Instalação de Recursos Opcionais
#
# Especifique quais dos recursos opcionais deseja instalar configurando cada
# recurso desejado como "true".  Configure quaisquer recursos opcionais que não queira
# instalar como "false".
#
# As opções selectServer, selectClient, selectPF e selectXSStreamQuery são
# válidas apenas quando a opção installLocation acima contém uma instalação do
# WebSphere Application Server.  As opções são ignoradas em uma instalação independente
# do WebSphere eXtreme Scale.
#
# Na instalação independente do WebSphere eXtreme Scale, o cliente e o servidor
# do eXtreme Scale são instalados automaticamente.
As opções de recurso para a
# instalação independente do eXtreme Scale são selectXSConsoleOther e
# selectXSStreamQueryOther.

#
# Esta opção, quando selecionada, instala os componentes que são necessários para a execução
# de servidores WebSphere eXtreme Scale e do provedor de serviço de cache dinâmico eXtreme
# Scale.
Se essa opção estiver selecionada, o WebSphere eXtreme Scale Client
```

```
# deve também ser selecionado ao não ser comentado e configurado para um valor de "true".
# Caso contrário, a instalação silenciosa FALHARÁ.
#
-OPT selectServer="true"

#
# Esta opção, quando selecionada, instala os componentes que são
 necessários para a execução
# de aplicativos cliente WebSphere eXtreme Scale. Se a
opção do servidor for selecionada
# acima, essa opção também deverá ser selecionada por meio
 da remoção do comentário e configurada para
# um valor "true" ou a instalação silenciosa FALHARÁ.
#
-OPT selectClient="true"

#
# Esta opção, quando selecionada, instala os componentes que são
necessários para a execução
# do WebSphere eXtreme Scale Console. Se esta opção for selecionada,
 o local de instalação
# especificado acima deve ser um diretório novo
 ou vazio porque a opção do console
# é válida apenas para a implementação
independente do WebSphere eXtreme Scale. Para
# instalar esta opção, a linha de
opção a seguir não deve ser comentada e deve ser configurada
# para um valor de "true".
#-OPT selectXSConsoleOther="false"

#
# As opções a seguir, se selecionadas, instalarão uma funcionalidade REPROVADA.
#
# Esta opção seleciona o WebSphere Partition Facility para instalação.
# Esta funcionalidade está REPROVADA.  Para a instalação dessa opção, a seguinte
# linha de opção deverá ter seu comentário removido e ser configurada para um valor "true".
#
#-OPT selectPF="false"

#
# Esta opção seleciona o WebSphere eXtreme Scale StreamQuery for WAS para
# instalação.  Esta funcionalidade está REPROVADA.  Para a instalação dessa opção,
# a linha de opção a seguir deverá ter seu comentário removido e ser
 configurada para um valor "true".
# Se essa opção estiver selecionada, o WebSphere eXtreme Scale Client
# deve também ser selecionado ao não ser comentado e configurado
 para um valor de "true".
# Caso contrário, a instalação silenciosa FALHARÁ.
#
#-OPT selectXSStreamQuery="false"

#
# Esta opção seleciona o WebSphere eXtreme Scale StreamQuery for J2SE para
# instalação.  Esta funcionalidade está REPROVADA.  Para a instalação dessa opção,
# a linha de opção a seguir deverá ter seu comentário
 removido e ser configurada para um valor "true".
# Se essa opção estiver selecionada, o WebSphere eXtreme Scale Client
# deve também ser selecionado ao não ser comentado
 e configurado para um valor de "true".
# Caso contrário, a instalação silenciosa FALHARÁ.
#
#-OPT selectXSStreamQueryOther="false"


###############################################################################
#  Lista de Perfis para Aumento
#
```

```
# Especifique qual dos perfis existentes deseja aumentar ou remover o comentário da
# linha para aumentar cada perfil existente detectado pela instalação.
#
# Para especificar vários perfis, utilize uma vírgula para separar
 diferentes nomes de perfis.
# Por exemplo, "AppSrv01,Dmgr01,Custom01". A lista não deve conter espaços.
#
-OPT profileAugmentList=""


##############################################################################
# Controle de Rastreio
#
# O formato de saída de rastreio pode ser controlado por meio da opção
# -OPT traceFormat=ALL
#
# As opções para o formato são 'text' e 'XML'. Por padrão, ambos os formatos
# serão produzidos em dois arquivos de rastreio diferentes.
#
# Se apenas um formato for necessário, utilize a opção traceFormat para especificar
# um, como a seguir:
#
# Valores Válidos:
#
# text  -  As linhas no arquivo de rastreio estarão em um formato de
 texto simples para facilitar a
#          capacidade de leitura.
# XML -    As linhas no arquivo de rastreio estarão no formato XML
de criação de log Java padrão,
#          que pode ser visualizado com o uso de qualquer editor de
texto ou XML ou com o uso da
#
ferramenta Chainsaw da Apache na seguinte URL:
#        (http://logging.apache.org/log4j/docs/chainsaw.html).
#
# A quantidade de informações de rastreio capturadas pode ser controlada
 utilizando a opção:
# -OPT traceLevel=INFO
#
# Valores Válidos:
#
# Rastreio   Numérico
# Nível      Nível    Descrição
# -------  ---------  ------------------------------------------------------
# OFF     0  Nenhum arquivo de rastreio é produzido
# SEVERE     1  Apenas erros graves são gerados para o
 arquivo de rastreio
# WARNING    2       Mensagens referentes a exceções não fatais e avisos são
#                    incluídas no arquivo de rastreio
#   INFO       3            As mensagens informativas são incluídas no
arquivo de rastreio
#                           (esse é o nível de rastreio padrão)
# CONFIG     4  As mensagens relacionadas à configuração
são incluídas no arquivo de rastreio
# FINE     5  Rastrear chamadas de método para métodos públicos
#   FINER      6            Rastrear chamadas de método para métodos não-públicos,
exceto
#                           getters e setters
#   FINEST     7            Rastrear todas as chamadas de método, a entrada/saída
de rastreio incluirão
#                           parâmetros e valor de retorno
```

Arquivo de resposta para instalação do WebSphere eXtreme Scale Client

```
##############################################################################
#
# Arquivo de Opções InstallShield do IBM WebSphere eXtreme Scale Client V7.1.0
```

```
#
# Nome do assistente: Install
# Origem do assistente: setup.jar
#
# Esse arquivo pode ser utilizado para configurar a Instalação com as opções
 especificadas abaixo
# quando o assistente for executado com a opção da linha de comandos "-options". Leia a
# documentação de cada configuração para obter informações sobre como alterar esse valor.
# Delimite todos os valores com um par simples de aspas duplas.
#
# Um uso comum de um arquivo de opções é executar o assistente no modo silencioso.
 Isso permite
# que o autor do arquivo de opções especifique as configurações do assistente
 sem ter que executar o
# assistente no modo gráfico ou de console. Para utilizar esse arquivo
 de opções na execução de modo
# silencioso, utilize os seguintes argumentos de linha de comando
s ao executar o assistente:
#
#    -options "D:\installImage\WXS_Client\wxssetup.response" -silent
#
# Observe que o nome do arquivo de resposta qualificado deve ser
utilizado.
#
###############################################################################

###############################################################################
#
# Aceitação de Licença
#
# Valores Válidos:
# true -  Aceita a licença. Instalará o produto.
# false - Recusa a licença. Não ocorrerá a instalação.
#
# Se não ocorrer nenhuma instalação, ela será registrada em um arquivo
 de log temporário no
# diretório temporário do usuário.
#
# Ao alterar a propriedade silentInstallLicenseAcceptance neste arquivo de resposta para
# "true", você concorda que revisou e concordou com os termos do
# Contrato de Licença do Programa Internacional IBM que acompanha este programa, que
# está localizado em
# CD_ROOT\WXS_Cleint\wxs.client.primary.pak\repository\legal.xs.client\license.xs.
# Se você não concordar com esses termos, não altere o valor ou, de outra forma,
# faça o download, instale, copie, acesse ou utilize o programa e
devolva imediatamente
# o programa e a prova de titularidade à parte de quem você o
adquiriu para
# obter um reembolso do valor pago.
#
-OPT silentInstallLicenseAcceptance="false"


###############################################################################
# Verificação de Pré-requisito Não de Bloqueio
#
# Se desejar desativar a verificação de pré-requisito não de bloqueio,
remova o comentário
# da seguinte linha. Isso notificará o instalador para continuar
# a instalação e registrará os avisos mesmo que a verificação de pré-requisito
# tenha falhado.
#
#-OPT disableNonBlockingPrereqChecking="true"


###############################################################################
#
```

```
# Local da Instalação
#
# O local de instalação do produto. Especifique um diretório válido no qual o
# produto deve ser instalado. Se o diretório contiver espaços, coloque-o entre
# aspas duplas conforme mostrado no exemplo do Windows abaixo. Observe que espaços no
# local de instalação são suportados apenas nos sistemas operacionais Windows.
# O comprimento máximo do caminho é 60 caracteres para Windows.
#
# Abaixo está uma lista dos locais de instalação padrão para cada sistema operacional
# suportado quando você estiver instalando como um usuário root.
Por padrão, neste arquivo de resposta,
# é utilizado o local de instalação do Windows.
 Se você desejar utilizar o local da instalação
# padrão para outro sistema operacional, remova o comentário da entrada do local da
# instalação padrão apropriada (removendo '#') e comente a linha
# (incluindo '#') na entrada do sistema operacional Windows abaixo.
#
# O local da instalação é utilizado para determinar se o WebSphere eXtreme Scale deve
# ser instalado como uma implementação independente ou se deve ser integrado a uma
# instalação existente do WebSphere Application Server.
#
# Se o local especificado for uma instalação existente do WebSphere Application Server
# ou WebSphere Network Deployment, o eXtreme Scale será integrado ao
# WebSphere Application Server existente.  Se o local especificado for
# um diretório novo ou vazio, o WebSphere eXtreme Scale será instalado como uma
# implementação independente.
#
# Nota: Se o local da instalação especificado contiver uma instalação anterior do
# WebSphere eXtreme Scale, WebSphere eXtended Deployment DataGrid ou
# ObjectGrid, a instalação falhará.
#
# Local de Instalação Padrão do AIX:
#
#  -OPT installLocation="/usr/IBM/WebSphere/eXtremeScale"
#
# Local de Instalação Padrão do HP-UX, Solaris ou Linux:
#
#  -OPT installLocation="/opt/IBM/WebSphere/eXtremeScale"
#
#
# Local de Instalação Padrão do Windows:
#
-OPT installLocation="C:\Program Files\IBM\WebSphere\eXtremeScale"

#
# Se você estiver instalando como um usuário não raiz no Unix ou um
não administrador no
# Windows, os seguintes locais de instalação padrão
serão sugeridos. Certifique-se de que
# você tenha permissões de gravação para o
local da instalação escolhido.
#
# Local de Instalação Padrão do AIX:
#
#  -OPT installLocation="<user's home>/IBM/WebSphere/eXtremeScale"
#
# Local de Instalação Padrão do HP-UX, Solaris ou Linux:
#
#  -OPT installLocation="<user's home>/IBM/WebSphere/eXtremeScale"
#
# Local de Instalação Padrão do Windows:
#
# -OPT installLocation="C:\IBM\WebSphere\eXtremeScale"


###############################################################################
#  Lista de Perfis para Aumento
```

```
#
# Especifique qual dos perfis existentes deseja aumentar ou remover
o comentário da
# linha para aumentar cada perfil existente detectado pela instalação.
#
# Para especificar vários perfis, utilize uma vírgula para separar
 diferentes nomes de perfis.
# Por exemplo, "AppSrv01,Dmgr01,Custom01". A lista não deve conter espaços.
#
-OPT profileAugmentList=""


################################################################################
# Controle de Rastreio
#
# O formato de saída de rastreio pode ser controlado por meio da opção
# -OPT traceFormat=ALL
#
# As opções para o formato são 'text' e 'XML'. Por padrão, ambos os formatos
# serão produzidos em dois arquivos de rastreio diferentes.
#
# Se apenas um formato for necessário, utilize a opção traceFormat para especificar
# um, como a seguir:
#
# Valores Válidos:
#
# text  -  As linhas no arquivo de rastreio estarão em um formato
 de texto simples para facilitar a
#          capacidade de leitura.
# XML -    As linhas no arquivo de rastreio estarão no formato
XML de criação de log Java padrão,
#          que pode ser visualizado com o uso de qualquer editor
de texto ou XML ou com o uso da
#
ferramenta Chainsaw da Apache na seguinte URL:
#        (http://logging.apache.org/log4j/docs/chainsaw.html).
#
# A quantidade de informações de rastreio capturadas pode ser
controlada utilizando a opção:
# -OPT traceLevel=INFO
#
# Valores Válidos:
#
# Rastreio   Numérico
# Nível      Nível    Descrição
# -------  ---------  ------------------------------------------------------
# OFF     0  Nenhum arquivo de rastreio é produzido
# SEVERE     1  Apenas erros graves são gerados
 para o arquivo de rastreio
# WARNING    2       Mensagens referentes a exceções não fatais e avisos são
#                    incluídas no arquivo de rastreio
#   INFO       3            As mensagens informativas são incluídas
no arquivo de rastreio
#                           (esse é o nível de rastreio padrão)
# CONFIG     4  As mensagens relacionadas à configuração são incluídas no arquivo de rastreio
# FINE     5  Rastrear chamadas de método
para métodos públicos
#   FINER      6            Rastrear chamadas de método para métodos não-públicos, exceto
#                           getters e setters
#   FINEST     7            Rastrear todas as chamadas de método,
 a entrada/saída de rastreio incluirão
#                           parâmetros e valor de retorno
```

Criando e Alterando Perfis para o WebSphere eXtreme Scale

Depois de instalar o produto, crie tipos exclusivos de perfis e aumente os perfis existentes para o WebSphere eXtreme Scale.

Antes de Iniciar

Instale o WebSphere eXtreme Scale. Consulte “Integrando o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client com o WebSphere Application Server” na página 22 para obter mais informações.

O aumento de perfis para uso com WebSphere eXtreme Scale é opcional, mas é necessário nos seguintes cenários de uso:

- Para iniciar automaticamente um serviço de catálogo ou contêiner em um processo do WebSphere Application Server. Sem o aumento dos perfis do servidor, os servidores só podem ser iniciados programaticamente com o uso da API ServerFactory ou como processos separados com o uso de scripts startOgServer.
- Para utilizar Performance Monitoring Infrastructure (PMI) para monitorar métricas de WebSphere eXtreme Scale.
- Para exibir a versão do WebSphere eXtreme Scale no console administrativo do WebSphere Application Server.

Sobre Esta Tarefa

Executando com o WebSphere Application Server Versão 6.0.2

Se o seu ambiente contiver o WebSphere Application Server Versão 6.0.2, use o comando wasprofile para criar e alterar os perfis para o WebSphere eXtreme Scale como mostra o seguinte exemplo:

```
install_root/bin/wasprofile.sh|bat -augment -profileName dmgr_01
-templatePath "C:/ProgramFiles/IBM/WebSphere/AppServer/profileTemplates/xs_augment/dmgr"
```

Consulte o comando wasprofile no Centro de Informações do WebSphere Application Server para obter mais informações.

Executando com o WebSphere Application Server Versão 6.1 ou 7.0

Se o seu ambiente contiver o WebSphere Application Server Versão 6.1 ou Versão 7.0, poderá usar o plug-in da Profile Management Tool ou o comando manageprofiles para criar e alterar perfis.

O que Fazer Depois

Dependendo de qual tarefa você escolher concluir, ative o Console de Primeiras Etapas para obter assistência sobre a configuração e o teste do ambiente do produto. O console do First Steps está no diretório `<install_root>\firststeps\wxs\firststeps.bat`. Você também pode criar ou aumentar perfis adicionais repetindo qualquer uma das tarefas anteriores.

Usando a Interface Gráfica com o Usuário para Criar Perfis

Use a interface gráfica com o usuário (GUI), que é fornecida pelo plug-in Profile Management Tool, para criar perfis para o WebSphere eXtreme Scale. Um perfil é um conjunto de arquivos que definem o ambiente de tempo de execução.

Antes de Iniciar

Nota: Se você estiver executando o WebSphere Application Server Versão 6.0.2 ou o WebSphere Application Server Network Deployment Versão 6.0.2, é necessário usar o comando wasprofile para criar ou aumentar um perfil para o WebSphere eXtreme Scale, como mostra o seguinte exemplo:

```
install_root/bin/wasprofile.sh|bat -create -profileName dmgr_01
-templatePath "C:/ProgramFiles/IBM/WebSphere/AppServer/profileTemplates/xs_augment/dmgr"
```

Consulte o comando wasprofile no Centro de Informações do WebSphere Application Server Versão 6.0 para obter mais informações.

Sobre Esta Tarefa

Para utilizar os recursos do produto, o plug-in Profile Management Tool ativa a GUI para ajudar você a configurar perfis, como perfil do WebSphere Application Server, um perfil de gerenciador de implementação, um perfil de célula e um perfil customizado.

Procedimento

Use a GUI do Profile Management Tool para criar perfis. Escolha uma das seguintes opções para iniciar o assistente:

- Selecione **Profile Management Tool** no console do First Steps.
- Acesse o Profile Management Tool a partir do menu **Iniciar**.
- Execute o script `./pmt.sh|bat` a partir do diretório *install_root*`/bin/ProfileManagement`.

O que Fazer Depois

É possível criar perfis adicionais ou alterar perfis existentes. Para reiniciar o Profile Management Tool, execute o comando `./pmt.sh|bat` no diretório *install_root*`/bin/ProfileManagement` ou selecione **Profile Management Tool** no console do First Steps.

Inicie um serviço de catálogo, inicie os contêineres e configure as portas TCP no seu ambiente do WebSphere Application Server. Consulte “Administrando o WebSphere eXtreme Scale com o WebSphere Application Server” na página 342 para obter mais informações.

Usando a Interface Gráfica com o Usuário para Alterar Perfis

Depois de instalar o produto, poderá alterar um perfil existente para torná-lo compatível com o WebSphere eXtreme Scale.

Antes de Iniciar

Nota: Se você estiver executando o WebSphere Application Server Versão 6.0.2 ou o WebSphere Application Server Network Deployment Versão 6.0.2, será necessário usar o comando wasprofile para criar ou alterar um perfil para o WebSphere eXtreme Scale, como mostra o seguinte exemplo:

```
install_root/bin/wasprofile.sh|bat -augment -profileName dmgr_01
-templatePath "C:/ProgramFiles/IBM/WebSphere/AppServer/profileTemplates/xs_augment/dmgr"
```

Consulte o Comando wasprofile no Centro de Informações do WebSphere Application Server para obter mais informações.

Sobre Esta Tarefa

Ao alterar um perfil existente, poderá alterar o perfil ao aplicar um modelo de aumento específico do produto. Por exemplo, os servidores do WebSphere eXtreme Scale não são iniciados automaticamente a menos que o perfil do servidor seja alterado com o modelo xs_augment.

- Altere o perfil com o modelo xs_augment se você instalou o cliente ou o cliente e o servidor do eXtreme Scale.
- Aumente o perfil com o modelo pf_augment apenas se você tiver instalado o recurso de particionamento.
- Aplique ambos os modelos se o seu ambiente contiver o cliente e o recurso de particionamento do eXtreme Scale.

Procedimento

Use a GUI do Profile Management Tool para alterar os perfis para o eXtreme Scale. Escolha uma das seguintes opções para iniciar o assistente:

- Selecione **Profile Management Tool** no console do First Steps.
- Acesse o Profile Management Tool a partir do menu **Iniciar**.
- Execute o script ./pmt.sh|bat no diretório *install_root*/bin/ProfileManagement.

O que Fazer Depois

É possível alterar perfis adicionais. Para reiniciar o Profile Management Tool, execute o comando ./pmt.sh|bat no diretório *install_root*/bin/ProfileManagement ou selecione **Profile Management Tool** no console do First Steps.

Inicie um serviço de catálogo, inicie os contêineres e configure as portas TCP no seu ambiente do WebSphere Application Server. Consulte "Administrando o WebSphere eXtreme Scale com o WebSphere Application Server" na página 342 para obter mais informações.

Comando manageprofiles

É possível usar o utilitário manageprofiles para criar perfis com o modelo WebSphere eXtreme Scale e alterar ou retornar o estado inalterado dos perfis do servidor de aplicativos existentes com os modelos de mudança do eXtreme Scale. Para usar os recursos do produto, seu ambiente deve conter pelo menos um perfil alterado para o produto.

- Antes de poder criar e alterar os perfis, é necessário instalar o eXtreme Scale . Consulte "Integrando o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client com o WebSphere Application Server" na página 22 para obter informações adicionais.
- Se o seu ambiente contiver o WebSphere Application Server Versão 6.0.2, é necessário usar o comando wasprofile para criar e alterar os perfis para o eXtreme Scale como mostra o seguinte exemplo:

  ```
  install_root/bin/wasprofile.sh|bat -augment -profileName dmgr_01
  -templatePath "C:/ProgramFiles/IBM/WebSphere/AppServer/profileTemplates/xs_augment/dmgr"
  ```

  Consulte o comando wasprofile no Centro de Informações do WebSphere Application Server para obter mais informações.

Propósito

O comando manageprofiles cria o ambiente de execução para um processo do produto em um conjunto de arquivos chamado perfil. O perfil define o ambiente de tempo de execução. É possível executar as seguintes ações com o comando manageprofiles:

- Criar e alterar um perfil do gerenciador de implementação
- Criar e alterar um perfil customizado
- Criar e alterar um perfil do servidor de aplicativos independente

- Criar e alterar um perfil de célula
- Retornar o estado inalterado de qualquer tipo de perfil

Ao alterar um perfil existente, poderá mudar o perfil ao aplicar um modelo de mudança específico do produto.

- Altere o perfil com o modelo xs_augment se você instalou o cliente ou o cliente e o servidor do eXtreme Scale.
- Altere o perfil com o modelo pf_augment se você instalou apenas o recurso de particionamento.
- Aplique os dois modelos se o seu ambiente contiver o cliente e o recurso de particionamento do eXtreme Scale.

Local

O arquivo de comando está no diretório *install_root*/bin.

Uso

Para obter ajuda detalhada, use o parâmetro **-help**:

```
./manageprofiles.sh|bat -create -templatePath install_root/profileTemplates/xs_augment/dmgr -help
```

Nas seguintes seções, cada tarefa que pode ser executada usando o comando manageprofiles junto com uma lista de parâmetros necessários é descrita. Para obter detalhes sobre os parâmetros opcionais a serem especificados para cada tarefa, consulte o comando manageprofiles no Centro de Informações do WebSphere Application Server.

Criar um Perfil do Gerenciador de Implementação

É possível usar o comandomanageprofiles para criar um perfil do gerenciador de implementação. O gerenciador de implementação administra os servidores de aplicativos que são federados na célula.

Parâmetros

-create
Cria um perfil. (Requerido)

-templatePath *template_path*
Especifica o caminho de arquivo para o modelo. (Obrigatório)

Utilize o seguinte formato:

```
-templatePath install_root/profileTemplates/template_type/dmgr
```

em que *template_type* é xs_augment ou pf_augment.

Exemplo

- Usando o modelo xs_augment:

```
./manageprofile.sh|bat -create -templatePath
install_root/profileTemplates/xs_augment/dmgr
```

- Usando o modelo pf_augment:

```
./manageprofile.sh|bat -create -templatePath
install_root/profileTemplates/pf_augment/dmgr
```

Criar um perfil customizado

É possível usar o comandomanageprofiles para Criar um perfil customizado. Um perfil customizado é um nó vazio que você customiza por meio do gerenciador de implementação para incluir servidores de aplicativos, clusters ou outros processos Java.

Parâmetros

-create
Cria um perfil. (Requerido)

-templatePath *template_path*
Especifica o caminho de arquivo para o modelo. (Obrigatório)

Utilize o seguinte formato:

```
-templatePath install_root/profileTemplates/template_type/managed
```

em que *template_type* é `xs_augment` ou `pf_augment`.

Exemplo

- Usando o modelo xs_augment:

```
./manageprofile.sh|bat -create -templatePath install_root/profileTemplates/xs_augment/managed
```

- Usando o modelo pf_augment:

```
./manageprofile.sh|bat -create -templatePath install_root/profileTemplates/pf_augment/managed
```

Criar um Perfil de Servidor de Aplicativos Independente

É possível usar o comandomanageprofiles para Criar um perfil do servidor de aplicativos independente.

Parameters

-create
Cria um perfil. (Requerido)

-templatePath *template_path*
Especifica o caminho de arquivo para o modelo. (Obrigatório)

Utilize o seguinte formato:

```
-templatePath install_root/profileTemplates/template_type/default
```

em que *template_type* é `xs_augment` ou `pf_augment`.

Exemplo

- Usando o modelo xs_augment:

```
./manageprofile.sh|bat -create -templatePath install_root/profileTemplates/xs_augment/default
```

- Usando o modelo pf_augment:

```
./manageprofile.sh|bat -create -templatePath install_root/profileTemplates/pf_augment/default
```

Criar o Perfil de uma Célula

É possível usar o comandomanageprofiles para Criar um perfil de célula que consiste de um gerenciador de implementação e de um servidor de aplicativos.

Parameters

Especifique os seguintes parâmetros no modelo do gerenciador de implementação:

-create
Cria um perfil. (Requerido)

-templatePath *template_path*
Especifica o caminho de arquivo para o modelo. (Obrigatório)

Utilize o seguinte formato:

```
-templatePath install_root/profileTemplates/template_type/cell/dmgr
```

em que *template_type* é `xs_augment` ou `pf_augment`.

Especifique os seguintes parâmetros com o modelo do servidor de aplicativos:

-create
Cria um perfil. (Requerido)

-templatePath *template_path*
Especifica o caminho de arquivo para o modelo. (Obrigatório)

Utilize o seguinte formato:

```
-templatePath install_root/profileTemplates/template_type/cell/default
```

em que *template_type* é `xs_augment` ou `pf_augment`.

Exemplo

- Usando o modelo xs_augment:

```
./manageprofile.sh|bat -create -templatePath install_root/profileTemplates/xs_augment/cell/dmgr
-nodeProfilePath install_root/profiles/AppSrv01 -cellName cell01dmgr -nodeName node01dmgr
-appServerNodeName node01

./manageprofile.sh|bat -create -templatePath install_root/profileTemplates/xs_augment/cell/default
-dmgrProfilePath install_root/profiles/Dmgr01 -portsFile
install_root/profiles/Dmgr01/properties/portdef.props -nodePortsFile
install_root/profiles/Dmgr01/properties/nodeportdef.props -cellName cell01dmgr
-nodeName node01dmgr -appServerNodeName node01
```

- Usando o modelo pf_augment:

```
./manageprofile.sh|bat -create -templatePath install_root/profileTemplates/pf_augment/cell/dmgr
-nodeProfilePath install_root/profiles/AppSrv01 -cellName cell01dmgr -nodeName node01dmgr
-appServerNodeName node01

./manageprofile.sh|bat -create -templatePath install_root/profileTemplates/pf_augment/cell/default
-dmgrProfilePath install_root/profiles/Dmgr01 -portsFile
install_root/profiles/Dmgr01/properties/portdef.props -nodePortsFile
install_root/profiles/Dmgr01/properties/nodeportdef.props -cellName cell01dmgr
-nodeName node01dmgr -appServerNodeName node01
```

Alterar um Perfil do Gerenciador de Implementação

É possível usar o comandomanageprofiles para Alterar um perfil do gerenciador de implementação.

Parameters

-augment
Altera o perfil existente. (Requerido)

-profileName
Especifica o nome do perfil. (Requerido)

-templatePath *template_path*
Especifica o caminho para os arquivos de modelo que estão localizados no diretório raiz da instalação. (Obrigatório)

Utilize o seguinte formato:

```
-templatePath install_root/profileTemplates/template_type/dmgr
```

em que *template_type* é `xs_augment` ou `pf_augment`.

Exemplo

- Usando o modelo xs_augment:

```
./manageprofile.sh|bat -augment -profileName profile01
-templatePath install_root/profileTemplates/xs_augment/dmgr
```

- Usando o modelo pf_augment:

```
./manageprofile.sh|bat -augment -profileName profile01
-templatePath install_root/profileTemplates/pf_augment/dmgr
```

Alterar um Perfil Customizado

É possível usar o comandomanageprofiles para Alterar um perfil customizado.

Parameters

-augment
Altera o perfil existente. (Requerido)

-profileName
Especifica o nome do perfil. (Requerido)

-templatePath *template_path*
Especifica o caminho para os arquivos de modelo que estão localizados no diretório raiz da instalação. (Obrigatório)

Utilize o seguinte formato:

```
-templatePath install_root/profileTemplates/template_type/managed
```

em que *template_type* é `xs_augment` ou `pf_augment`.

Exemplo

- Usando o modelo xs_augment:

```
./manageprofile.sh|bat -augment -profileName profile01 -templatePath install_root/profileTempla
```

- Usando o modelo pf_augment:

```
./manageprofile.sh|bat -augment -profileName profile01 -templatePath install_root/profileTempla
```

Alterar um Perfil de Servidor de Aplicativos Independente

É possível usar o comandomanageprofiles para Alterar um perfil do servidor de aplicativos independente.

Parameters

-augment
Altera o perfil existente. (Requerido)

-profileName
Especifica o nome do perfil. (Requerido)

-templatePath *template_path*
Especifica o caminho para os arquivos de modelo que estão localizados no diretório raiz da instalação. (Obrigatório)

Utilize o seguinte formato:

```
-templatePath install_root/profileTemplates/template_type/default
```

em que *template_type* é `xs_augment` ou `pf_augment`.

Exemplo

- Usando o modelo xs_augment:

  ```
  ./manageprofile.sh|bat -augment -profileName profile01 -templatePath install_root/profileTemplates
  ```

- Usando o modelo pf_augment:

  ```
  ./manageprofile.sh|bat -augment -profileName profile01 -templatePath install_root/profileTemplates
  ```

Alterar o Perfil de uma Célula

É possível usar o comandomanageprofiles para Alterar um perfil de célula.

Parameters

Especifique os seguintes parâmetros para o perfil do gerenciador de implementação:

-augment
Altera o perfil existente. (Requerido)

-profileName
Especifica o nome do perfil. (Requerido)

-templatePath *template_path*
Especifica o caminho para os arquivos de modelo que estão localizados no diretório raiz da instalação. (Obrigatório)

Utilize o seguinte formato:

```
-templatePath install_root/profileTemplates/template_type/cell/dmgr
```

em que *template_type* é `xs_augment` ou `pf_augment`.

Especifique os seguintes parâmetros com o perfil do servidor de aplicativos:

-augment
Altera o perfil existente. (Requerido)

-profileName
Especifica o nome do perfil. (Requerido)

-templatePath *template_path*
Especifica o caminho para os arquivos de modelo que estão localizados no diretório raiz da instalação. (Obrigatório)

Utilize o seguinte formato:

```
-templatePath install_root/profileTemplates/template_type/cell/default
```

em que *template_type* é `xs_augment` ou `pf_augment`.

Exemplo

- Usando o modelo xs_augment:

  ```
  ./manageprofile.sh|bat -augment -profileName profile01 -templatePath install_root/profileTemplates/xs_augment/cell/dmgr

  ./manageprofile.sh|bat -augment -profileName profile01 -templatePath install_root/profileTemplates/xs_augment/cell/default
  ```

- Usando o modelo pf_augment:

  ```
  ./manageprofile.sh|bat -augment -profileName profile01 -templatePath install_root/profileTemplates/pf_augment/cell/dmgr

  ./manageprofile.sh|bat -augment -profileName profile01 -templatePath install_root/profileTemplates/pf_augment/cell/default
  ```

Retornar o Estado Inalterado de um Perfil

Para reverter o aumento de um perfil, especifique o parâmetro **-ignoreStack** com o parâmetro **-templatePath** além de especificar os parâmetros requiridos **-unaugment** e **-profileName**.

Parameters

-unaugment
Retorna o estado inalterado de um perfil alterado anteriormente. (Requerido)

-profileName
Especifica o nome do perfil. O parâmetro é emitido por padrão se nenhum valor for especificado. (Requerido)

-templatePath *template_path*
Especifica o caminho para os arquivos de modelo que estão localizados no diretório raiz da instalação. (Opcional)

Utilize o seguinte formato:

```
-templatePath install_root/profileTemplates/template_type/profile_type
```

em que *template_type* é `xs_augment` ou `pf_augment` e *profile_type* é um dos quatro tipos de perfil:

- `dmgr`: perfil do gerenciador de implementação
- `managed`: perfil customizado
- `default`: perfil do servidor de aplicativos independente
- `cell`: perfil de célula

-ignoreStack
Usado com o parâmetro **-templatePath** para alterar um perfil específico que foi alterado. (Opcional)

Exemplo

- Usando o modelo xs_augment:

```
./manageprofile.sh|bat -unaugment -profileName profile01 -ignoreStack
-templatePath install_root/profileTemplates/xs_augment/profile_type
```

- Usando o modelo pf_augment:

```
./manageprofile.sh|bat -unaugment -profileName profile01 -ignoreStack
-templatePath install_root/profileTemplates/pf_augment/profile_type
```

Perfis Não-root

Dá permissões a um usuário não-root para arquivos e diretórios para que o usuário não-root possa criar um perfil para o produto. O usuário não-root também pode alterar um perfil que foi criado por um usuário root, um usuário não-root diferente ou pelo mesmo usuário não-root.

Em um ambiente do WebSphere Application Server, usuários não raiz (não administrador) são limitados à capacidade de criar e usar perfis em seu ambiente. Dentro do plug-in do Profile Management tool, nomes exclusivos e valores de porta são desativados para usuários não raiz. O usuário não-root deve alterar os valores de campo padrão na ferramenta Gerenciamento de Perfil para o nome do perfil, o nome do nó, o nome da célula e as designações de porta. Considere designar aos usuários não-root um intervalo de valores para cada um dos campos. É possível designar responsabilidade para os usuários não-root para aderirem a seus intervalos de valor apropriados e para manterem a integridade de suas próprias definições.

O termo *instalador* refere-se a um usuário root ou não-root. Como um instalador, é possível conceder permissões de usuários não-root para criar e estabelecer seus próprios ambientes de produto. Por exemplo, um usuário não raiz pode criar um ambiente do produto para testar a implementação do aplicativo com um perfil que o usuário possui. Tarefas específicas que é possível concluir para permitir a criação de perfil não-root incluem os seguintes itens:

- Criar um perfil e designar a propriedade do diretório de perfil para um usuário não-raiz, para que o usuário não-raiz possa iniciar o WebSphere Application Server para um perfil específico.
- Conceder permissão de gravação dos arquivos e diretórios apropriados para um usuário não-root, permitindo que o usuário não-root crie o perfil. Com essa tarefa, é possível criar um grupo para usuários autorizados a criar perfis ou pode fornecer aos usuários individuais a capacidade de criar perfis.
- Instalar os pacotes de manutenção para o produto que inclui o serviço necessário para os perfis existentes de propriedade de um usuário não-root. Como o instalador, você é o proprietário de qualquer novo arquivo criado pelo pacote de manutenção.

Para obter mais informações sobre a criação de perfis para usuários não raiz, consulte Criando Perfis para Usuários Não Raiz.

Como um instalador, você também pode conceder permissões para um usuário não-root alterar perfis. Por exemplo, um usuário não raiz pode aumentar um perfil que é criado por um instalador ou aumentar um perfil criado por ele mesmo. Siga o processo de aumento do usuário não raiz do WebSphere Application Server Network Deployment.

Entretanto, quando um usuário não raiz aumenta um perfil que é criado pelo instalador, o usuário não raiz não precisa criar os seguintes arquivos antes do aumento. Os seguintes arquivos foram estabelecidos durante o processo de criação de perfil:

- *app_server_root*`/logs/manageprofiles.xml`
- *app_server_root*`/properties/fsdb.xml`
- *app_server_root*`/properties/profileRegistry.xml`

Quando um usuário não raiz aumenta um perfil que o usuário cria, o usuário não raiz deve modificar as permissões para os documentos que estão localizados dentro dos modelos de perfil do eXtreme Scale.

Atenção: Também é possível utilizar um perfil não raiz (não administrador) para o WebSphere eXtreme Scale em um ambiente independente fora do WebSphere Application Server. Você deve alterar o proprietário do diretório do ObjectGrid para o perfil não raiz. Em seguida, é possível efetuar login com esse perfil não raiz e operar o eXtreme Scale como faria normalmente para um perfil raiz (administrador).

Instalando o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client Silenciosamente

Use um arquivo completo de resposta, que você configura especificamente para suas necessidades ou transmita parâmetros para a linha de comandos para instalar de forma silenciosa o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client.

Antes de Iniciar

- Pare todos os processos em execução no ambiente do WebSphere Application Server ou do WebSphere Application Server Network Deployment. Consulte Usando ferramentas de linha de comandos de script wsadmin para obter mais informações sobre os comandos stopManager, stopNode e stopServer.

  CUIDADO:
  Certifique-se de que quaisquer processos em execução sejam interrompidos. Se os processos em execução não forem interrompidos, a instalação continuará, criando resultados imprevisíveis e deixando a instalação em um estado indeterminado em algumas plataformas.
- Verifique se o diretório de instalação de destino está vazio ou não existe.

  Importante: Se uma versão anterior do WebSphere eXtreme Scale ou componente ObjectGrid existir no diretório que você especificar para instalar a Versão 7.1, o produto não será instalado. Por exemplo, é possível ter uma pasta `<wxs_install_root>/ObjectGrid` previamente existente. É possível selecionar um diretório de instalação diferente ou cancelar a instalação. Em seguida, remova a instalação anterior e execute o assistente novamente.

Sobre Esta Tarefa

Uma instalação silenciosa utiliza o mesmo programa que a versão de interface gráfica com o usuário (GUI) utiliza. Entretanto, no lugar de exibir uma interface de assistente, a instalação silenciosa lê todas as suas respostas de um arquivo que você customiza ou dos parâmetros que você transmite para a linha de comandos. Consulte um exemplo de um "Arquivo `wxssetup.response.txt`" na página 35, que inclui uma descrição de cada opção.

Procedimento

1. Opcional: Se você escolher instalar o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client usando um arquivo de resposta, primeiro customize o arquivo `wxssetup.response.txt`.

   Lembre-se: É necessário especificar o nome completo do arquivo de resposta. Especificar o caminho relativo faz com que a instalação falhe sem nenhuma indicação de que ocorreu um erro.

   a. Faça uma cópia do arquivo de resposta a ser customizado.

      Para a instalação completa do WebSphere eXtreme Scale, copie o arquivo de resposta do DVD do produto para sua unidade de disco.

      Para o WebSphere eXtreme Scale Client, descompacte o arquivo zip do WebSphere eXtreme Scale Client em seu disco rígido e localize o arquivo de resposta.

   b. Abra e edite o arquivo de resposta no editor de texto de sua escolha. O arquivo de resposta de exemplo anterior fornece detalhes sobre como especificar cada um dos parâmetros. É necessário especificar os seguintes parâmetros:

      - O contrato de licença
      - O diretório de instalação

      Dica: Quando você instala o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client em um ambiente do WebSphere Application Server, o instalador usa o diretório de instalação para determinar onde a instância

existente do WebSphere Application Server está instalada. Se você instalar em um nó que contenha várias instâncias do WebSphere Application Server, defina claramente o seu local.

c. Execute o seguinte script para iniciar a instalação.

Para a instalação completa do WebSphere eXtreme Scale:

```
./install.sh|bat -options C:/drive_path/response_file.txt -silent
```

Para a instalação do WebSphere eXtreme Scale Client:

```
./WXS_Client/install.sh|bat -options C:/drive_path/response_file.txt -silent
```

Você também pode usar o arquivo de resposta quando executar uma instalação da GUI. O arquivo de resposta pode ser usado com a instalação da GUI para depurar problemas que ficam ocultos com a instalação silenciosa. Quando especifica o arquivo `wxssetup.response` para a GUI ou para instalações silenciosas, você deve usar o caminho completo. Execute o seguinte script para executar a instalação da GUI com seu arquivo de resposta:

- Linux UNIX `<install_home>/install.sh -options <full_install_path_required>/wxssetup.response`
- Windows `<install_home>\install.exe -options c:\<full_install_path_required>\wxssetup.response`

2. Opcional: Se você escolher instalar o eXtreme Scale ao transmitir determinados parâmetros para a linha de comandos, execute o seguinte script para iniciar a instalação:

Para a instalação completa do WebSphere eXtreme Scale:

```
./install.sh|bat -silent -OPT silentInstallLicenseAcceptance=true -OPT installLocation=install_location
```

Para a instalação do WebSphere eXtreme Scale Client:

```
./WXS_Client/install.sh|bat -silent -OPT silentInstallLicenseAcceptance=true -OPT installLocation=install_location
```

Parâmetros de Instalação

Especifique os parâmetros na linha de comandos para customizar e configurar a instalação do produto.

Nota: É necessário especificar o nome completo do arquivo de resposta. Especificar o caminho relativo faz com que a instalação falhe sem nenhuma indicação de que ocorreu um erro.

Parâmetros

É possível transmitir os seguintes parâmetros durante a instalação da linha de comandos ou do arquivo de opções do produto:

-silent
Suprime a interface gráfica com o usuário (GUI). Especifique o parâmetro **-options** para indicar que o instalador conclui a instalação de acordo com um arquivo de opções customizadas. Se você não especificar o parâmetro **-options**, os valores padrão serão usados no lugar.

Exemplo de Uso

```
./install.sh|bat -silent -options options_file.txt
```

-options *path_name/file_name*
Especifica um arquivo de opções usado pelo instalador para executar uma instalação silenciosa. As propriedades na linha de comandos têm precedência.

Exemplo de Uso

```
./install.sh|bat -options c:/path_name/options_file.txt
```

-log #!file_name *@event_type*
Gera um arquivo de log de instalação que registra os seguintes tipos de eventos:

- err
- wrn
- msg1
- msg2
- dbg
- TODOS

Exemplo de Uso

```
./install.sh|bat -log # !c:/temp/logfiles.txt @ALL
```

-is:log *path_name/file_name*
Cria um arquivo de log que contém as procuras do Java Virtual Machine (JVM) do instalador enquanto tenta iniciar a GUI. O arquivo de log não é criado, a menos que seja especificado.

Exemplo de Uso

```
./install.sh|bat -is:log c:/logs/javalog.txt
```

-is:javaconsole
Exibe uma janela do console durante o processo de instalação.

Exemplo de Uso

```
./install.sh|bat -is:javaconsole
```

-is:silent
Suprime a janela de inicialização Java que é exibida quando o instalador é iniciado.

Exemplo de Uso

```
./install.sh|bat -is:silent
```

-is:tempdir *path_name*
Especifica o diretório temporário que o instalador usa durante a instalação.

Exemplo de Uso

```
./install.sh|bat -is:tempdir c:/temp
```

Usando o Update Installer para instalar os pacotes de manutenção

Use o IBM Update Installer para atualizar o ambiente do WebSphere eXtreme Scale com vários tipos de manutenção, como correções temporárias, fix packs e pacotes de atualização.

Sobre Esta Tarefa

Use o IBM Update Installer para instalar e aplicar vários tipos de pacotes de manutenção para o WebSphere eXtreme Scale. Como o Update Installer executa manutenção regular, é necessário usar a versão mais atual da ferramenta.

Procedimento

1. Pare todos os processos que estão em execução no seu ambiente.

- Para parar todos os processos que estão em execução em seu ambiente do eXtreme Scale independente, consulte “Parando Servidores eXtreme Scale Independentes” na página 337 para obter mais informações.
- Para parar todos os processos que estão em execução no seu ambiente do WebSphere Application Server independente, consulte Utilitários da Linha de Comandos .

2. Faça download da versão mais recente do Update Installer. Consulte Correções recomendadas para obter mais informações.
3. Instale o Update Installer. Consulte Instalando o Update Installer para WebSphere Software no Centro de Informações do WebSphere Application Server para obter mais informações.
4. Faça download no diretório *updi_root*/maintenance dos pacotes de manutenção que deseja instalar. Consulte o Site de Suporte para obter mais informações.
5. Utilize o Update Installer para instalar a correção temporária, fix pack ou pacote de atualizações. É possível instalar o pacote de manutenção executando a interface gráfica com o usuário (GUI) ou executando o Update Installer no modo silencioso.

   Execute o seguinte comando a partir do diretório *updi_root* para iniciar a GUI:

   - Linux UNIX `update.sh`
   - Windows `update.bat`

   Execute o seguinte comando a partir do diretório *updi_root* para executar o Update Installer no modo silencioso:

   - Linux UNIX `./update.sh -silent -options responsefile/file_name`
   - Windows `update.bat -silent -options responsefile\file_name`

   Se o processo de instalação falhar, consulte o arquivo de log temporário, que está no diretório *updi_root*/logs/update/tmp. O Update Installer cria o diretório *install_root*/logs/update/*maintenance_package*.install no qual os arquivos de log de instalação estão localizados.

Desinstalando o WebSphere eXtreme Scale

Para remover o WebSphere eXtreme Scale do seu ambiente, é possível usar o assistente ou desinstalar silenciosamente o produto.

Antes de Iniciar

Atenção: O desinstalador remove todos os arquivos binários e toda a manutenção, como fix packs e correções temporárias, ao mesmo tempo.

Procedimento

1. Pare todos os processos que executam o eXtreme Scale.

   CUIDADO:
   Certifique-se de que quaisquer processos em execução sejam interrompidos. Se os processos em execução não forem interrompidos, a desinstalação continuará, criando resultados imprevisíveis e deixando a desinstalação em um estado indeterminado em algumas plataformas.

   - Se você instalou o eXtreme Scale independente, leia sobre como parar servidores independentes para parar processos.

- Se você instalou o eXtreme Scale com uma instalação existente do WebSphere Application Server, leia sobre utilitários de linha de comandos para obter mais informações sobre como parar processos do WebSphere Application Server.

2. Desinstale o produto. É possível executar a desinstalação em uma GUI ou silenciosamente.

   Nota: Ao especificar o arquivo `wxssetup.response` do arquivo de resposta para instalações ou desinstalações silenciosas ou da GUI, o caminho completo deve sempre ser especificado. O arquivo de resposta é opcional para a desinstalação da GUI.

   - **Para executar uma desinstalação usando a GUI:**
     - Linux UNIX `<install_home>/uninstall_wxs/uninstall`
     - Windows `<install_home>\uninstall_wxs\uninstall.exe`

     Se você quiser executar a desinstalação usando a GUI e o arquivo `wxssetup.response`, use um dos comandos a seguir:

     - Linux UNIX

       ```
       <install_home>/uninstall_wxs/uninstall -options
       <full_install_path_required>/wxssetup.response
       ```

     - Windows

       ```
       <install_home>\uninstall_wxs\uninstall.exe -options
       <full_install_path_required>\wxssetup.response
       ```

   - **Para executar a desinstalação silenciosamente usando o script `wxssetup.response` do arquivo de resposta:**
     - Linux UNIX

       ```
       <install_home>/uninstall_wxs/uninstall -options
       <full_install_path_required>/wxssetup.response -silent
       ```

     - Windows

       ```
       <install_home>\uninstall_wxs\uninstall.exe -options
       <full_install_path_required>\wxssetup.response -silent
       ```

Resultados

Você removeu o eXtreme Scale do seu ambiente.

Capítulo 4. Customizando o WebSphere eXtreme Scale for z/OS

Usando o WebSphere Customization Tools, é possível gerar e executar tarefas customizadas para customizar o WebSphere eXtreme Scale para z/OS.

Antes de Iniciar

- Verifique se o seu sistema contém o nível mais recente do WebSphere Application Server Network Deployment:
  - Se você estiver executando a Versão 6.1, seu sistema deverá conter o fix pack 31 no mínimo. Consulte Instalando o ambiente de serviço de aplicativo da Versão 6.1 para obter mais informações.
  - Se você estiver executando a Versão 7.0, seu sistema deverá conter o fix pack 9 no mínimo. Consulte Instalando o ambiente de serviço de aplicativo da Versão 7.0 para obter mais informações.
- Instale o WebSphere eXtreme Scale para z/OS. Consulte o Diretório do Programa do WebSphere eXtreme Scale *WebSphere eXtreme Scale* na Página de Biblioteca para obter mais informações.

Sobre Esta Tarefa

Usando o WebSphere Customization Tools, gere definições de customização e faça upload e execute tarefas customizadas para customizar o WebSphere eXtreme Scale para z/OS. Consulte os tópicos a seguir para obter mais informações:

Procedimento

- "Instalando o WebSphere Customization Tools"
- "Gerando Definições de Customização" na página 61
- "Fazendo Upload e Executando Tarefas Customizadas" na página 62

Instalando o WebSphere Customization Tools

Instale o WebSphere Customization Tools Versão 7.0.0.6 ou posterior para customizar o WebSphere eXtreme Scale para o ambiente do z/OS.

Antes de Iniciar

Instale o WebSphere eXtreme Scale para z/OS. Consulte o Diretório do Programa do *WebSphere eXtreme Scale* na Página de Biblioteca para obter mais informações.

Sobre Esta Tarefa

WebSphere Customization Tools é uma ferramenta gráfica baseada na estação de trabalho que você usa para criar jobs customizados que constróem ambientes de tempo de execução do WebSphere eXtreme Scale for z/OS.

Procedimento

1. Use o FTP para copiar os arquivos de extensão `xs.wct` e `xspf.wct` a partir do sistema z/OS para a estação de trabalho na qual você está instalando o

WebSphere Customization Tools. Os arquivos de extensão estão no diretório `/usr/lpp/zWebSphereXS/util/V7R1/WCT` no sistema z/OS.

2. Faça o download e instale o WebSphere Customization Tools Versão 7.0.0.6 ou posterior a partir do Web site apropriado:
   - Windows WebSphere Customization Tools para Windows
   - Linux WebSphere Customization Tools para Linux®
3. Transfira por upload o arquivo `xs.wct` para o aplicativoWebSphere Customization Tools.
   a. Inicie o aplicativo WebSphere Customization Tools na sua estação de trabalho.
   b. Clique em **Ajuda** → **Atualizações de Software** → **Instalar Extensão**.
   c. No painel Locais de Extensão do WebSphere Customization Tools, clique em **Instalar novo local de extensão**.
   d. No painel Arquivo Archive de Origem, clique em **Procurar**, navegue para o diretório no qual você copiou o arquivo `xs.wct` na etapa 1 e clique em **Abrir**.
   e. Clique em **Avançar** no painel Resumo.

      Nota: O painel Instalação com Êxito é exibido. Antes de clicar em **Concluir**, é necessário copiar e salvar os dados no campo de local:

      ```
      C:\Documents and Settings\Administrator\WCT\workspace\configuration\
      com.ibm.ws.pmt.update\com.ibm.ws390.pmt.xs_7.1.0.0\eclipse
      ```

   f. No painel Configuração do Produto, clique em **Incluir um Local de Extensão**. Cole os dados copiados na etapa anterior no campo Local e clique em **OK**.
   g. Clique em **Sim** para reiniciar o WebSphere Customization Tools.
4. Transfira por upload o arquivo `xspf.wct` para o aplicativo WebSphere Customization Tools.
   a. Clique em **Ajuda** → **Atualizações de Software** → **Instalar Extensão**.
   b. No painel Locais de Extensão do WebSphere Customization Tools, clique em **Instalar novo local de extensão**.
   c. No painel Arquivo Archive de Origem, clique em **Procurar**, navegue para o diretório no qual você copiou o arquivo `xspf.wct` na etapa 1 e clique em **Abrir**.
   d. Clique em **Avançar** no painel Resumo.

      Nota: O painel Instalação com Êxito é exibido. Antes de clicar em **Concluir**, é necessário copiar e salvar os dados do campo de local:

      ```
      C:\Documents and Settings\Administrator\WCT\workspace\configuration\
      com.ibm.ws.pmt.update\com.ibm.ws390.pmt.xs_7.1.0.0\eclipse
      ```

   e. No painel Configuração do Produto, clique em **Incluir um Local de Extensão**. Cole os dados copiados na etapa anterior no campo Local e clique em **OK**.
   f. Clique em **Sim** para reiniciar o WebSphere Customization Tools.

O que Fazer Depois

Depois de fazer upload dos dois arquivos de extensão e de reiniciar o WebSphere Customization Tools, use a ferramenta de Gerenciamento de Perfil para gerar definições de customização para o eXtreme Scale para z/OS. Consulte o "Gerando Definições de Customização" na página 61 para obter informações adicionais.

Gerando Definições de Customização

Utilize a função Profile Management Tool no WebSphere Customization Tools para gerar definições de customização e criar tarefas customizadas para WebSphere eXtreme Scale para z/OS.

Antes de Iniciar

Instale o WebSphere Customization Tools e faça upload dos arquivos de extensão `xs.wct` e `xspf.wct`. Consulte o "Instalando o WebSphere Customization Tools" na página 59 para obter mais informações.

Sobre Esta Tarefa

É possível gerar definições de customização usando a Profile Management Tool, que é fornecido no WebSphere Customization Tools. Uma *definição de customização* é um conjunto de arquivos usados para criar jobs customizados para configurar o WebSphere eXtreme Scale for z/OS.

Procedimento

1. Inicie a Ferramenta Gerenciamento de Perfis.
   - Windows Clique em **Iniciar** → **Programas** → **IBM WebSphere** → **WebSphere Customization Tools**. Depois que o aplicativo for iniciado, clique na guia **Profile Management Tool**.
   - Linux Clique em *operating_system_menus* → **IBM WebSphere** → **WebSphere Customization Tools**. Depois que o aplicativo for iniciado, clique na guia **Profile Management Tool**.
2. Inclua um local existente ou crie um local da definição de customização que você deseja criar. Na guia **Locais de Customização**, clique em **Incluir**. Se você criar um local, a caixa Versão fará referência à versão do produto WebSphere Application Server existente instalada no seu sistema z/OS.

   Nota: Não use o mesmo local que está sendo usado para outras definições de customização do eXtreme Scale.
3. Gere a definição de customização. Na guia **Definições de Customização**, clique em **Aumentar**.
4. Selecione o tipo de ambiente de definição a ser criado:
   - Nó do Servidor de Aplicativos Independente
   - Gerenciador de implementação
   - Servidor de Aplicativos
   - Nó gerenciado (customizado)
5. Preencha os campos nos painéis. Especifique os valores dos parâmetros que são usados para criar o sistema z/OS.
6. Clique em **Aumentar** para gerar a definição de customização.

O que Fazer Depois

Faça upload da tarefa customizada para o sistema z/OS de destino. Consulte o "Fazendo Upload e Executando Tarefas Customizadas" na página 62 para obter mais informações.

Fazendo Upload e Executando Tarefas Customizadas

Após gerar as definições de customização, será possível fazer o upload e executar tarefas customizadas associadas com as definições para seu WebSphere eXtreme Scale para sistema z/OS.

Antes de Iniciar

Crie as definições de customização para as tarefas que você deseja transferir por upload para um sistema z/OS. Consulte o "Gerando Definições de Customização" na página 61 para obter informações adicionais.

Sobre Esta Tarefa

Faça o upload e execute as tarefas customizadas que você criou utilizando o WebSphere Customization Tools para administrar e monitorar o WebSphere eXtreme Scale para o ambiente do z/OS.

Procedimento

1. Fazer upload das tarefas customizadas. Na guia **Definições de Customização**, selecione as tarefas que deseja fazer upload e clique em **Processar**.
2. Faça upload das tarefas para o servidor de FTP no seu sistema z/OS. Especifique as informações necessárias no painel **Fazer Upload da Definição de Customização**.
3. Clique em **Concluir**.
4. Execute as tarefas customizadas. Clique na guia **Instruções de Customização** e siga as instruções de customização para cada tarefa.

Capítulo 5. Planejando Implementação do Aplicativo

Antes de implementar aplicativos no WebSphere eXtreme Scale, você deve rever os requisitos de hardware e software, implementação de rede e configurações de ajuste, configurações de implementação, e assim por diante. Você também pode usar a verificação operacional para garantir que seu ambiente esteja pronto para ter um aplicativo implementado.

Para uma discussão das boas práticas que é possível usar ao projetar seus aplicativos WebSphere eXtreme Scale, leia o seguinte artigo em developerWorks: Princípios e boas práticas para construir aplicativos WebSphere eXtreme Scale de alta execução e alta resiliência.

Visão Geral da Implementação do Aplicativo

Antes de utilizar o WebSphere eXtreme Scale em um ambiente de produção, considere os seguintes problemas para otimizar sua implementação.

Planejando Implementação do Aplicativo

A lista a seguir inclui itens a serem considerados:

- Número de sistemas e processadores: Quantas máquinas físicas e processadores são necessários no ambiente?
- Número de servidores: Quantos servidores do eXtreme Scale para hospedar mapas do eXtreme Scale?
- Número de partições: A quantidade de dados armazenados nos mapas é um fator na determinação do número de partições necessárias.
- Número de réplicas: Quantas réplicas são necessárias para cada primário no domínio?
- Replicação síncrona ou assíncrona: Os dados são vitais, portanto, tal replicação síncrona é necessária? Ou o desempenho é a maior prioridade, tornando a replicação assíncrona a escolha correta?
- Tamanhos de heap: Qual é quantidade de dados a ser armazenada em cada servidor?

Requisitos de Hardware e Software

Pesquisar uma visão geral de hardware e de requisitos do sistema operacional. Embora não seja obrigatório você usar um nível específico de hardware ou sistema operacional para WebSphere eXtreme Scale, uma lista detalhada de opções de hardware e software formalmente suportados pelo sistema operacional está disponível na página Requisitos de Sistemas do site de suporte ao produto. Se houver um conflito entre as informações fornecidas no centro de informações e as informações sobre a página Requisitos do Sistema, as informações no Web site tem precedência. As informações de pré-requisito no centro de informações são fornecidas apenas como uma conveniência.

Consulte a página Requisitos do Sistema.

Requisitos de Hardware

O WebSphere eXtreme Scale não requer um nível específico de hardware. Os requisitos de hardware dependem do hardware suportado para a instalação do Java Platform, Standard Edition que é utilizado para executar o WebSphere eXtreme Scale. Se você estiver usando o eXtreme Scale com oWebSphere Application Server ou outra implementação do Java Platform, Enterprise Edition, os requisitos de hardware dessas plataformas são suficientes para oWebSphere eXtreme Scale.

Requisitos do Sistema Operacional

O eXtreme Scale não requer um nível de sistema operacional específico. Cada implementação de Java SE e Java EE necessita de diferentes níveis de sistema operacional ou correções para problemas que são descobertos durante o teste da implementação de Java. Os níveis que as implementações necessitam são suficientes para oeXtreme Scale.

Requisitos do WebSphere Application Server

Clientes e servidores do eXtreme Scale em execução em um ambiente distribuído e ObjectGrids de memória local são suportados no WebSphere Application Server Versão 6.0.2 e superior.

Nota: Para usar o provedor de cache dinâmico, o seu sistema deve satisfazer a um dos seguintes requisitos mínimos:

- WebSphere Application Server Versão 6.1.0.25 ou superior e Correção Temporária PK85622
- WebSphere Application Server Versão 7.0.0.3 ou superior e Correção Temporária PK85622

Consulte as Correções recomendadas para o WebSphere Application Server para obter mais informações.

Outros Requisitos do Servidor de Aplicativos

Outras implementações de Java EE podem usar o tempo de execução do eXtreme Scale como uma instância local ou como um cliente para servidores eXtreme Scale. Para implementar o Java SE, você deve usar a Versão 1.4.2 ou mais recente.

Considerações sobre o Java Platform, Enterprise Edition

Conforme você se prepara para integrar o WebSphere eXtreme Scale em um ambiente Java Platform, Enterprise Edition, considere certos itens, como opções de configuração, requisitos e limitações, além de implementação e gerenciamento de aplicativo.

Executando Aplicativos eXtreme Scale em um Ambiente Java EE

Um aplicativo Java EE pode se conectar a um aplicativo eXtreme Scale remoto. Além disso, o ambiente do WebSphere Application Server suporta o início de um servidor eXtreme Scale conforme um aplicativo é iniciado no servidor de aplicativos.

Se você utilizar um arquivo XML para criar uma instância de ObjectGrid, e o arquivo XML estiver no módulo do arquivo enterprise archive (EAR), acesse o

arquivo utilizando o método getClass().getClassLoader().getResource("META-INF/objGrid.xml") para obter um objeto URL para utilizar para criar uma instância de ObjectGrid. Substitua o nome do arquivo XML que você está utilizando na chamada de método.

É possível utilizar beans de inicialização para um aplicativo para autoinicializar uma instância do ObjectGrid quando o aplicativo for iniciado e para destruir a instância quando o aplicativo for parado. Um bean de inicialização é um bean de sessão stateless com um local remoto com.ibm.websphere.startupservice.AppStartUpHome e uma interface remota com.ibm.websphere.startupservice.AppStartUp. A interface remota possui dois métodos: o método de iniciar e o método de parar. Utilize o método start para autoinicializar a instância e o método stop para destruir a instância. O aplicativo utiliza o método ObjectGridManager.getObjectGrid para manter uma referência à instância. Consulte as informações sobre como acessar um ObjectGrid com o ObjectGridManager no *Guia de Programação* para obter mais informações.

Utilizando Carregadores de Classes

Quando módulos aplicativos que utilizam carregadores de classes diferentes compartilharem uma única instância de ObjectGrid em um aplicativo Java EE, verifique os objetos que estão armazenados no eXtreme Scale e se os plug-ins para o produto estão em um Utilitário de Carga comum no aplicativo.

Gerenciando o Ciclo de Vida de Instâncias do ObjectGrid em um Servlet

Para gerenciar o ciclo de vida de uma instância do ObjectGrid em um servlet, é possível utilizar o método init para criar a instância e o método destroy para remover a instância. Se a instância estiver armazenada em cache, ela será recuperada e manipulada no código do servlet. Consulte as informações sobre como acessar um ObjectGrid com o ObjectGridManager no *Guia de Programação* para obter mais informações.

Topologia de Armazenamento em Cache: Armazenamento em Cache em Memória e Distribuído

Com WebSphere eXtreme Scale, sua arquitetura pode utilizar armazenamento em cache de dados em memória local ou armazenamento em cache de dados de cliente/servidor distribuídos.

O WebSphere eXtreme Scale requer infraestrutura adicional mínima para operar. A infraestrutura consiste em scripts para instalar, iniciar e parar um aplicativo Java Platform, Enterprise Edition em um servidor. Os dados armazenados em cache são armazenados no servidor eXtreme Scale e os cliente conectam-se remotamente ao servidor.

Caches distribuídos oferecem desempenho, disponibilidade e escalabilidade melhorados e podem ser configurados usando topologias dinâmicas, nas quais os servidores são equilibrados automaticamente. Também é possível incluir servidores adicionais sem restaurar seus servidores eXtreme Scale existentes. É possível criar implementações simples ou grandes a nível de terabytes, em que milhares de servidores são necessários.

Cache de Memória Local

Em um caso mais simples, o eXtreme Scale pode ser utilizado como um cache de grade de dados em memória local (não distribuído). O caso local pode beneficiar especialmente aplicativos de alta simultaneidade nos quais vários encadeamentos precisam acessar e modificar dados transientes. Os dados mantidos em uma grade do eXtreme Scale local podem ser indexados e recuperados usando o suporte de consulta do WebSphere eXtreme Scale. A habilidade de consultar os dados pode ajudar muito os desenvolvedores ao trabalharem com grandes conjuntos de dados de memória em relação o suporte limitado da estrutura de dados oferecido pelo Java Virtual Machine (JVM), que está pronto para ser usado como está.

A topologia de cache em memória local para eXtreme Scale é usado para oferecer acesso transacional e consistente aos dados temporários dentro de uma única Java virtual machine.

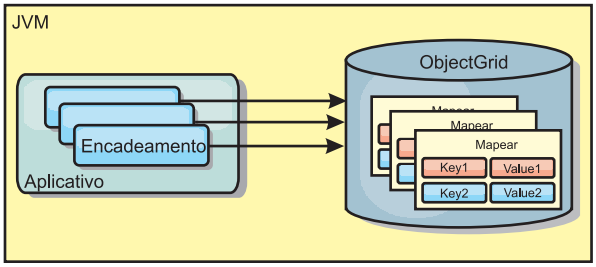


Figura 1. Cenário de Cache em Memória Local

Vantagens

- Configuração simples: Um ObjectGrid pode ser criado programaticamente ou declarativamente com o arquivo XML do descritor de implementação ObjectGrid ou com outras estruturas como Spring.
- Rápido: Cada BackingMap pode ser ajustado de maneira independente para utilização de memória e simultaneidade ideais.
- Ideal para topologias de uma única Java virtual machine com pequenos conjuntos de dados ou para armazenamento em cache de dados frequentemente acessados.
- Transacional. As atualizações BackingMap podem ser agrupadas em uma única unidade de trabalho e podem ser integradas como um último participante nas transações de duas fases como transações JTA (Java Transaction Architecture).

Desvantagens

- Não tolerante a falhas.
- Os dados não são replicados. Caches em memória são melhores para dados de referência somente para leitura.
- Não escalável. A quantidade de memória necessária pelo banco de dados pode ultrapassar a capacidade da Java virtual machine.
- Ocorrem problemas na inclusão de Java virtual machines:
  - Os dados não podem ser facilmente particionados
  - Você deve replicar manualmente o estado entre as Java virtual machines ou cada instância do cache poderá ter diferentes versões dos mesmos dados.
  - A invalidação é custosa.

– Cada cache deve ser aquecido de maneira independente. O aquecimento é o período de carregamento de um conjunto de dados para que o cache seja preenchido com dados válidos.

Quando Utilizar

A topologia de implementação de cache em memória local deve ser usada somente quando a quantidade de dados a serem armazenados em cache for pequena (puder ser colocada em uma única Java virtual machine) e for relativamente estável. Dados antigos devem ser tolerados com esta abordagem. A utilização de evictors para manter os dados mais frequentemente ou recentemente usados no cache pode ajudar a diminuir o tamanho do cache e a aumentar a relevância dos dados.

Cache em Memória Local Replicado pelo Peer

Para um cache do WebSphere eXtreme Scale local, você deve garantir que o cache seja sincronizado se houver vários processos com instâncias de cache independentes. Para isso, ative um cache replicado por peer com JMS.

WebSphere eXtreme Scale inclui dois plug-ins que propagam automaticamente mudanças de transação entre instâncias do ObjectGrid peer. O plug-in JMSObjectGridEventListener automaticamente propaga alterações do eXtreme Scale usando o JMS (Java Messaging Service).

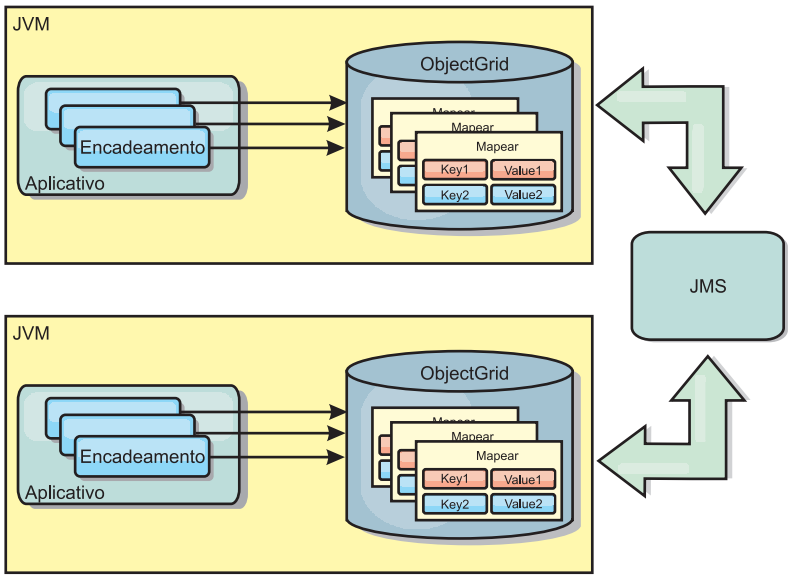


Figura 2. Cache Replicado pelo Peer com Alterações que são Propagadas com JMS

Se você estiver executando em um ambiente WebSphere Application Server, o plug-in TranPropListener também está disponível. O plug-in TranPropListener usa o gerenciador de alta disponibilidade (HA) para propagar as alterações em cada instância do cache do eXtreme Scale de peer.

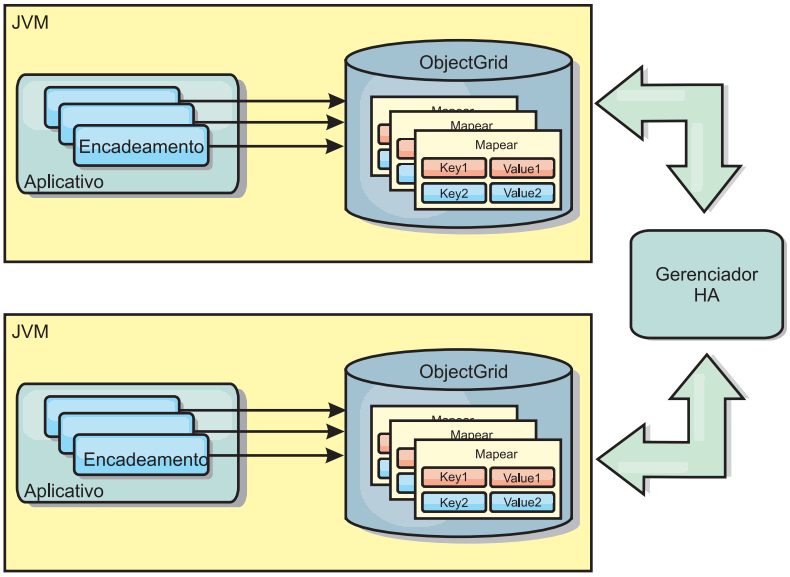


Figura 3. Cache Replicado pelo Peer com Alterações que são Propagadas com o Gerenciador de Alta Disponibilidade

Vantagens

- Os dados são mais válidos porque os dados são atualizados mais frequentemente.
- Com o plug-in TranPropListener, como no ambiente local, o eXtreme Scale pode ser criado programaticamente ou declarativamente com o arquivo XML descritor de implementação do eXtreme Scale ou com outras estruturas como Spring. A integração com o gerenciador de alta disponibilidade é feita automaticamente.
- Cada BackingMap pode ser independentemente ajustado para melhor utilização da memória e concorrência.
- As atualizações BackingMap podem ser agrupadas em uma única unidade de trabalho e podem ser integradas como um último participante nas transações de duas fases como transações JTA (Java Transaction Architecture).
- Ideal para poucas topologias JVM com um conjunto de dados razoavelmente pequeno ou para armazenamento em cache de dados frequentemente acessados.
- As atualizações em cada eXtreme Scale são replicadas para todas as instâncias do eXtreme Scale do peer. As alterações são consistentes desde que uma assinatura durável seja utilizada.

Desvantagens

- A configuração e a manutenção para o JMSObjectGridEventListener podem ser complexas. O eXtreme Scale pode ser criado programaticamente ou declarativamente com o arquivo XML descritor de implementação do eXtreme Scale ou com outras estruturas tais como Spring.
- Não escalável: A quantidade de memória necessária para que o banco de dados possa dominar a JVM.
- Funciona inadequadamente ao incluir Java Virtual Machines:
  - Os dados não podem ser facilmente particionados
  - A invalidação é custosa.
  - Cada cache deve ser aquecido de maneira independente

Quando Utilizar

Esta topologia de implementação deve ser utilizada apenas quando a quantidade de dados a ser armazenada em cache for pequena (podendo ajustar-se a uma única JVM) e for relativamente estável.

Cache Distribuído

O WebSphere eXtreme Scale é mais frequentemente usado como um cache compartilhado, para fornecer acesso transacional a dados para múltiplos componentes onde, caso contrário, um banco de dados tradicional seria usado. O cache compartilhado elimina a necessidade de configurar um banco de dados.

O cache é coerente porque todos os clientes vêem os mesmos dados no cache. Cada pedaço de dado é armazenado em exatamente um servidor no cache, evitando cópias de registros desperdiçadas que poderiam potencialmente conter diferentes versões dos dados. Um cache coerente também pode conter mais dados à medida que mais servidores são incluídos à grade, e escalar de forma linear à medida que a grade cresce em tamanho. Porque os clientes acessam dados desta grade com chamadas procedurais remotas, ela também é conhecida como um cache remoto (ou cache distante). Pelo particionamento de dados, cada processo contém um subconjunto exclusivo do conjunto de dados total. Grades maiores podem conter mais dados e atender mais pedidos para tais dados. A coerência também elimina a necessidade de inserir dados de invalidação ao redor da grade porque não há dados antigos. O cache coerente retém somente a cópia mais recente de cada pedaço de dados.

Se você estiver executando um ambiente doWebSphere Application Server, o plug-in TranPropListener também estará disponível. O plug-in TranPropListener usa o componente de alta disponibilidade (Gerenciador HA) do WebSphere Application Server para propagar as alterações para cada instância de cache ObjectGrid de peer.

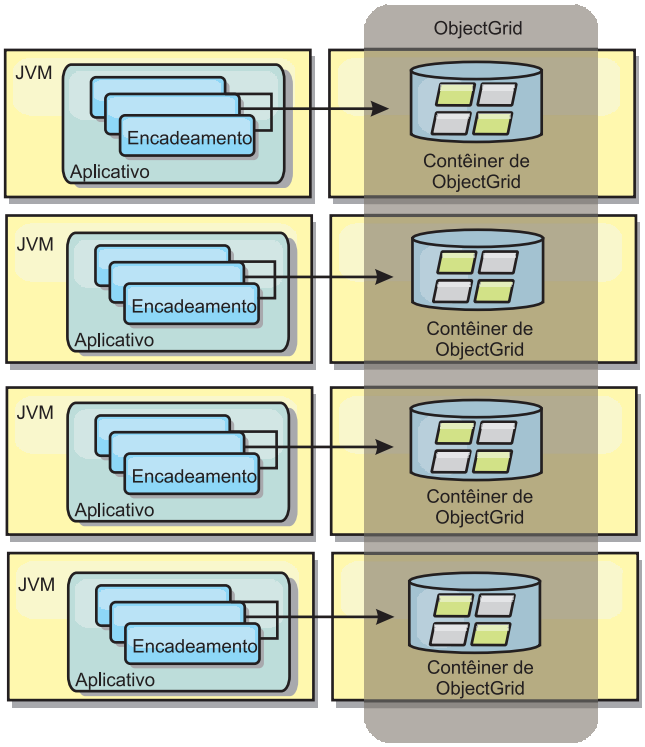


Figura 4. Cache Distribuído

Cache Local

Opcionalmente, os clientes têm um cache local, sequencial quando o eXtreme Scale é usado em uma topologia distribuída. Este cache opcional é chamado de cache local, um ObjectGrid independente em cada cliente, servindo como um cache para o cache remoto, do lado do cliente. O cache local é ativado por padrão quando o bloqueio é configurado como otimista ou nenhum e não pode ser utilizado quando é configurado como pessimista.

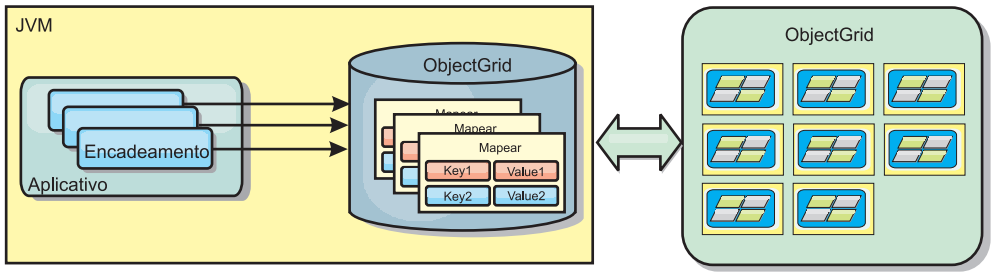


Figura 5. Cache Local

Um cache local é muito rápido porque fornece acesso em memória a um subconjunto do conjunto inteiro de dados em cache que é armazenado remotamente nos servidores do eXtreme Scale. O cache local não é particionado e contém dados de qualquer uma das partições eXtreme Scale remotas. O WebSphere eXtreme Scale pode ter até três camadas de cache, como a seguir.

1. O cache da camada da transação contém todas as alterações para uma única transação. O cache da transação contém uma cópia de trabalho dos dados até que a transação seja confirmada. Quando uma transação do cliente solicita dados de um ObjectMap, a transação é verificada primeiro

2. O cache local na camada do cliente contém um subconjunto de dados da camada do servidor. Quando a camada da transação não possui os dados, eles são buscados em um cache local, se disponíveis, e inseridos no cache da transação.
3. A grade na camada do servidor contém a maioria dos dados e é compartilhada entre todos os clientes. A camada do servidor pode ser particionada, o que permite que uma grande quantidade de dados seja armazenada em cache. Quando o cache local do cliente não possui os dados, eles são buscados na camada do servidor e inseridos no cache cliente. A camada do servidor também pode ter um plug-in do Utilitário de Carga. Quando a grade não tem os dados necessários, o Utilitário de Carga é chamado e os dados resultantes são inseridos do armazém de dados de backend para a grade.

Para desativar o cache local, configure o atributo numberOfBuckets como 0 no arquivo descritor do ObjectGrid de substituição do cliente. Consulte o tópico sobre bloqueio de entrada de mapa para obter detalhes sobre estratégias de bloqueio do eXtreme Scale. O cache local também pode ser configurado para ter uma política de despejo configurada e diferentes plug-ins usando uma configuração do descritor do eXtreme Scale de substituição.

Vantagem

- Tempo de resposta rápido porque todos os acessos aos dados é local.

Desvantagens

- Aumenta a duração dos dados antigos.
- Deve utilizar um evictor para invalidar dados para evitar a falta de memória.

Quando Utilizar

Utilize quando o tempo de resposta for importante e dados antigos puderem ser tolerados.

Cache integrado

As grades do eXtreme Scale podem ser executadas nos processos existentes como servidores eXtreme Scale integrados ou podem ser gerenciadas como processos externos. As grades integradas são úteis quando você está executando em um servidor de aplicativos, como o WebSphere Application Server. É possível iniciar servidores eXtreme Scale que não são integrados usando scripts da linha de comandos e executar em um processo Java.

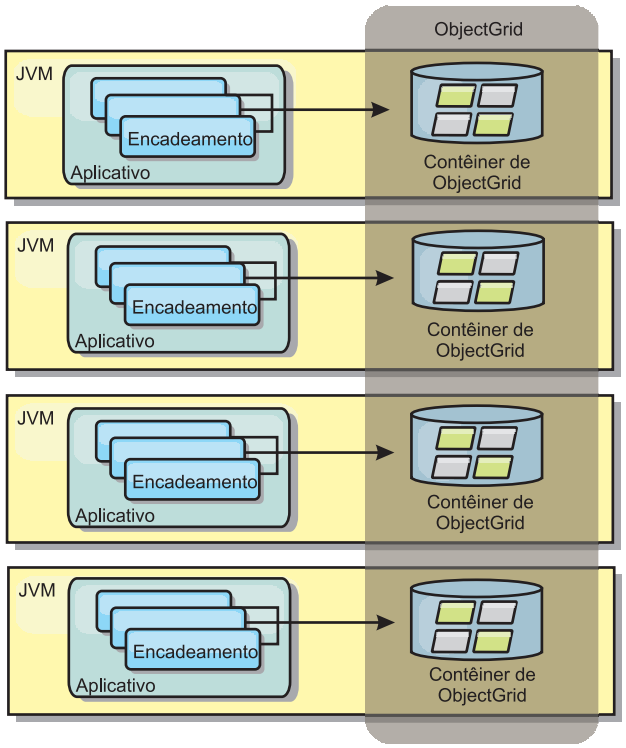


Figura 6. Cache Integrado

Vantagens

- Administração simplificada já que há menos processos para gerenciar.
- Implementação do aplicativo simplificada, já que a grade está utilizando o carregador de classe do aplicativo cliente.
- Particionamento de suporte e alta disponibilidade.

Desvantagens

- Aumento da área de cobertura da memória no processo do cliente já que todos os dados são colocados no processo.
- Aumento da utilização da CPU para atender pedidos de clientes.
- Mais difícil para manipular atualizações de aplicativo, pois os clientes estão usando os mesmos arquivos archive de Java do aplicativo que os servidores.
- Menos flexível. A escala dos clientes e servidores de grade não pode aumentar na mesma proporção. Quando os servidores são externamente definidos, é possível ter mais flexibilidade no gerenciamento do número de processos.

Quando Utilizar

Utilize grades integradas quando há grande quantidade de memória livre no processo do cliente para dados da grade e dados de failover potenciais.

Para obter mais informações, consulte o tópico sobre a ativação do mecanismo de invalidação do cliente no *Guia de Administração*.

Topologias de Replicação de Grade Multimestre (AP)

Usando o recurso de replicação assíncrona multimestre, duas ou mais grades podem se tornar espelhos exatos umas das outras. Este espelhamento é executado

usando a replicação assíncrona entre links conectando as grades. Cada grade é hospedada dentro de um "domínio" completamente independente, que possui seu próprio serviço de catálogo, servidores de contêiner e um nome de domínio exclusivo. Com o recurso de replicação assíncrona multimestre, É possível usar links para interconectar uma coleta desses domínios e, em seguida, sincronizar os domínios, usando a replicação sobre os links. O eXtreme Scale permite construir quase qualquer topologia, pois a definição dos links entre domínios é deixada para você.

7.1+ A replicação de grade multimestre é um novo recurso significativo na Versão 7.1.

Domínios: Grades com Características Específicas

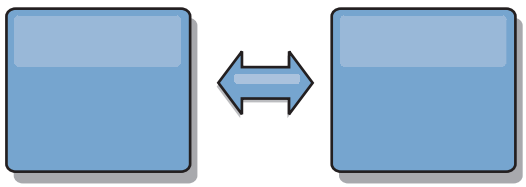
Grades usadas em topologias de replicação multimestre são conhecidas como *domínios*. Cada domínio deve ter as seguintes características:

- Ter um serviço de catálogo privado com um nome de domínio exclusivo
- Ter o mesmo nome de grade que outras grades no domínio
- Ter o mesmo número de partições que outras grades no domínio
- Ser uma grade FIXED_PARTITION (grades PER_CONTAINER não podem ser replicadas)
- Ter o mesmo número de partições (pode o não ter o mesmo número e tipos de réplicas)
- Ter os mesmos tipos de dados sendo replicados como outras grades no domínio
- Ter o mesmo nome do conjunto de mapas, nomes de mapas e modelos de mapas dinâmicos que outras grades no domínio

Depois que os domínios na topologia tiverem sido iniciados, quaisquer grades com as características precedentes serão replicadas. Observe que sua política de replicação será ignorada.

Links Conectando Domínios

Uma infraestrutura de grade de replicação é um gráfico conectado de domínios com links bidirecionais entre eles. Um link permite que dois domínios troquem mudanças de dados. Por exemplo, a topologia mais simples é um par de domínios com um único link entre eles. Os domínios são denominados começando com “A” e, em seguida, “B” e assim por diante a partir da esquerda. O link pode cruzar uma WAN (Wide Area Network), ampliando grandes distâncias. Se o link for interrompido, as mudanças ainda podem ser feitas nos dados em qualquer um dos domínios. As mudanças são reconciliadas mais tarde, quando o link reconecta os domínios. Os links tentarão automaticamente reconectar se a conexão com a rede for interrompida.

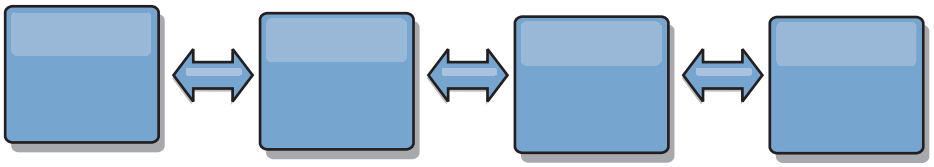


Depois que você configurar os links, o eXtreme Scale primeiro tentará tornar cada domínio idêntico e, em seguida, tentará manter as condições idênticas conforme as mudanças ocorrerem em qualquer domínio. O objetivo do eXtreme Scale é que

cada domínio seja um espelho exato de cada outro domínio conectado pelos links. Os links de replicação entre os domínios ajudam a garantir que as mudanças feitas em um domínio sejam copiadas para os outros domínios.

Topologias em Linha

Embora esteja entre as topologias mais simples, uma topologia em linha demonstra algumas qualidades dos links. Primeiro, não é necessário que um domínio esteja conectado diretamente a outro domínio para receber as mudanças. O domínio B puxará as mudanças do Domínio A. O domínio C recebe mudanças do Domínio A por meio do Domínio B, que conecta os Domínios A e C. De forma semelhante, o Domínio D recebe mudanças dos outros domínios por meio do Domínio C. Esta habilidade envia o carregamento das mudanças de distribuição para longe da origem das mudanças.



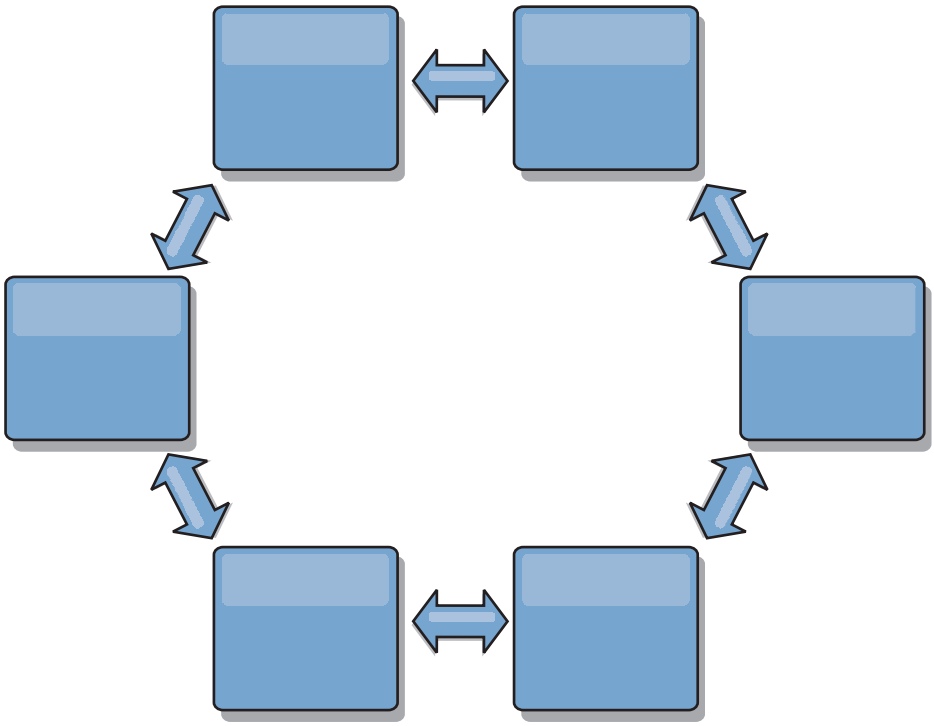
Observe que se o Domínio C falhar, os seguintes eventos poderão ocorrer.

1. O Domínio D pode ficar órfão até o Domínio C ser reiniciado
2. o Domínio C pode se sincronizar com o Domínio B, que é uma cópia do Domínio A
3. O Domínio D pode usar o Domínio C para se sincronizar com as mudanças nos Domínios A e B ocorridas enquanto o Domínio D estava órfão (enquanto o Domínio C estava inativo)

No final, os Domínios A, B, C e D podem se tornar, todos, idênticos uma aos outros novamente.

Topologias em Anel

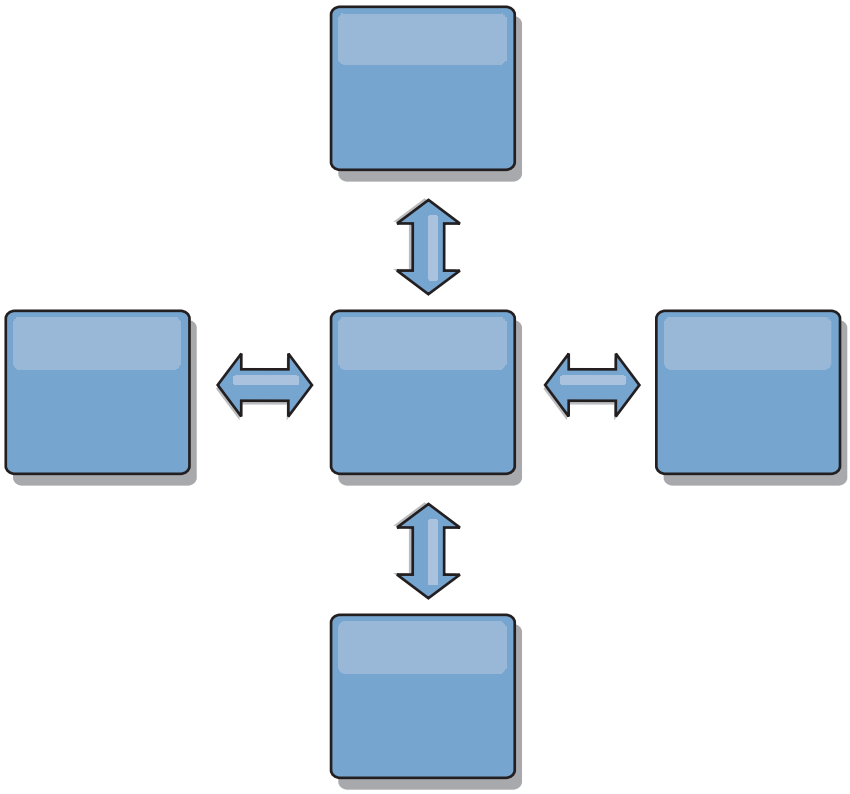
As topologias em anel são um exemplo de uma topologia mais resiliente. Embora um domínio ou um único link possa falhar, os domínios sobreviventes ainda podem obter mudanças viajando ao redor do anel, longe da falha. Cada domínio tem dois links para os outros domínios. Cada domínio tem no máximo dois links, não importa o tamanho de uma topologia em anel. A latência para propagar mudanças pode ser grande, pois as mudanças de um domínio particular podem precisar viajar por meio de vários domínios antes que todos os domínios vejam todas as mudanças. Uma topologia em linha tem o mesmo problema.



Descreva uma topologia em anel mais sofisticada, com um domínio-raiz no centro do anel. O domínio-raiz age como um centro de roteamento central, enquanto os outros domínios agem como centros de roteamento remotos para as mudanças que ocorrem no domínio-raiz. O domínio-raiz pode arbitrar mudanças entre os domínios. Se uma topologia em anel contiver mais de um anel ao redor de um domínio-raiz, o domínio-raiz poderá arbitrar mudanças apenas entre os domínios no anel mais interno. No entanto, os resultados da arbitragem se espalham para os domínios nos outros anéis.

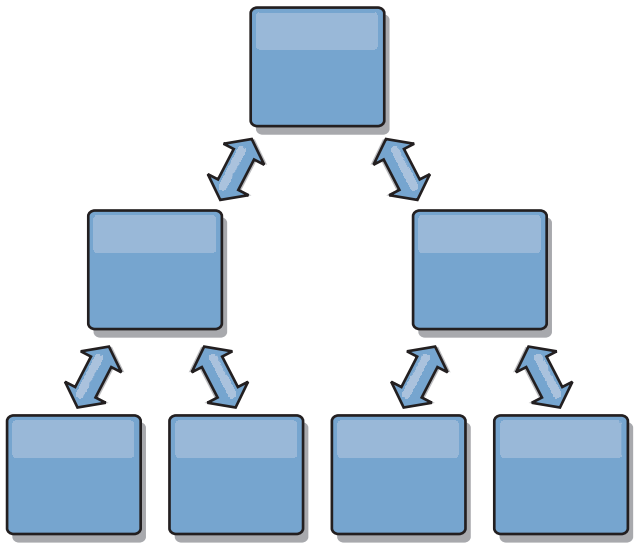
Topologias Hub e Spoke

Uma topologia hub e spoke tem melhor latência, o que significa que as mudanças viajam no máximo por meio de um domínio intermediário (o hub), mas apresenta alguns outros problemas. Ela tem um domínio central que age como um hub. Este "domínio hub" é conectado a cada "domínio spoke" usando um link. Claramente, o encargo de distribuir mudanças entre os domínios fica com o hub. O hub age como um centro de roteamento para colisões, uma configuração que pode ser importante para determinados cenários. Em um ambiente com uma taxa de atualização alta, talvez seja necessário executar o hub em mais hardware que o spoke, para que ele possa continuar ativo. O eXtreme Scale é projetado para escalar linearmente, o que significa que é possível aumentar o hub, conforme necessário, sem dificuldade. No entanto, se o hub falhar, as mudança não serão distribuídas até ele ser reiniciado. As mudanças nos domínios spoke serão distribuídas depois que o hub for reconectado.



Topologias em Árvore

Um último exemplo de topologia é uma árvore direcionada acíclica. Acíclica significa que não há ciclos ou loops. Direcionada significa que existem links apenas entre pais e filhos. Esta configuração pode ser útil para topologias como muitos domínios que não é prático ter um hub central conectado a cada spoke possível ou no qual é necessário ter a habilidade de incluir domínios-filhos sem atualizar o domínio-raiz.



Esta topologia ainda pode ter um centro de roteamento central no domínio-raiz, mas o segundo nível pode agir como centros de roteamento remotos para as mudanças que ocorrem no domínio abaixo deles. O domínio-raiz pode arbitrar as

mudanças entre os domínios no segundo nível apenas. Árvores N-ary também são possíveis. Uma árvore N-ary tem N filhos em cada nível. Cada domínio tem N espalhados.

Considerações sobre Arbitragem no Design de Topologia

Poderão ocorrer colisões de mudanças se os mesmos registros puderem ser alterados simultaneamente em dois locais. Configure cada domínio para ter quase a mesma quantia de CPU, memória, recursos de rede. É necessário observar que os domínios que executam a manipulação da colisão de mudanças (arbitragem) usam mais recursos que outros domínios. As colisões são detectadas automaticamente. Elas são resolvidas usando um entre dois mecanismos:

- **Árbitro de colisão padrão.** O protocolo padrão é usar as mudanças do domínio lexicalmente mais baixo denominado. Por exemplo, se os domínios A e B gerarem um conflito para um registro, a mudança do domínio B será ignorada. O domínio A mantém sua versão e o registro no domínio B é alterado para que corresponda ao registro do domínio A. Isso também se aplica a aplicativos nos quais os usuários ou as sessões são normalmente ligados ou têm afinidade com uma das grades.
- **Árbitro de colisão customizado.** Os aplicativos podem fornecer um árbitro customizado. Quando um domínio detecta uma colisão, ele chama o árbitro. Para obter informações sobre como desenvolver um 'bom' árbitro customizado, consulte Desenvolvendo árbitros customizados para replicação multimestre.

Para topologias nas quais as colisões são possíveis, considere a possibilidade de usar uma topologia hub e spoke ou uma topologia em árvore. Essas duas topologias são úteis para evitar colisões sem fim, que podem acontecer quando:

1. Vários domínios experimentam uma colisão
2. Cada domínio resolve a colisão localmente, produzindo revisões
3. As revisões colidem, resultando em revisões de revisões
4. E assim por diante, uma vez que as revisões são propagadas entre os vários domínios tentando atingir sincronicidade

Para evitar colisões sem fim, escolha um domínio específico – um *domínio de arbitragem* – como o manipulador de colisão para um subconjunto de domínios. Por exemplo, uma topologia hub e spoke pode usar o hub como o manipulador de colisão. O manipulador de colisão spoke ignora quaisquer colisões detectadas pelos domínios spoke. O domínio hub criará revisões, evitando revisões de colisão fora de controle. O domínio designado para tratar colisões deve ser vinculado a todos os domínios para os quais ele é responsável por resolver colisões. Em uma topologia em árvore, os domínios pais internos resolvem colisões para seus filhos imediatos. Em contraste, se usar uma topologia em anel, não será possível designar um domínio no anel como o resolvedor.

A tabela a seguir resume as abordagens de arbitragem que são mais compatíveis com várias topologias.

Tabela 5. Abordagens de Arbitragem. Esta tabela define se a arbitragem de aplicativo é compatível com várias tecnologias.

Topologia	Arbitragem de aplicativo?	Notas
Uma linha de dois domínios	Sim	Escolha um domínio como o árbitro.

Tabela 5. Abordagens de Arbitragem (continuação). Esta tabela define se a arbitragem de aplicativo é compatível com várias tecnologias.

Topologia	Arbitragem de aplicativo?	Notas
Uma linha de três domínios	Sim	O domínio do meio deve ser o árbitro. Pense no domínio do meio como o hub em uma topologia hub e spoke simples.
Uma linha de mais de três domínios	Não	A arbitragem de aplicativo não é suportada.
Um hub com N spokes	Sim	Hub com links para todos os spokes deve ser o domínio de arbitragem.
Um anel de N domínios	Não	A arbitragem de aplicativo não é suportada.
Uma árvore acíclica, direcionada (árvore N-ary)	Sim	Todos os nós-raiz devem arbitrar seus descendentes diretos apenas.

Considerações sobre Links no Design de Topologia

De forma ideal, uma topologia inclui um número mínimo de links enquanto otimiza trade-offs entre latência de mudança, tolerância a falhas e características de desempenho.

- **Latência de mudança**

  A latência de mudança é determinada pelo número de domínios intermediários pelos quais uma mudança deve passar antes de chegar a um domínio específico.

  Uma topologia tem a melhor latência de mudança quando elimina domínios intermediários vinculando cada domínio a outro domínio. No entanto, um domínio deve executar o trabalho de replicação em proporção ao seu número de links. Para topologias grandes, o número completo de links a ser definido pode representar uma carga administrativa.

  A velocidade com que uma mudança é copiada para outros domínios depende de fatores adicionais, como:

  – CPU e largura de banda da rede no domínio de origem
  – O número de domínios e links intermediários entre o domínio de origem e o de destino
  – Os recursos de CPU e de rede disponíveis para os domínios de origem, destino e intermediários

- **Tolerância a Falhas**

  A tolerância a falhas é determinada por quantos caminhos existem entre dois domínios para a replicação de mudança.

  Se existir apenas um único link entre os domínios, se o link falhar, as mudanças não serão propagadas. Se o único link de um domínio para outro domínio passar por domínios intermediários, as mudanças não serão propagadas se qualquer um dos domínios intermediários estiver inativo.

  Considere a topologia em linha com três domínios A, B e C:

  ```
  A <-> B <-> C
  ```

  Se qualquer uma dessas condições for mantida, o Domínio C não verá nenhuma mudança de A:

  – O Domínio A está ativo e o domínio B está inativo
  – O link entre A e B está inativo

– O link entre B e C está inativo

Em contraste, uma topologia em anel permite que cada domínio pegue mudanças de qualquer direção.

```
A <-> B <-> C <-> de volta para A
```

Por exemplo, se o domínio B estiver inativo, o domínio C ainda pode pegar mudanças diretamente do domínio A.

Um design hub e spoke é suscetível a que o hub fique inativo, uma vez que todas as mudanças são pegas por meio do hub. No entanto, vale lembrar que um único domínio ainda é uma grade totalmente tolerante a falhas que pode ter falhas grosseiras como problemas na WAN ou no datacenter físico.

- **Desempenho**

  O número de links definido em um domínio afeta o desempenho. Mais links usam mais recursos e o desempenho de replicação pode diminuir como resultado. A habilidade de pegar mudanças para um domínio A por meio de outros domínios efetivamente libera o domínio A de replicar suas transações em todo lugar. *O carregamento de distribuição de mudança em um domínio é limitado ao número de links que ele usa. Não tem nenhuma ligação com quantos domínios estão na topologia.* Esta propriedade fornece escalabilidade e permite que a carga de distribuição de mudança seja compartilhada pelos domínios na topologia, em vez de colocar a carga em um único domínio.

  Um domínio pode pegar mudanças indiretamente por meio de outros domínios. Considere uma topologia em linha com cinco domínios.

  ```
  A <=> B <=> C <=> D <=> E
  ```

  – A pega mudanças de B, C, D e E por meio de B

  – B pega mudanças de A e C diretamente e mudanças de D e E por meio de C

  – C pega mudanças de B e D diretamente e mudanças de A por meio de B e E por meio de D

  – D pega mudanças de C e E diretamente e mudanças de A e B por meio de C

  – E pega mudanças de D diretamente e mudanças de A, B e C por meio de D

  O carregamento de distribuição nos domínios A e E é o mais baixo, porque eles têm, cada um, um link apenas para um único domínio. O carregamento de distribuição nos domínios B, C e D é o dobro do carregamento nos domínios A e E porque os domínios B, C e D têm, cada um, um link para dois domínios. Essa distribuição de carregamentos poderia permanecer constante, mesmo que a linha contivesse 1000 domínios, pois o carregamento depende do número de links de cada domínio, não do número geral de domínios na topologia.

Considerações sobre Desempenho

Leve em consideração as seguintes limitações quando usar as topologias de replicação multimestre.

- **Ajuste de distribuição de mudança** (discutido anteriormente)
- **Latência de replicação** (discutido anteriormente)
- **Desempenho do link de replicação** O eXtreme Scale cria um único soquete TCP/IP entre qualquer par de JVMs. Todo tráfego entre essas JVMs ocorre por meio do soquete, incluindo a replicação multimestre. Como os domínios são hospedados em pelo menos N JVMs de contêiner, fornecendo pelo menos N links TCP para os domínios do mesmo nível, os domínios com números maiores de contêineres terão níveis mais altos de desempenho de replicação. Mais contêineres significa mais recursos de CPU e de rede.

- **Ajuste da janela de deslizamento TCP e RFC 1323** A ativação do suporte RFC 1323 em ambos os terminais de um link permite mais dados para um roundtrip, resultando em um maior rendimento. A técnica expande a capacidade da janela em um fator de aproximadamente 16.000.

  A rechamada desses soquetes TCP usa um mecanismo de janela de deslizamento para controlar o fluxo de dados em massa, o que geralmente limita o soquete a 64 KB para um intervalo de roundtrip. Se o intervalo de roundtrip for de 100 ms, a largura de banda será limitada a 640 KB/segundo sem ajuste adicional. Usar totalmente a largura de banda disponível em um link pode exigir um ajuste específico para um sistema operacional. A maioria dos sistemas operacionais inclui parâmetros de ajuste, incluindo opções RFC 1323, para aprimorar o rendimento por meio dos links de alta latência.

  Vários fatores podem afetar o desempenho de replicação:

  – A velocidade com que o eXtreme Scale pode pegar mudanças.
  – A velocidade com que o eXtreme Scale pode atender pedidos de replicação de tomada.
  – A capacidade da janela de deslizamento.
  – Ajuste do buffer de rede em ambos os lados de um link para permitir que o eXtreme Scale pegue mudanças por meio do soquete o mais rápido possível.
- **Serialização de Objeto** Todos os dados devem ser serializáveis. Se um domínio não estiver usando COPY_TO_BYTES, o domínio deverá usar a serialização Java ou ObjectTransformers para otimizar o desempenho da serialização.
- **Compactação** O eXtreme Scale compacta todos os dados enviados entre os domínios por padrão. Não há nenhuma opção para desativar a compactação no release atual.
- **Ajuste de memória** *O uso de memória para uma topologia de replicação multimestre é independente do número de domínios na topologia.*

  A ativação da replicação multimestre inclui uma sobrecarga fixa por Entrada de mapa para tratar a versão. Cada contêiner também rastreia uma quantia fixa de dados para cada domínio na topologia. Uma topologia com dois domínios usa aproximadamente a mesma memória que uma topologia com 50 domínios. O eXtreme Scale não usa logs de reprodução ou filas semelhantes em sua implementação, o que significa que se um link de replicação não estiver disponível por um período substancial de tempo, não haverá nenhuma estrutura de dados aumentando de tamanho, aguardando para retomar a replicação quando o link for reiniciado.

Vários Datacenters com FIXED_PARTITION

Agora, é possível usar uma grade FIXED_PARTITION entre dois ou mais datacenters. Cada datacenter precisa de seu próprio domínio, em termos de replicação multimestre. Cada datacenter pode ler e gravar dados em relação ao domínio local. Essas mudanças serão propagadas para os outros datacenters usando os links que você definiu.

Clientes Totalmente Replicados

Essa variação de topologia envolve um par de servidores do eXtreme Scale sendo executados como um hub. Cada cliente cria uma grade autocontida de contêiner único com um catálogo na JVM do cliente. Um cliente usa sua grade para conectar ao catálogo do hub, fazendo com que o cliente sincronize com o hub assim que obtiver uma conexão com o hub.

Todas as mudanças feitas pelo cliente são locais para o cliente e são replicadas de forma assíncrona para o hub. O hub age como um domínio de arbitragem, distribuindo mudanças para todos os clientes conectados. A topologia de clientes totalmente replicada fornece um bom cache L2 para um mapeador relacional de objeto, como OpenJPA. As mudanças serão distribuídas rapidamente entre JVMs de cliente por meio do hub. Contanto que o tamanho do cache possa estar contido no espaço de heap disponível dos clientes, esta topologia será uma boa arquitetura para este estilo de L2.

Use várias partições para escalar o domínio do hub em várias JVMs, se necessário. Como todos os dados ainda devem se ajustar em uma única JVM de cliente, usar várias partições aumenta a capacidade do hub para distribuir e arbitrar mudanças, mas não altera a capacidade de um único domínio.

Limitações

Leve em consideração as seguintes limitações ao decidir se e como usar as topologias de replicação multimestre.

- **Tenha cuidado com a configuração de carregadores de classe com vários domínios**

  Os domínios devem ter acesso a todas as classes que são usadas como chaves e valores. Todas as dependências devem ser refletidas em todos os caminhos da classe para as JVMs do contêiner da grade para todos os domínios. Se um plug-in CollisionArbiter recuperar o valor para uma entrada de cache, as classes para os valores deverão estar presentes para o domínio que está chamando o árbitro.

- **O uso de carregadores não é recomendado**

  Os carregadores podem ser usados para mudanças de interface entre uma grade e um banco de dados. É improvável que todas as grades (domínios) em uma topologia sejam colocadas geograficamente com o mesmo banco de dados. A latência de WAN e outros fatores podem renderizar este caso de uso não desejável.

  O pré-carregamento da grade é outro problema que requer um design cuidadoso. Geralmente, quando uma grade é reiniciada, ela é pré-carregada novamente. O pré-carregamento não é necessário ou mesmo desejável quando estiver usando a replicação multimestre. Assim que um domínio estiver on-line, ele se recarregará automaticamente com o conteúdo dos domínios aos quais está vinculado. Como resultado, não há necessidade de iniciar um pré-carregamento manual para uma grade que é um domínio em uma topologia de replicação multimestre.

  Os carregadores, geralmente, obedecem as regras de inserção e atualização. Com a replicação multimestre, as inserções devem ser tratadas como mesclagens. Quando os dados estiverem sendo pegos remotamente após o reinício de um domínio, os dados existentes serão ‘inseridos' no domínio local. Como esses dados podem já ter estado no banco de dados local, uma inserção típica falhará com uma exceção de chave duplicada no banco de dados. Em vez disso, a semântica de mesclagem deve ser usada.

  O eXtreme Scale pode ser configurado para fazer um pré-carregamento baseado em shard, usando os métodos de pré-carregamento nos plug-ins do Carregador. Não use esta técnica em uma topologia de replicação multimestre. Em vez disso, use um pré-carregamento baseado no cliente quando a topologia for iniciada (inicialmente). Permita que a topologia multimestre atualize os domínios reiniciados com uma cópia atual do que está armazenado em outros domínios

na topologia. Depois que os domínios tiverem sido iniciados, a topologia multimestre será responsável por mantê-los em sincronia.

- **EntityManager não é suportado**

  Um conjunto de mapas contendo um mapa de entidade não é replicado por meio dos domínios.

- **Mapas de matriz de byte não são suportados**

  Um conjunto de mapas contendo um mapa configurado com COPY_TO_BYTES não é replicado por meio dos domínios.

- **Write-behind não é suportado**

  Um conjunto de mapas contendo um mapa configurado com o suporte write-behind não é replicado por meio dos domínios.

Configurações de Implementação para o eXtreme Scale

O WebSphere eXtreme Scale pode ser implementado em dois tipos de ambientes: local ou distribuído. A configuração necessária depende do tipo do ambiente de implementação.

Ambientes Locais

Ao implementar em um ambiente local, o eXtreme Scale é executado em uma única Java Virtual Machine e não é replicado. Um ambiente local requer um arquivo XML do ObjectGrid ou APIs do ObjectGrid.

Ambientes Distribuídos

Ao implementar em um ambiente distribuído, o eXtreme Scale é executado em um conjunto de Java Virtual Machines. Com essa configuração, poderá utilizar a replicação de dados e criação de partições. Um ambiente distribuído pode ser ainda classificado como uma topologia de implementação dinâmica na qual é possível incluir servidores, conforme necessário. Assim como ocorre com um ambiente local, um arquivo XML do ObjectGrid, ou uma configuração programática equivalente, é necessário em um ambiente distribuído. Além disso, é necessário fornecer um arquivo XML de política de implementação com detalhes de configuração para sua grade de dados de memória do eXtreme Scale.

Serviço de Catálogo de Alta Disponibilidade

Um domínio do serviço de catálogo é a grade de servidores de catálogos que você está usando, que retém informações de topologia para todos os contêineres no ambiente do eXtreme Scale. O serviço de catálogo controla o equilíbrio e o roteamento de todos os clientes. Para implementar o eXtreme Scale como um espaço de processamento de banco de dados em memória, você deve armazenar em cluster o serviço de catálogo em um domínio do serviço de catálogo para alta disponibilidade.

Componentes do Domínio do Serviço de Catálogo

Quando múltiplos servidores de catálogo iniciam, um dos servidores é eleito como o servidor de catálogo principal que aceita pulsações do Internet Inter-ORB Protocol (IIOP) e manipula as alterações dos dados do sistema em resposta a qualquer serviço de catálogo ou as alterações de contêiner.

Quando os clientes entram em contato com qualquer um dos servidores de catálogos, a tabela de roteamento para o domínio do serviço de catálogo é propagada para os clientes por meio do contexto de serviço de Common Object Request Broker Architecture (CORBA).

Configure pelo menos três servidores de catálogos. Se a sua configuração tem zonas, é possível configurar um servidor de catálogos por zona.

Quando um servidor e contêiner do eXtreme Scale entram em contato com um dos servidores de catálogos, a tabela de roteamento para o domínio do serviço de catálogo também é propagada para o servidor e contêiner do eXtreme Scale por meio do contexto de serviço de CORBA. Além disso, se o servidor de catálogos contactado atualmente não for o servidor de catálogos principal, o pedido é automaticamente roteado para o servidor de catálogos principal atual e a tabela de roteamento para o servidor de catálogos é atualizada.

Nota: Uma grade do servidor de catálogos e a grade do servidor de contêineres são muito diferentes. O domínio do serviço de catálogo é para alta disponibilidade dos dados do sistema. A grade do contêiner é para a alta disponibilidade de dados, escalabilidade e gerenciamento de carga de trabalho. Portanto, existem duas tabelas de roteamento diferentes: a tabela de roteamento para o domínio do serviço de catálogo e a tabela de roteamento para os shards da grade do servidor.

As responsabilidades do catálogo são divididas em uma série de serviços. O gerenciador do grupo principal executa agrupamento peer para monitoramento de funcionamento, o serviço de posicionamento executa alocação, o serviço de administração fornece acesso à administração e o serviço de local gerencia a localidade.

Implementação do Domínio do Serviço de Catálogo

Gerenciador de grupo principal

O serviço de catálogo usa o gerenciador de alta disponibilidade (gerenciador HA) para agrupar processos para o monitoramento de disponibilidade. Cada agrupamento dos processos é um grupo principal. Com o eXtreme Scale, o gerenciador do grupo principal agrupa dinamicamente o processo. Estes processos são mantidos pequenos para permitir a escalabilidade. Cada grupo principal elege um líder que possui a responsabilidade adicional de enviar o status para o gerenciador do grupo principal quando membros individuais falham. O mesmo mecanismo de status é usado para descobrir quando todos os membros de um grupo falham, o que causa a falha da comunicação com o líder.

O gerenciador do grupo principal é um serviço completamente automático responsável pela organização de contêineres em pequenos grupos de servidores que são então automaticamente associados de maneira livre para criar um ObjectGrid. Quando um contêiner contata pela primeira vez o serviço de catálogo, ele aguarda para ser designado a um grupo novo ou existente. Uma implementação do eXtreme Scale consiste em vários desses grupos, e esse agrupamento é um importante ativador de escalabilidade. Cada grupo é um grupo do Java Virtual Machines que usa a pulsação para monitorar a disponibilidade dos outros grupos. Um destes membros do grupo é eleito o líder e possui uma responsabilidade adicional de retransmitir informações de disponibilidade para o serviço de catálogo para permitir reação por meio de realocação e encaminhamento de rotas.

Serviço de disposição

O serviço de catálogo gerencia o posicionamento de shards por todo o conjunto de contêineres disponíveis. O serviço de disposição é responsável pela manutenção de um equilíbrio de recursos entre recursos físicos. O serviço de disposição é responsável pela alocação de shards individuais em seu contêiner de host. O serviço de posicionamento é executado como um serviço eleito Um de N na grade de dados, portanto, exatamente uma instância do serviço está em execução. Se tal instância falhar, outro processo é então eleito e assume. Para redundância, o estado do serviço de catálogo é replicado entre todos os servidores que estão hospedando o serviço de catálogo.

Administração

O serviço de catálogo também é o ponto de entrada lógico para administração do sistema. O serviço de catálogo hospeda um Managed Bean (MBean) e fornece as URLs do Java Management Extensions (JMX) para qualquer um dos servidores que o serviço está gerenciando.

Serviço de local

O serviço de local atua como o ponto de contato para ambos os clientes que estão procurando contêineres que hospedam o aplicativo que procuram, bem como os contêineres que estão registrando aplicativos hospedados com o serviço de disposição. O serviço de local é executado em todos os membros da grade para efetuar o scale out desta função.

O serviço de catálogo hospeda lógica que é tipicamente inativa durante estados estáveis. Como resultado, o serviço de catálogo influencia a escalabilidade minimamente. O serviço é baseado em centenas de serviços de contêineres que se tornam disponíveis simultaneamente. Para disponibilidade, configure o serviço de catálogo numa grade.

Planejamento

Depois que um domínio do serviço de catálogo é iniciado, os membros da grade são ligados juntos. Planeje com cuidado sua topologia de domínio do serviço de catálogo, pois não é possível modificar a configuração do domínio do serviço de catálogo no tempo de execução. Disperse a grade o mais diversamente possível para evitar erros.

Iniciando um domínio do serviço de catálogo

Para obter mais informações sobre como criar um domínio do serviço de catálogo, consulte .

Conectando contêineres do eXtreme Scale integrados no WebSphere Application Server a um domínio do serviço de catálogo independente

É possível configurar contêineres do eXtreme Scale integrados em um ambiente do WebSphere Application Server para conectar a um domínio do serviço de catálogo independente.

- **7.1+** É possível criar domínios do serviço de catálogo no console administrativo. Consulte .

-  (reprovado) Em releases anteriores, você conectou os serviços de catálogo a um domínio do serviço de catálogo criando uma propriedade customizada. Esta propriedade ainda pode ser usada, mas é reprovada. Para obter mais informações sobre esta propriedade customizada, consulte as informações sobre como iniciar o processo do serviço de catálogo em um WebSphere Application Server no *Guia de Administração*..

  Nota: Conflito de nome do servidor: Como esta propriedade é usada para iniciar o servidor de catálogos do eXtreme Scale, assim como para se conectar a ele, os servidores de catálogo não devem ter o mesmo nome de nenhum servidor do WebSphere Application Server.

Consulte as informações sobre os quoruns do servidor de catálogos no *Visão Geral do Produto* para obter mais informações.

Quorums de Servidores de Catálogo

Quorum é o número mínimo de servidores de catálogo que são necessários para conduzir operações de posicionamento para a grade de dados. O número mínimo é o conjunto completo de servidores de catálogo que você substitui o valor do quorum.

Termos Importantes

A seguir, é apresentada uma lista de termos relacionados a considerações do quorum para o WebSphere eXtreme Scale.

- **Brown out**: Um brown out é a perda temporária de conectividade entre um ou mais servidores.
- **Black out**: Um black out é a perda permanente de conectividade entre um ou mais servidores.
- **Datacenter**: Um datacenter é um grupo de servidores localizado geograficamente geralmente conectado a uma rede local (LAN).
- **Zona**: Uma zona é uma opção de configuração usada para agrupar servidores que compartilham alguma característica física. A seguir estão alguns exemplos de zonas para um grupo de servidores: um datacenter, conforme descrito no marcador anterior, uma rede local, um edifício, o andar de um edifício, entre outras.
- **Pulsação**: Pulsação é um mecanismo usado para determinar se uma determinada Java virtual machine (JVM) está ou não em execução.

Topologia

Esta seção explica como o WebSphere eXtreme Scale opera por meio de uma rede que inclui componentes não confiáveis. Os exemplos dessa rede incluem uma rede que se estende por vários datacenters.

Espaço de endereço IP

O WebSphere eXtreme Scale requer uma rede em que qualquer elemento endereçável na rede possa conectar a qualquer outro elemento endereçável na rede não impedida. Isso significa que o WebSphere eXtreme Scale requer um espaço de nomenclatura de endereço IP simples e requer que todos os firewalls permitam que todo o tráfego flua entre os endereços IP e portas usados pelas Java virtual machines (JVM) que hospedam elementos do WebSphere eXtreme Scale.

LANs Conectadas

Cada LAN recebe um identificador de zona para os requisitos do WebSphere eXtreme Scale. O WebSphere eXtreme Scale faz pulsar agressivamente as JVMs em uma única zona. Uma pulsação ausente resultará em um evento de failover se o serviço de catálogo tiver quorum.

Domínio do serviço de catálogo e servidores de contêiner

Um domínio do serviço de catálogo é uma coleta de JVMs semelhantes. Um domínio do serviço de catálogo é uma grade composta de servidores de catálogos e é fixa no tamanho. Entretanto, o número de servidores de contêiner é dinâmico. Os servidores de contêiner podem ser incluídos e removidos sob demanda. Em uma configuração de três datacenters, o WebSphere eXtreme Scale requer uma JVM do serviço de catálogo por datacenter.

O domínio do serviço de catálogo usa um mecanismo de quorum completo. Por causa deste mecanismo de quorum completo, todos os membros da grade de dados deve concordar em qualquer ação.

As JVMs do servidor de contêiner são identificadas com um identificador de zona. A grade das JVMs de contêiner é dividida automaticamente em pequenos grupos principais de JVMs. Um grupo principal inclui apenas JVMs da mesma zona. JVMs de zonas diferentes nunca estão no mesmo grupo principal.

Um grupo principal tenta agressivamente detectar a falha das JVMs de seus membros. As JVMs de contêiner de um grupo principal nunca devem se estender por várias LANs conectadas com links, como em uma ampla rede local. Isso significa que um grupo de núcleo não pode ter contêineres na mesma zona em execução em datacenters diferentes.

Ciclo de Vida do Servidor

Inicialização do servidor de catálogo

Os servidores de catálogos são iniciados usando o comando startOgServer. O mecanismo de quorum é desativado por padrão. Para ativar o quorum, passe o sinalizador ativado `-quorum` no comando startOgServer ou inclua a propriedade `enableQuorum=true` no arquivo de propriedade. Todos os servidores de catálogo devem receber a mesma configuração de quorum.

```
# bin/startOgServer cat0 –serverProps objectGridServer.properties
```

```
objectGridServer.properties file
catalogClusterEndPoints=cat0:cat0.domain.com:6600:6601,
cat1:cat1.domain.com:6600:6601
catalogServiceEndPoints= cat0.domain.com:2809, cat1.domain.com:2809
enableQuorum=true
```

Inicialização do servidor de contêiner

Os servidores de contêiner são iniciados usando o comando startOgServer. Ao executar uma grade de dados por meio de datacenters, os servidores devem usar a marcação de zona para identificar o datacenter no qual residem. Configurar a zona nos servidores da grade permite que o WebSphere eXtreme Scale monitore o funcionamento dos servidores com escopo no datacenter, minimizando o tráfego pelo datacenter.

```
# bin/startOgServer gridA0 –serverProps objectGridServer.properties –
objectgridfile xml/objectgrid.xml –deploymentpolicyfile xml/
deploymentpolicy.xml
```

objectGridServer.properties file

```
catalogServiceEndPoints= cat0.domain.com:2809, cat1.domain.com:2809
zoneName=ZoneA
```

Encerramento do servidor de grade

Os servidores de grade são parados usando o comando stopOgServer. Ao encerrar um datacenter inteiro para manutenção, transmita a lista de todos os servidores que pertencerem a essa zona. Isso permitirá que a zona mude normalmente do estado teardown para uma ou mais zonas no estado surviving.

```
# bin/stopOgServer gridA0,gridA1,gridA2 –catalogServiceEndPoints
cat0.domain.com:2809,cat1.domain.com:2809
```

Detecção de Falha

O WebSphere eXtreme Scale detecta o fim do processo por meio de eventos anormais de encerramento do soquete. O serviço de catálogo será informado imediatamente quando um processo se encerrará. Um blecaute é detectado por meio de pulsações perdidas. O WebSphere eXtreme Scale se protege contra condições de brown out por meio dos datacenters usando uma implementação de quorum.

Implementação de Pulsação

Esta seção descreve como a verificação de ativação é implementada no WebSphere eXtreme Scale.

Pulsação de membro de grupo principal

O serviço de catálogo coloca as JVMs de contêiner em grupos principais de tamanho limitado. Um grupo principal tentará detectar a falha de seus membros usando dois métodos. Se um soquete de JVM for fechado, essa JVM será considerada inativa. Cada membro também efetua pulsação sobre esses soquetes a uma taxa determinada pela configuração. Se uma JVM não responder a essas pulsações dentro de um período máximo de tempo especificado, a JVM será considerada inativa.

Um único membro de um grupo principal é sempre escolhido para ser o líder. O core group leader (CGL) é responsável por informar periodicamente ao serviço de catálogo que o grupo principal está ativo e relatar quaisquer alterações de associação no serviço de catálogo. Uma mudança de associação pode ser uma JVM com falha ou uma JVM recém incluída que se une ao grupo principal.

Se o líder do grupo principal não puder entrar em contato com nenhum membro do domínio do serviço de catálogo, ele continuará tentando novamente.

Pulsação do domínio do serviço de catálogo

O domínio do serviço de catálogo é semelhante a um grupo principal privado com uma associação estática e um mecanismo de quorum. Ele detecta falhas da mesma

forma que um grupo principal normal. Porém, o comportamento é modificado para incluir a lógica de quorum. O serviço de catálogo também usa uma configuração de pulsação menos rigorosa.

Pulsação grupo principal

O serviço de catálogo precisa saber quando os servidores de contêineres falharão. Cada grupo principal é responsável por determinar a falha da JVM de contêiner e relatar isso para o serviço de catálogo por meio do core group leader. Também é possível uma falha completa de todos os membros de um grupo principal. Se o grupo principal inteiro falhar, o serviço de catálogo é responsável por detectar essa perda.

Se o serviço de catálogo marcar uma JVM de contêiner como com falha e o contêiner for posteriormente relatado como ativo, será solicitado que a JVM de contêiner encerre os servidores de contêiner do WebSphere eXtreme Scale. Uma JVM neste estado não é visível nas consultas do comando xsadmin. As mensagens nos logs da JVM de contêiner indicam que a JVM de contêiner falhou. É necessário reiniciar manualmente estas JVMs.

Se ocorrer um evento de perda de quorum, a pulsação será suspensa até o quorum ser reestabelecido.

Comportamento do Quorum de Serviço de Catálogo

Normalmente, os membros do serviço de catálogo possuem conectividade completa. O domínio do serviço de catálogo é um conjunto estático de JVMs. O WebSphere eXtreme Scale espera que todos os membros do serviço de catálogo estejam sempre on-line. O serviço de catálogo responde a eventos do contêiner apenas enquanto o serviço de catálogo tem quorum.

Se o serviço de catálogo perde quorum, ele espera até que o quorum seja restabelecido. Durante o período de tempo no qual o serviço de catálogo não tem quorum, ele ignora eventos dos servidores de contêiner. Os servidores de contêiner repetem os pedidos rejeitados pelo servidor de catálogo durante esse tempo uma vez que o WebSphere eXtreme Scale espera o quorum ser restabelecido.

A seguinte mensagem indica que o quorum foi perdido. Procure por essa mensagem nos logs de serviço de catálogo.

```
CWOBJ1254W: O serviço de catálogo está aguardando pelo quorum.
```

O WebSphere eXtreme Scale espera perder o quorum pelos seguintes motivos:

- Falha do membro da JVM de serviço de catálogo
- Queda de energia da rede
- perda do datacenter

Parar uma instância do servidor de catálogos usando`stopOgServer` não causa perda de quorum porque o sistema sabe que a instância do servidor parou, o que é diferente de uma falha de JVM ou queda de energia.

Perda de quorum a partir de uma falha de JVM

Uma falha do servidor de catálogos causará perda de quorum. Nesse caso, o quorum deverá ser substituído o mais rápido possível. O serviço de catálogo com falha não pode unir novamente a grade até o quorum ser substituído.

Perda de quorum a partir de uma queda de energia de rede

O WebSphere eXtreme Scale é projetado para esperar a possibilidade de brown outs. Uma queda de energia é quando ocorre uma perda temporária de conectividade entre os datacenters. Geralmente essa perda é temporária por natureza e a conectividade logo é restabelecida dentro de alguns segundos ou minutos. Enquanto o WebSphere eXtreme Scale tenta manter a operação normal durante o período de brown out, um brown out é considerado um evento de falha única. Espera-se que a falha seja corrigida e, em seguida, a operação normal é retomada sem que nenhuma ação do WebSphere eXtreme Scale seja necessária.

Uma queda de energia de longa duração pode ser classificada como um blacaute apenas sob intervenção do usuário. Substituir o quorum em uma queda de energia é necessário para que o evento seja classificado como um blecaute.

Ciclo JVM de serviço de catálogo

Se um servidor de catálogos for parado usando o comando stopOgServer, o quorum diminuirá um servidor a menos. Isso significa que os servidores restantes ainda possuem quorum. Reiniciar o servidor de catálogos retornará o quorum imediatamente para o número anterior.

Consequências de perda de quorum

Se uma JVM de contêiner tiver que falhar durante a perda de um quorum, a recuperação não ocorrerá até voltar a energia ou, no caso de um blecaute, o cliente executa um comando de substituição de quorum. WebSphere eXtreme Scale considera um evento de perda do quorum e uma falha do contêiner como uma falha dupla, que é um evento raro. Isso significa que os aplicativos podem perder o acesso de gravação dos dados que foram armazenados na JVM com falha até o quorum ser restaurado no momento em que a recuperação normal ocorrer.

Da mesma forma, se você tentar iniciar um contêiner durante um evento de perda de quorum, o contêiner não iniciará.

A conectividade completa do cliente é permitida durante a perda de quorum. Se não ocorrer nenhuma falha de contêiner ou problemas de conectividade durante o evento de perda de quorum, os clientes ainda poderão interagir completamente com os servidores de contêineres.

Se ocorrer uma queda de energia, alguns clientes poderão não ter acesso às cópias primárias ou réplicas dos dados até voltar a energia.

Novos clientes poderão ser iniciados, já que deve haver uma JVM de serviço de catálogo em cada datacenter, portanto, pelo menos uma JVM de serviço de catálogo pode ser obtida pelo cliente mesmo durante uma queda de energia.

Recuperação de quorum

Se o quorum for perdido por algum motivo, quando o quorum for reestabelecido, um protocolo de recuperação será executado. Quando ocorrer um evento de perda de quorum, toda a verificação de atividade dos grupos principais será suspensa e

os relatórios de falha também serão ignorados. Quando o quorum retornar, o serviço de catálogo executará uma verificação de atividade de todos os grupos principais para determinar imediatamente a associação. Quaisquer shards anteriormente hospedados nas JVMs de contêiner relatados como falha serão recuperados nesse ponto. Se os shards primários forem perdidos, as réplicas sobreviventes serão promovidas para primárias. Se os shards de réplicas foram perdidos, réplicas adicionais serão criadas nos que sobreviveram.

Substituindo Quorum

Isso deve ser usado apenas quando ocorrer uma falha de datacenter. A perda de quorum devido a uma falha da JVM do serviço de catálogo ou a um brownout da rede deve ser recuperada automaticamente depois que a JVM do serviço de catálogo for reiniciada ou que o brownout da rede for resolvido.

Os administradores são os únicos que sabem quando ocorre uma falha do datacenter. O WebSphere eXtreme Scale trata um brown out e um black out de forma semelhante. É necessário informar ao ambiente do eXtreme Scale dessas falhas usando o comando xsadmin para substituir o quorum. Isso informará ao serviço de catálogo a assumir que o quorum foi obtido com a associação atual e que uma recuperação completa ocorrerá. Ao emitir um comando de substituição de quorum, você estará garantindo que as JVMs no datacenter com falha realmente falharam e não serão recuperadas.

A lista a seguir considera alguns cenários para substituição de quorum. Suponha que você possua três servidores de catálogo: A, B e C.

- Queda de energia: Suponha que ocorra uma queda de energia e o servidor C fique isolado temporariamente. O serviço de catálogo perderá quorum e esperará que o brown out seja resolvido e nesse ponto o C une novamente o domínio do serviço de catálogo e o quorum é restabelecido. Seu aplicativo não terá problemas durante esse tempo.
- Falha temporária: Aqui, o servidor C falha e o serviço de catálogo perde o quorum, portanto, o quorum deve ser substituído. Quando o quorum for reestabelecido, o servidor C poderá ser reiniciado. C unirá novamente o domínio do serviço de catálogo quando for reiniciado. O aplicativo não terá problemas durante esse tempo.
- Falha do datacenter: Verifique se o datacenter realmente falhou e se ele foi isolado na rede. Em seguida, emita o comando xsadmin de substituição de quorum. Os dois datacenters sobreviventes executam recuperação completa ao substituir os shards que estavam hospedados no datacenters com falha. O serviço de catálogo agora está em execução com um servidor de quorum completo A e B. Pode ocorrer atrasos ou exceções no aplicativo durante o intervalo entre o início do blecaute e a substituição do quorum. Quando o quorum for substituído, a grade será recuperada e a operação normalizada.
- Recuperação do datacenter: Os datacenters sobreviventes já estão em execução com o quorum substituído. Quando o datacenter que contém o servidor C for reiniciado, todas as JVMs no datacenter deverão ser reiniciadas. Em seguida, C unirá novamente o domínio do serviço de catálogo existente e o quorum será revertido para a situação normal sem nenhuma intervenção do usuário.
- Falha e queda de energia do datacenter: O datacenter que contém o servidor C falha. O quorum é substituído e recuperado nos datacenters restantes. Se ocorrer uma queda de energia entre os servidores A e B, as regras de recuperação normal da queda de energia serão aplicadas. Quando voltar a energia, o quorum será restabelecido e a recuperação necessária da perda do quorum ocorrerá.

Comportamento do contêiner

Essa seção descreve como as JVMs do servidor de contêiner se comportam enquanto o quorum for perdido e recuperado.

Os contêineres hospedam um ou mais shards. Os shards são primários ou réplicas de uma partição específica. O serviço de catálogo designa shards para um contêiner e o contêiner respeitará essa designação até novas instruções chegarem a partir do serviço de catálogo. Isso significa que, se um shard primário em um contêiner não puder se comunicar com um shard réplica devido a uma queda de energia, ele continuará tentando novamente até receber novas instruções do serviço de catálogo.

Se cair a energia da rede e um shard primário perder comunicação com a réplica, ele tentará novamente a conexão até o serviço de catálogo fornecer novas instruções.

Comportamento de réplica síncrona

Enquanto a conexão estiver dividida, o shard primário pode aceitar novas transações enquanto houver, pelo menos, quantas réplicas on-line que a propriedade minsync possuir para o conjunto de mapa. Se alguma das novas transações for processada no shard primário enquanto o link para a réplica síncrona estiver quebrado, todos os dados da réplica serão excluídos e a réplica será ressincronizada com o estado atual do shard primário quando o link for reestabelecido.

A replicação síncrona é altamente não recomendada entre os datacenter ou por meio de um link de estilo WAN.

Comportamento de réplica assíncrona

Enquanto a conexão estiver interrompida, o shard primário pode aceitar novas transações. O shard primário armazenará em buffer as alterações até um limite. Se a conexão com a réplica for reestabelecida antes de atingir esse limite, a réplica será atualizada com as alterações em buffer. Se esse limite for atingido, o shard primário destruirá a lista armazenada em buffer e, quando a réplica for reconectada, ela será limpa e ressincronizada.

Comportamento do Cliente

Os clientes sempre são capazes de conectar ao servidor de catálogo para realizar uma autoinicialização para a grade se o domínio do serviço de catálogo tiver quorum ou não. O cliente tentará se conectar com qualquer instância do servidor de catálogos para obter uma tabela de rota e, em seguida, interagir com a grade. A conectividade de rede pode impedir que o cliente interaja com algumas partições devido à configuração da rede. O cliente pode se conectar com réplicas locais para dados remotos se ele for configurado para isso. Os clientes não poderão atualizar os dados se a partição primária para esses dados não estiver disponível.

Comandos do Quorum com xsadmin

Essa seção descreve os comandos xsadmin úteis para situações de quorum.

Consultando status de quorum

O status de quorum de uma instância do servidor de catálogos pode ser examinado usando o comando xsadmin.

```
xsadmin –ch cathost –p 1099 –quorumstatus
```

Há cinco resultados possíveis.

- Quorum está desativado: Os servidores de catálogos estão em execução em um modo de quorum desativado. Esse é um desenvolvimento ou apenas um modo de datacenter único. Ele não é recomendado para configurações de vários datacenters.
- O Quorum está ativado e o servidor de catálogos possui quorum: O quorum é ativado e o sistema trabalha normalmente.
- O Quorum está ativado mas o servidor de catálogos está aguardando quorum: O quorum está ativado mas o quorum foi perdido.
- O Quorum está ativado mas o quorum foi substituído: O quorum está ativado mas o quorum foi substituído.
- O status do quorum é declarado ilegal: Em uma queda de energia, o servidor de catálogos é dividido em duas partições, A e B. O servidor de catálogos A possui quorum substituído. A partição de rede é resolvida e o servidor na partição B é declarado ilegal, requerendo um reinício da JVM. Isso também ocorre se a JVM de catálogo na partição B for reiniciada durante a queda de energia e, em seguida, a energia voltar.

Substituindo Quorum

O comando xsadmin pode ser utilizado para substituir o quorum. Qualquer instância do servidor de catálogos sobrevivente pode ser usada. Todas as instâncias sobreviventes são notificadas quando uma for instruída para substituir o quorum. A sintaxe para isso é a seguinte.

```
xsadmin –ch cathost –p 1099 –overridequorum
```

Comandos de diagnóstico

- Status de quorum: Conforme descrito na seção anterior.
- Lista de grupo principal: Exibe uma lista de todos os grupos principais. Os membros e líderes dos grupos principais são exibidos.

  ```
  xsadmin –ch cathost –p 1099 –coregroups
  ```
- Servidores em estado teardown: Esse comando remove um servidor manualmente da grade. Normalmente isso não é necessário já que os servidores são automaticamente removidos quando são detectados como com falha, mas o comando é fornecido para uso na ajuda de suporte da IBM.

  ```
  xsadmin –ch cathost –p 1099 –g Grid –teardown server1,server2,server3
  ```
- Exibir tabela de rota: Esse comando mostra a tabela de rota atual ao simular uma nova conexão do cliente com a grade. Ele também valida a tabela de rota ao confirmar que todos os servidores de contêineres reconhecem sua função na tabela de rota, como o tipo de shard para uma determinada partição.

  ```
  xsadmin –ch cathost –p 1099 –g myGrid -routetable
  ```
- Exibir shards não-designados: Se algum shard não puder ser colocado na grade, isso poderá ser usado para listá-los. Isso acontece apenas quando o serviço de colocação possuir uma restrição que impede a colocação. Por exemplo, se você iniciar as JVMs em uma única caixa física enquanto estiver no modo de produção, apenas os shards primários poderão ser colocados. As réplicas não serão designadas até o início das JVMs em uma segunda caixa. O serviço de

colocação coloca apenas as réplicas nas JVMs com endereços IP diferentes das JVMs que hospedam os shards primários. Não ter nenhuma JVM em uma zona também fazem com que os shards não sejam designados.

```
xsadmin –ch cathost –p 1099 –g myGrid –unassigned
```

- Definir configurações de rastreio: Esse comando define as configurações de rastreio para todas as JVMs que correspondem ao filtro especificado para o comando xsadmin. Essa definição altera apenas as configurações de rastreio até outro comando ser usado ou as JVMs modificadas falharem ou pararem.

  ```
  xsadmin –ch cathost –p 1099 –g myGrid –fh host1 –settracespec
  ObjectGrid*=event=enabled
  ```

  Isso ativa o rastreio para todas as JVMs na caixa com o nome do host especificado que, neste caso, é host1.
- Verificando tamanhos de mapa: O comando de tamanhos de mapa é útil para verificar se a distribuição de chaves é uniforme entre os shards na chave. Se algum contêiner contiver significativamente mais chaves do que outros, provavelmente a função hash nos objetos de chave está distribuindo incorretamente.

  ```
  xsadmin -ch cathost -p 1099 -g myGrid -m myMapSet -mapsizes myMap
  ```

Considerações de Segurança de Transporte

Como normalmente os datacenters são implementados em diferentes localizações geográficas, os usuários podem querer ativar a segurança do transporte entre os datacenters por motivos de segurança.

Leia sobre a segurança da camada de transporte no *Guia de Administração*.

Lista de Verificação Operacional

Use a lista de verificação operacional para preparar o ambiente para implementar o WebSphere eXtreme Scale.

Tabela 6. Lista de Verificação Operacional

Item de Lista de Verificação	Para obter informações adicionais

Se você estiver usando o AIX, ajuste as seguintes configurações do sistema operacional:
TCP_KEEPINTVL
A configuração TCP_KEEPINTVL faz parte de um protocolo de soquete keep-alive que permite a detecção de uma interrupção de rede. A propriedade especifica o intervalo entre os pacotes que são enviados para validar a conexão. Ao usar o WebSphere eXtreme Scale, configure o valor para `10`. Para verificar a configuração atual, execute o seguinte comando:
`# no –o tcp_keepintvl`
Para alterar a configuração atual, execute o seguinte comando:
`# no –o tcp_keepintvl=10`
A configuração TCP_KEEPINTVL está em unidades de meio segundo.
TCP_KEEPINIT
A configuração TCP_KEEPINIT faz parte de um protocolo de soquete keep-alive que permite a detecção de uma interrupção de rede. A propriedade especifica o valor de tempo limite inicial para a conexão TCP. Ao usar o WebSphere eXtreme Scale, configure o valor para `40`. Para verificar a configuração atual, execute os seguintes comandos:
`# no –o tcp_keepinit`
Para alterar a configuração atual, execute o seguinte comando:
`# no –o tcp_keepinit=40`
A configuração TCP_KEEPINIT está em unidades de meio segundo.
	• Para obter informações de ajusta do AIX, consulte Ajustando Sistemas AIX.

Atualize o arquivo `orb.properties` para modificar o comportamento de transporte da grade. O arquivo `orb.properties` está no diretório `java/jre/lib`.	"Arquivo de Propriedades ORB" na página 192

Tabela 6. Lista de Verificação Operacional (continuação)

Item de Lista de Verificação	Para obter informações adicionais
Use parâmetros no script startOgServer. Em particular, use os seguintes parâmetros: • Defina as configurações de heap com o parâmetro **-jvmArgs**. • Defina o caminho de classe e as propriedades do aplicativo com o parâmetro **-jvmArgs**. • Defina os parâmetros **-jvmArgs** para configurar o monitoramento de agente. **Configurações de Porta** O WebSphere eXtreme Scale precisa abrir portas para comunicações para alguns transportes. Essas portas são todas definidas dinamicamente. Porém, se um firewall estiver em uso entre os contêineres, será necessário especificar as portas. Use as seguintes informações sobre as portas: **Porta Listener** É possível usar o argumento **-listenerPort** para especificar a porta que é usada para comunicação entre processos. **Porta do grupo principal** É possível usar o argumento **-haManagerPort** para especificar a porta que é usada para detecção de falha. Este argumento é o mesmo que peerPort. Observe que os grupos principais não precisam se comunicar entre as zonas, portanto, pode não ser necessário configurar essa porta se o firewall estiver aberto para todos os membros de uma única zona. **Porta de serviço JMX** É possível usar o argumento **-JMXServicePort** para especificar a porta que o serviço JMX deve usar. **Porta SSL** Transmitir `-Dcom.ibm.CSI.SSLPort=1234` como um argumento **-jvmArgs** configura a porta SSL para `1234`. A porta SSL é a porta protegida equivalente à porta listener. **Porta do cliente** Usada apenas no serviço de catálogo. É possível especificar esse valor com o argumento **-catalogServiceEndPoints**. O formato do valor desse parâmetro está no formato: `serverName:hostName:clientPort:peerPort`	"Script startOgServer" na página 334
Verifique se as configurações de segurança estão definidas corretamente: • Transporte (SSL) • Aplicativo (Autenticação e Autorização) Para verificar as configurações de segurança, é possível tentar usar um cliente malicioso para conectar-se com a sua configuração. Por exemplo, quando a configuração SSL necessária for definida, um cliente que possuir a configuração TCP_IP ou um cliente com o truststore incorreto não conseguirá se conectar ao servidor. Quando a autenticação é necessária, um cliente com nenhuma credencial, como um ID de usuário e senha, não conseguirá se conectar ao servidor. Quando a autorização é forçada, um cliente com nenhuma autorização de acesso não conseguira acessar os recursos do servidor.	"Integração de Segurança com Provedores Externos" na página 368
Escolha como você monitorará seu ambiente. • xsAdmin – As portas JMX dos servidores de catálogo precisam estar visíveis para a ferramenta XSAdmin. As portas do contêiner também precisam estar acessíveis para alguns comandos que reúnem informações a partir dos contêineres. • É possível escolher entre as seguintes ferramentas de monitoramento do fornecedor: – Tivoli Enterprise Monitoring Agent – CA Wily Introscope – Hyperic HQ	• "Monitorando com o Utilitário de Amostra xsAdmin" na página 385 • "Segurança do Java Management Extensions (JMX)" na página 366 • "Monitorando com o IBM Tivoli Enterprise Monitoring Agent para WebSphere eXtreme Scale" na página 408 • "Monitorando o eXtreme Scale com o Hyperic HQ" na página 418 • "Monitorando Aplicativos eXtreme Scale com o CA Wily Introscope" na página 414

Capítulo 6. Configurando o Ambiente de Implementação

É possível configurar WebSphere eXtreme Scale para execução em um ambiente independente ou pode configurar eXtreme Scale para execução em um ambiente com WebSphere Application Server ou WebSphere Application Server Network Deployment. Para que uma implementação do eXtreme Scale selecione alterações de configuração no lado da grade do servidor, será necessário reiniciar os processos para que essas alterações tomem efeito em vez de serem aplicadas dinamicamente. Porém, no lado do cliente, embora não seja possível alterar as definições de configuração para uma instância de cliente existente, será possível criar um novo cliente com as configurações necessárias ao usar um arquivo XML ou fazer isso programaticamente. Ao criar um cliente, será possível substituir as configurações padrão fornecidas com a configuração do servidor atual.

Métodos de Configuração

Os arquivos XML e os arquivos de propriedades são os métodos não programáticos mais comuns de configuração do produto. Consulte o Guia de Programação para obter informações sobre métodos alternativos, incluindo interfaces de programação do sistema e do aplicativo, plug-ins e beans gerenciados.

Configurando com os Arquivos XML

O WebSphere eXtreme Scale é configurado por uma coleta de arquivos XML.

As seguintes configurações podem ser feitas para o eXtreme Scale usando arquivos XML:

- Política de implementação - Para configurar uma política de implementação, use um arquivo XML do descritor de política de implementação. Leia sobre o Arquivo XML do descritor da política de implementação para definir os elementos e atributos do arquivo XML do descritor para vários requisitos.
- Descritor do ObjectGrid - Para configurar os detalhes para instâncias individuais do ObjectGrid, use um arquivo XML do descritor. Leia sobre o “Arquivo XML descritor do ObjectGrid” na página 141.
- Configurando entidades - Com base em suas necessidades de entidades em sua implementação conforme definido em umm esquema lógico, é possível configurá-las usando classes Java anotadas, XML ou uma combinação de ambos. As entidades definidas são, então, registradas com um servidor eXtreme Scale e ligadas a BackingMaps, índices e outros plug-ins. Um esquema de entidade é um conjunto de entidades e relacionamentos entre as entidades. Leia sobre o “Configurando Entidades” na página 214 para obter mais detalhes sobre como otimizar seus esquemas de entidade por meio de opções de configuração.
- Configuração de segurança - É possível ativar a segurança para uma determinada implementação usando arquivos de configuração XML. Leia sobre o “Arquivo XML Descritor de Segurança” na página 373 para obter mais informações sobre opções de configuração de ativação de segurança.
- Configuração do cliente - Use um arquivo de propriedades e outros métodos para especificar nomes do host do cliente, portas, segurança e outras informações. Leia sobre o “Configurando Clientes” na página 201 para obter detalhes adicionais sobre como customizar clientes com um arquivo XML.

- Configurando a integração de Spring - É possível ativar o Spring Framework para trabalhar com um ambiente de implementação do eXtreme Scale com scripts XML e uma definição de esquema juntamente com outros métodos, como com beans de extensão e suporte a espaço de nomes. Leia sobre o "Configurando Integração de Spring" na página 273 para obter detalhes sobre como usar o eXtreme Scale com o Spring Framework.

Resolução de Problemas de Configuração XML

Ocasionalmente, quando configura eXtreme Scale, é possível encontrar um comportamento inesperado na configuração XML.

Na seção a seguir são apresentados diversos problemas de configuração XML que podem ocorrer.

Política de Implementação e Arquivos XML do Incompatíveis

A política de implementação e os arquivos XML de ObjectGrid devem corresponder. Se eles não possuem nomes de ObjectGrid e nomes de mapas correspondentes, ocorrem erros.

BackingMap e referências de mapa incorretos

Se a lista de backingMap em um arquivo XML de ObjectGrid não corresponder à lista de referências de mapas em um arquivo XML da política de implementação, ocorre um erro no servidor de catálogos.

Por exemplo, o seguinte arquivo XML de ObjectGrid e arquivo XML da política de implementação são utilizados para iniciar um processo de contêiner. O arquivo da política de implementação possui mais referências de mapas que são listados no arquivo XML de ObjectGrid.

ObjectGrid.xml - exemplo incorreto

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
    <objectGrids>
        <objectGrid name="accounting">
            <backingMap name="payroll" readOnly="false" />
        </objectGrid>
    </objectGrids>
</objectGridConfig>
```

deploymentPolicy.xml - exemplo incorreto

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">
    <objectgridDeployment objectgridName="accounting">
        <mapSet name="mapSet1" numberOfPartitions="4" minSyncReplicas="1" maxSyncReplicas="2" maxAsyncReplicas="1">
            <map ref="payroll"/>
            <map ref="ledger"/>
        </mapSet>
    </objectgridDeployment>
</deploymentPolicy>
```

Mensagens

Uma mensagem de erro ocorre no arquivo `SystemOut.log` quando a política de implementação é incompatível com o arquivo XML do ObjectGrid. Para o exemplo precedente, ocorre a seguinte mensagem:

```
CWOBJ3179E: A referência do ledger do mapa no mapSet mapSet1 do arquivo do descritor de
 implementação de contabilidade do ObjectGrid
não faz referência a um mapa de retorno válido do XML do ObjectGrid
.
```

Se a política de implementação estiver sem referências de mapa para os backingMaps listados no arquivo XML do ObjectGrid, ocorrerá uma mensagem de erro no arquivo `SystemOut.log`. Por exemplo:

```
CWOBJ3178E: O ledger do mapa na contabilidade do ObjectGrid referido no XML do ObjectGrid
não foi localizado no arquivo do descritor de implementação.
```

Problema

A lista backingMap no arquivo XML de ObjectGrid e as referências de mapas na política de implementação devem ser correspondentes.

Solução

Determine qual lista é correta para seu ambiente e corrija o arquivo XML que contém a lista incorreta.

Nomes Incorretos do ObjectGrid

O nome do ObjectGrid é referido no arquivo XML ObjectGrid e no arquivo XML da política de implementação.

Mensagem

Ocorre uma ObjectGridException causada por uma exceção de IncompatibleDeploymentPolicyException. A seguir, está um exemplo.

Caused by:
com.ibm.websphere.objectgrid.IncompatibleDeploymentPolicyException: The objectgridDeployment with objectGridName accountin does not have a corresponding objectGrid in the ObjectGrid XML.

Problema

O arquivo XML ObjectGrid é a lista principal dos nomes ObjectGrid. Ocorrerá um erro se uma política de implementação possuir um nome ObjectGrid que não está contido no arquivo XML ObjectGrid.

Solução

Verifique se o nome ObjectGrid está correto. Remova quaisquer nomes extras ou inclua nomes ObjectGrid ausentes nos arquivos XML ObjectGrid ou da política de implementação. Na mensagem de exemplo, o objectGridName está escrito incorretamente como "accountin" em vez de "accounting".

O Valor de Atributo XML não é Válido

A alguns dos atributos no arquivo XML podem ser designados apenas alguns valores. Estes atributos possuem valores aceitáveis enumerados pelo esquema. A lista a seguir fornece alguns dos atributos:

- atributo authorizationMechanism no elemento objectGrid
- atributo copyMode no elemento backingMap
- atributo lockStrategy no elemento backingMap
- atributo ttlEvictorType no elemento backingMap
- atributo type no elemento property
- initialState no elemento objectGrid

- evictionTriggers no elemento backingMap

Se a um destes atributos for designado um valor inválido, a validação XML falhará. No arquivo XML de exemplo a seguir, é utilizado um valor incorreto de INVALID_COPY_MODE:

exemplo de INVALID_COPY_MODE

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
    <objectGrids>
        <objectGrid name="accounting">
            <backingMap name="payroll" copyMode="INVALID_COPY_MODE"/>
        <objectGrid/>
   </objectGrids>
</objectGridConfig>
```

A mensagem a seguir aparece no log.

```
CWOBJ2403E: O arquivo XML é inválido. Foi detectado um problema com <
null > na linha 5. A mensagem de erro é cvc-enumeration-valid: O valor 'INVALID_COPY_MODE' não possu
Ela deve ser um valor a partir da enumeração.
```

Atributos ou Tags Ausentes

Ocorrem erros se um arquivo XML não possuir os atributos ou tags corretos. Por exemplo, o seguinte arquivo XML do ObjectGrid não possui a tag final < /objectGrid >:

atributos ausentes - XML de exemplo

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
    <objectGrids>
        <objectGrid name="accounting">
            <backingMap name="payroll" />

   </objectGrids>
</objectGridConfig>
```

A mensagem a seguir aparece no log.

```
CWOBJ2403E: O arquivo XML é inválido. Foi detectado um
problema com < null > na linha 7. A mensagem de erro é: The
end-tag for element type "objectGrid" must end with a '>' delimiter.
```

Ocorre uma ObjectGridException sobre o arquivo XMl inválido com o nome do arquivo XML

Erros de Sintaxe

Se um arquivo XML é formatado com sintaxe incorreta ou ausente, o CWOBJ2403E aparece no log. Por exemplo, a mensagem a seguir é exibida quando um sinal de aspas está ausente em um dos atributos XML.

```
CWOBJ2403E: O arquivo XML é inválido. Foi detectado
um problema com < null > na linha 7. A mensagem de erro é: Open quote
 is expected for attribute "maxSyncReplicas"
associated with an element type "mapSet".
```

An ObjectGridException about the invalid XML file also occurs.

Fazendo Referência a uma Coleta de Plug-in Inexistente

Ao utilizar o XML para definir plug-ins do BackingMap, o atributo pluginCollectionRef do elemento backingMap deve fazer referência a um backingMapPluginCollection. O atributo pluginCollectionRef deve corresponder com o ID de um dos elementos backingMapPluginCollection.

Mensagem

Se o atributo pluginCollectionRef não corresponder a nenhum dos atributos de ID de algum dos elementos backingMapPluginConfiguration, uma mensagem semelhante à seguinte será exibida no log.

```
[7/14/05 14:02:01:971 CDT] 686c060e XmlErrorHandl E CWOBJ9002E:
This is an English only Error message: Invalid XML file. Line: 14; URI: null;
Message: Key 'pluginCollectionRef' with
value 'bookPlugins' not found for identity
constraint of element 'objectGridConfig'.
```

Problema

O arquivo XML a seguir é utilizado para produzir o erro. Observe que o nome do BackingMap book possui seu atributo pluginCollectionRef configurado como bookPlugins e o backingMapPluginCollection único possui um ID de collection1.

Referenciando um XML de atributo não-existente - exemplo

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
   <objectGrids>
      <objectGrid name="bookstore">
         <backingMap name="book" pluginCollectionRef="bookPlugin" />
      </objectGrid>
   </objectGrids>
   <backingMapPluginCollections>
    <backingMapPluginCollection id="collection1">
     <bean id="Evictor"
       className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" />
     </backingMapPluginCollection>
    </backingMapPluginCollections>
</objectGridConfig>
```

Solução

Para corrigir o problema, certifique-se de que o valor de cada pluginCollectionRef corresponda ao ID de um dos elementos backingMapPluginCollection. Simplesmente altere o nome de pluginCollectionRef para collection1 para não receber este erro. Outras maneiras de corrigir o problema incluem a mudança do ID do backingMapPluginCollection existente para corresponder ao pluginCollectionRef ou a inclusão de um backingMapPluginCollection adicional com um ID que corresponda ao pluginCollectionRef.

Validando XML sem Suporte de uma Implementação

O IBM Software Development Kit (SDK) Versão 1.4.2 contém uma implementação de alguma função JAXP (Java API for XML Processing) para usar para validação XML contra um esquema.

Ao utilizar um SDK que não contém esta implementação, as tentativas de validação poderão falhar. Se desejar validar o XML usando um SDK que não contém esta implementação, efetue download do Apache Xerces e inclua seus arquivos JAR (Java archive) no caminho de classe.

Ao tentar validar o XML com um SDK que não possui a implementação necessária, o log contém o seguinte erro:

```
A validação XML de XmlConfigBuild está ativada
SystemErr R com.ibm.websphere.objectgrid
SystemErr R at
com.ibm.ws.objectgrid.ObjectGridManagerImpl.getObjectGridConfigurations
 (ObjectGridManagerImpl.java:182)
SystemErr R at com.ibm.ws.objectgrid.ObjectGridManagerImpl.createObjectGrid(ObjectGridManagerImpl.java:309)
SystemErr R at com.ibm.ws.objectgrid.test.config.DocTest.main(DocTest.java:128)
SystemErr R Caused by: java.lang.IllegalArgumentException: No attributes are implemented
SystemErr R at org.apache.crimson.jaxp.DocumentBuilderFactoryImpl.setAttribute(DocumentBuilderFactoryImpl.java:93)
SystemErr R at com.ibm.ws.objectgrid.config.XmlConfigBuilder.<init>XmlConfigBuilder.java:133)
SystemErr R at com.ibm.websphere.objectgrid.ProcessConfigXML$2.runProcessConfigXML.java:99)...
```

O SDK utilizado não contém uma implementação da função JAXP necessária para validar arquivos XML em um esquema.

Para evitar este problema, depois de efetuar download do Xerces e incluir os arquivos JAR no caminho de classe,é possível validar o arquivo XML com êxito.

Referência de Arquivo de Propriedades

Os arquivos de propriedades do servidor contêm configurações para executar os servidores de catálogo e servidores de contêiner. É possível especificar um arquivo de propriedades do servidor para uma configuração do WebSphere Application Server ou independente. Os arquivos de propriedades do cliente contêm configurações para o cliente.

Amostras de Arquivos de Propriedades

É possível usar as seguintes amostras de arquivos de propriedades que estão no diretório *extremescale_root*\properties para criar o arquivo de propriedades:

- sampleServer.properties
- sampleClient.properties

Propriedades de Sistema Reprovadas

-Dcom.ibm.websphere.objectgrid.CatalogServerProperties
A propriedade foi reprovada no WebSphere eXtreme Scale Versão 7.0. Use a propriedade **-Dobjectgrid.server.props**.

-Dcom.ibm.websphere.objectgrid.ClientProperties
A propriedade foi reprovada no WebSphere eXtreme Scale Versão 7.0. Use a propriedade **-Dobjectgrid.client.props**.

-Dobjectgrid.security.server.prop
A propriedade foi reprovada no WebSphere eXtreme Scale Versão 6.1.0.3. Use a propriedade **-Dobjectgrid.server.prop**.

-serverSecurityFile
Este argumento foi reprovado no WebSphere eXtreme Scale Versão 6.1.0.3. Essa opção é transmitida no script startOgServer. Use a propriedade **-serverProps**.

Configurando Grades

Use um arquivo XML do descritor de ObjectGrid para configurar grades, mapas de apoio, plug-ins etc. Para configurar o WebSphere® eXtreme Scale, use um arquivo XML do descritor de ObjectGrid e a API de ObjectGrid. Para uma topologia distribuída, será necessário não apenas um arquivo XML do descritor de ObjectGrid, mas um arquivo XML de política de implementação.

Configurando Implementações Locais

Uma configuração do eXtreme Scale na memória local pode ser criada com o uso de um arquivo XML do descritor de ObjectGrid ou APIs do eXtreme Scale.

Sobre Esta Tarefa

O arquivo `companyGrid.xml` a seguir é um exemplo de um XML de descritor de ObjectGrid. As primeiras linhas do arquivo incluem o cabeçalho obrigatório de cada arquivo XML do ObjectGrid. O arquivo define uma instância de ObjectGrid denominada "CompanyGrid" e vários BackingMaps denominados "Customer," "Item," "OrderLine" e "Order."

arquivo companyGrid.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Customer"/>
   <backingMap name="Item" />
   <backingMap name="OrderLine" />
   <backingMap name="Order"/>
  </objectGrid>
 </objectGrids>

</objectGridConfig>
```

Transmita o arquivo XML para um dos métodos na createObjectGrid na interface ObjectGridManager. A amostra de código a seguir valida o arquivo `companyGrid.xml` com relação ao esquema XML e cria a instância de ObjectGrid denominada "CompanyGrid." A instância do ObjectGrid recém-criada não é armazenada em cache.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid",
new URL("file:etc/test/companyGrid.xml"), true, false);
```

Como uma alternativa, é possível criar instâncias do ObjectGrid programaticamente sem nenhum XML. Por exemplo, é possível utilizar o seguinte fragmento de código no lugar do XML e código anteriores.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
BackingMap customerMap = companyGrid.defineMap("Customer");
BackingMap itemMap = companyGrid.defineMap("Item");
BackingMap orderLineMap = companyGrid.defineMap("OrderLine");
BackingMap orderMap = companyGrid.defineMap("Order");
```

Para obter uma descrição completa do arquivo XML do ObjectGrid, consulte eXtreme Scale referência de configuração.

Configurando o HashIndex

O HashIndex (classe com.ibm.websphere.objectgrid.plugins.index.HashIndex integrada) é um plug-in MapIndexPlugin que é possível incluir no BackingMap para construir índices estáticos ou dinâmicos. Ele suporta as interfaces MapIndex e MapRangeIndex. Definir e utilizar índices adequadamente pode aprimorar significativamente o desempenho da consulta.

Para obter informações sobre indexação, consulte Indexação e HashIndex Composto. Para obter informações sobre como usar a indexação, consulte Usando a indexação para acesso a dados não principais e HashIndex Composto.

Atributos para Configurar HashIndex

É possível usar os seguintes atributos para configurar HashIndex usando o arquivo XML do descritor de implementação ObjectGrid ou uma abordagem programática:

Name Especifica o nome do índice. O nome deve ser exclusivo para cada mapa. O nome é usado para recuperar o objeto do índice da instância de ObjectMap para o BackingMap.

AttributeName
Especifica os nomes delimitados por vírgula dos atributos a serem indexados. Para índices acessados por campo, os nomes de atributos são equivalentes aos nomes de campo. Para índices acessados por propriedade, os nomes de atributos são os nomes da propriedade compatível com JavaBean. Se existir somente um nome de atributo, o HashIndex é um índice de atributo único, e se este atributo for um relacionamento, ele também é um índice de relacionamento. Se múltiplos nomes de atributo são incluídos nos nomes de atributo, o HashIndex é um índice composto.

FieldAccessAttribute
Usado para mapas sem entidade. Se `true`, o objeto será acessado usando os campos diretamente. Se não especificado ou `false`, o método getter do atributo será usado para acessar os dados.

POJOKeyIndex
Usado para mapas sem entidade. Se `true`, o índice examinará o objeto na parte principal do mapa. Isto é útil quando a chave é uma chave composta e o valor não tem a chave integrada dentro dele. Se não especificado ou `false`, o índice examinará o objeto na parte do valor do mapa.

RangeIndex
Se `true`, a indexação do intervalo será ativada e o aplicativo poderá lançar o objeto do índice recuperado para a interface MapRangeIndex. Se a propriedade RangeIndex for configurada como `false`, o aplicativo poderá apenas lançar o objeto do índice recuperado para a interface MapIndex.

Incluindo HashIndex em BackingMap

Há poucas abordagens que podem ser usadas para incluir HashIndex em BackingMap. O exemplo a seguir ilustra a abordagem de configuração XML incluindo plug-ins de índice estático:

Abordagem de configuração XML para incluir HashIndex em BackingMap

```
<backingMapPluginCollection id="person">
   <bean id="MapIndexplugin"
      className="com.ibm.websphere.objectgrid.plugins.index.HashIndex">
         <property name="Name" type="java.lang.String" value="CODE"
```

```
description="index name" />
         <property name="RangeIndex" type="boolean" value="true"
            description="true for MapRangeIndex" />
         <property name="AttributeName" type="java.lang.String"
value="employeeCode" description="attribute name" />
   </bean>
</backingMapPluginCollection>
```

Neste exemplo de configuração XML, a classe HashIndex integrada é usada como o plug-in de índice. O HashIndex suporta propriedades que os usuários podem configurar, como Name, RangeIndex e AttributeName no exemplo anterior.

- A propriedade Name é configurada como `CODE`, uma cadeia que identifica este plug-in de índice. O valor da propriedade Name deve ser único dentro do escopo do BackingMap, e pode ser usado para recuperar o objeto do índice pelo nome a partir da instância de ObjectMap para o BackingMap.
- A propriedade RangeIndex é configurada como `true`, o que significa que o aplicativo pode lançar o objeto do índice recuperado para a interface MapRangeIndex. Se a propriedade RangeIndex for configurada como `false`, o aplicativo poderá apenas lançar o objeto do índice recuperado para a interface MapIndex. Um MapRangeIndex suporta funções para localizar dados usando funções de intervalo, como maior que, menor que, ou ambas, enquanto um MapIndex suporta apenas funções iguais. Se o índice for usado para consulta, a propriedade RangeIndex deverá ser configurada para `true` em índices de atributo único. Para um índice de relacionamento e um índice composto, a propriedade RangeIndex deve ser configurada para `false`.
- A propriedade AttributeName é configurada como `employeeCode`, o que significa que o atributo employeeCode do objeto em cache é usado para construir um índice de atributo único. Se um aplicativo precisar procurar por objetos em cache com múltiplos atributos, a propriedade AttributeName pode ser configurada para uma lista de atributos delimitados por vírgula, rendendo um índice composto.

Em resumo, o exemplo anterior define um HashIndex do intervalo de atributo único. É um hashIndex de atributo único. É também um HashIndex de intervalo.

HashIndex de Atributo Único Versus HashIndex Composto

Quando a propriedade AttributeName de HashIndex inclui múltiplos nomes de atributos, o HashIndex é um índice composto. Caso contrário, se ela incluir somente um nome de atributo, ela é um índice de atributo único. Por exemplo, o valor da propriedade AttributeName de um HashIndex composto pode ser `city,state,zipcode`. Ela inclui três atributos delimitados por vírgulas. Se o valor da propriedade AttributeName for apenas `zipcode` que tem apenas um atributo, ele será um HashIndex de atributo único. O exemplo precedente é um HashIndex de atributo único porque o valor da propriedade AttributeName é “employeeCode”, que inclui apenas um nome de atributo.

O HashIndex composto fornece uma maneira eficiente de consultar objetos em cache quando os critérios de busca envolvem muitos atributos. Porém, ele não suporta o índice de intervalo e sua propriedade RangeIndex deve ser definida para “false”.

Consulte o tópico sobre HashIndex composto no *Guia de Administração*.

HashIndex de Relacionamento

Se o atributo indexado de um HashIndex de atributo único for um relacionamento, tanto com valor único ou múltiplos valores, o HashIndex é um HashIndex de relacionamento. Para HashIndex de relacionamento, a propriedade RangeIndex de HashIndex deve ser definida para “false”.

O HashIndex de relacionamento pode acelerar as consultas que usam referências cíclicas ou usam os filtros de consulta IS NULL, IS EMPTY, SIZE e MEMBER OF. Consulte otimização de consulta usando índices no *Guia de Programação*.

HashIndex Principal

Para mapas sem entidade, quando a propriedade POJOKeyIndex de HashIndex é configurada para `true`, o HashIndex é um HashIndex principal e a parte principal da entrada será usada para indexação. Quando a propriedade AttributeName do HashIndex não é especificada, a chave inteira é indexada; caso contrário, o HashIndex principal poderá ser apenas um HashIndex de atributo único.

Por exemplo, a inclusão da seguinte propriedade na amostra precedente faz com que o HashIndex se torne o HashIndex principal porque o valor da propriedade POJOKeyIndex é `true`.

```
<property name="POJOKeyIndex" type="boolean" value="true"
description="indicates if POJO key HashIndex" />
```

No exemplo precedente do índice principal, como o valor da propriedade AttributeName é especificado como `employeeCode`, o atributo indexado é o campo employeeCode da parte principal da entrada do mapa. Se você quiser construir o índice principal na parte principal inteira da entrada do mapa, remova a propriedade AttributeName.

HashIndex de Intervalo

Quando a propriedade RangeIndex do HashIndex é configurada para `true`, o HashIndex é um índice de intervalo e pode suportar a interface MapRangeIndex. UmMapRangeIndex suporta funções para localizar dados usando funções de intervalo, como maior que, menor que, ou ambas, enquanto um MapIndex suporta apenas funções iguais. Para um índice de atributo único, a propriedade RangeIndex pode ser configurada para `true` apenas se o atributo indexado for do tipo Comparable. Se o índice de atributo único for usado pela consulta, a propriedade RangeIndex deverá ser configurada para `true` e o atributo indexado deverá ser do tipo Comparable. Para o HashIndex de relacionamento e o HashIndex composto, a propriedade RangeIndex deve ser configurada para `false`.

A amostra precedente é um HashIndex de intervalo porque o valor da propriedade RangeIndex é `true`.

A tabela a seguir fornece um resumo do uso do índice de intervalo.

Tabela 7. Suporte para Índice de Intervalo. Define se os tipos de HashIndex suportam o índice de intervalo.

Tipo HashIndex	**Suporta índice de intervalo**
HashIndex de atributo único: chave ou atributo indexado é do tipo Comparable	Yes

Tabela 7. Suporte para Índice de Intervalo (continuação). Define se os tipos de HashIndex suportam o índice de intervalo.

Tipo HashIndex	Suporta índice de intervalo
HashIndex de atributo único: chave ou atributo indexado não é do tipo Comparable	Não
HashIndex Composto	Não
HashIndex de Relacionamento	Não

Otimização de Consulta Utilizando HashIndex

Definir e utilizar índices adequadamente pode aprimorar significativamente o desempenho da consulta. As consultas do WebSphere eXtreme Scale podem usar plug-ins do HashIndex integrados para aprimorar o desempenho das consultas. Embora o uso de índices possa aprimorar significativamente o desempenho da consulta, isso pode ter um impacto no desempenho nas operações do mapa transacional.

Configurando Evictores

Os evictores podem ser configurados usando o arquivo descritor XML ObjectGrid ou programaticamente.

Sobre Esta Tarefa

Para obter informações de referência sobre como configurar evictors com XML, consulte "Arquivo XML descritor do ObjectGrid" na página 141.

Evictor TimeToLive (TTL)

O WebSphere eXtreme Scale fornece um mecanismo padrão para despejar entradas do cache e um plug-in para criar evictors customizados. Um Evictor controla a associação de entradas em cada instância do BackingMap.

Ativar o Evictor TTL Programaticamente

Os evictors TTL estão associados a instâncias do BackingMap. O evictor padrão utiliza uma política de despejo time-to-live (TTL) para cada instância de BackingMap. Se você fornecer um mecanismo de evictor conectável, geralmente ele utilizará uma política de evicção baseada no número de entradas em vez de baseada no tempo.

O fragmento de código a seguir utiliza a interface BackingMap para configurar o prazo de expiração de cada entrada para 10 minutos após a entrada ser criada.

```
evictor time-to-live
programáticoimport
com.ibm.websphere.objectgrid.ObjectGridManagerFactory;
import com.ibm.websphere.objectgrid.ObjectGridManager;
import com.ibm.websphere.objectgrid.ObjectGrid;
import com.ibm.websphere.objectgrid.BackingMap;
import com.ibm.websphere.objectgrid.TTLType;
ObjectGridManager ogManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid og = ogManager.createObjectGrid("grid");
BackingMap bm = og.defineMap( "myMap" );
bm.setTtlEvictorType( TTLType.CREATION_TIME );
bm.setTimeToLive( 600 );
```

O argumento do método setTimeToLive é 600 porque isso indica que o valor de time-to-live está em segundos. O código anterior deve ser executado antes de o método initialize ser chamado na instância do ObjectGrid. Estes atributos de BackingMap não podem ser alterados após a inicialização da instância do ObjectGrid. Após a execução do código, qualquer entrada inserida no BackingMap myMap tem um tempo de expiração. Ao final do tempo de expiração, o evictor de TTL remove a entrada.

Para configurar o prazo de expiração para o horário do último acesso mais 10 minutos, altere o argumento que é transmitido para o método setTtlEvictorType de TTLType.CREATION_TIME para TTLType.LAST_ACCESS_TIME. Com este valor, o tempo de expiração é calculado como a hora do último acesso mais 10 minutos. Quando uma entrada é criada pela primeira vez, a hora do último acesso é a hora de criação. Para basear o prazo de expiração na última *atualização*, em vez de simplesmente no último *acesso* (se ele envolveu ou não uma atualização), substitua a configuração TTLType.LAST_UPDATE_TIME pela configuração TTLType.LAST_ACCESS_TIME.

Ao usar a configuração TTLType.LAST_ACCESS_TIME ou TTLType.LAST_UPDATE_TIME, é possível usar as interfaces ObjectMap e JavaMap para substituir o valor time-to-live de BackingMap. Esse mecanismo permite que um aplicativo utilize um valor de time-to-live diferente para cada entrada criada. Suponha que o fragmento de conjunto de códigos precedente configure o atributo ttlType como LAST_ACCESS_TIME e configure o valor de time-to-live como 10 minutos. Qualquer aplicativo pode substituir o valor de time-to-live para cada entrada executando o seguinte código antes de criar ou modificar uma entrada:

```
import com.ibm.websphere.objectgrid.Session;
import com.ibm.websphere.objectgrid.ObjectMap;
Session session = og.getSession();
ObjectMap om = session.getMap( "myMap" );
int oldTimeToLive1 = om.setTimeToLive( 1800 );
om.insert("key1", "value1" );
int oldTimeToLive2 = om.setTimeToLive( 1200 );
om.insert("key2", "value2" );
```

No fragmento de código anterior, a entrada com a chave key1 tem um prazo de expiração composto pelo tempo de inserção mais 30 minutos como resultado da solicitação de método setTimeToLive(1800) na instância de ObjectMap. A variável oldTimeToLive1 é configurada como 600 porque o valor de time-to-live de BackingMap é utilizado como um valor padrão se o método setTimeToLive não foi chamado anteriormente na instância de ObjectMap.

A entrada com a chave key2 tem um prazo de expiração composto pelo tempo de inserção mais 20 minutos como resultado da chamada de método setTimeToLive(1200) na instância de ObjectMap. A variável oldTimeToLive2 é configurada como 1800 porque o valor de time-to-live da solicitação de método ObjectMap.setTimeToLive anterior configura o valor de time-to-live como 1800.

O exemplo anterior mostra duas entradas de mapa sendo inseridas no mapa myMap para as chaves key1 e key2. Posteriormente, o aplicativo ainda pode atualizar essas entradas de mapa enquanto retém os valores de time-to-live que são utilizados no tempo de inserção para cada entrada de mapa. O exemplo a seguir ilustra como reter os valores de TTL (Time-to-Live), utilizando uma constante definida na interface ObjectMap:

```
Session session = og.getSession();
ObjectMap om = session.getMap( "myMap" );
om.setTimeToLive( ObjectMap.USE_DEFAULT );
```

```
session.begin();
om.update("key1", "updated value1" );
om.update("key2", "updated value2" );
om.insert("key3", "value3" );
session.commit();
```

Como o valor especial de ObjectMap.USE_DEFAULT é utilizado na chamada de método setTimeToLive, a chave key1 retém seu valor de time-to-live de 1800 segundos e a chave key2 retém seu valor de time-to-live de 1200 segundos, pois esses valores foram utilizados quando essas entradas de mapa foram inseridas pela transação anterior.

O exemplo anterior também mostra uma nova entrada de mapa para a inserção da chave key3. Neste caso, o valor especial USE_DEFAULT indica a utilização da configuração padrão do valor time-to-live para este mapa. O valor padrão é definido pelo atributo time-to-live de BackingMap. Consulte atributos da interface BackingMap para obter informações sobre como o atributo time-to-live é definido na instância do BackingMap.

Consulte a documentação da API para o método setTimeToLive nas interfaces ObjectMap e JavaMap. A documentação explica que o resultado será uma exceção IllegalStateException se o método BackingMap.getTtlEvictorType retornar qualquer coisa diferente do valor TTLType.LAST_ACCESS_TIME ou TTLType.LAST_UPDATE_TIME. As interfaces ObjectMap e JavaMap podem substituir o valor time-to-live apenas quando você está usando a configuração LAST_ACCESS_TIME ou TTLType.LAST_UPDATE_TIME para o tipo de evictor TTL. O método setTimeToLive não pode ser utilizado para substituir o valor de time-to-live quando você estiver utilizando a configuração de tipo de evictor CREATION_TIME ou NONE.

Ativar o Evictor TTL Utilizando a Configuração XML

Em vez de usar a interface BackingMap para programaticamente configurar os atributos de BackingMap a serem usados pelos Evictor TTL, é possível usar um arquivo XML para configurar cada instância de BackingMap. O código a seguir demonstra como configurar tais atributos para três mapas BackingMap diferentes:

Ativando evictor de tempo de vida usando XML

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
<objectGrids>
    <objectGrid name="grid1">
        <backingMap name="map1" ttlEvictorType="NONE" />
        <backingMap name="map2" ttlEvictorType="LAST_ACCESS_TIME|LAST_UPDATE_TIME"
     timeToLive="1800" />
        <backingMap name="map3" ttlEvictorType="CREATION_TIME" timeToLive="1200" />
    </objectgrid>
</objectGrids>
```

O exemplo anterior mostra que a instância map1 de BackingMap utiliza um tipo de evictor NONE TTL. A instância BackingMap map2 usa o tipo de evictor TTL LAST_ACCESS_TIME ou LAST_UPDATE_TIME – especifique uma ou outra dessas configurações – e tem um valor time-to-live de 1800 segundos ou 30 minutos. A instância map3 de BackingMap é definida para utilizar um tipo de evictor CREATION_TIME TTL e tem um valor de time-to-live de 1200 segundos ou 20 minutos.

Conexão de um Evictor programável

Como os evictors são associados com BackingMaps, utilize a interface BackingMap para especificar o evictor conectável.

Evictores Conectáveis Opcionais

O evictor TTL padrão utiliza uma política de evicção baseada no tempo e o número de entradas no BackingMap não tem efeito sobre o tempo de expiração de uma entrada. É possível utilizar um evictor conectável opcional para despejar entradas com base no número de entradas existentes em vez de no tempo.

Os seguintes evictores conectáveis opcionais fornecem alguns algoritmos comumente utilizados para decidir quais entradas liberar quando um BackingMap crescer além de algum limite de tamanho.

- O evictor LRUEvictor utiliza um algoritmo LRU (Least Recently Used) para decidir quais entradas serão despejadas quando o BackingMap exceder um número máximo de entradas.
- O evictor LFUEvictor utiliza um algoritmo LFU (Least Frequently Used) para decidir quais entradas serão despejadas quando o BackingMap exceder um número máximo de entradas.

O BackingMap informa um evictor conforme as entradas são criadas, modificadas ou removidas de uma transação. O BackingMap acompanha estas entradas e escolhe quando liberar uma ou mais entradas da instância do BackingMap.

Uma instância do BackingMap não possui informações de configuração para um tamanho máximo. Em vez disso, as propriedades do evictor são configuradas para controlar o comportamento do evictor. O LRUEvictor e o LFUEvictor possuem uma propriedade de tamanho máximo utilizada para fazer o evictor começar a liberar entradas quando o tamanho máximo for excedido. Assim como o evictor TTL, os evictores LRU e LFU podem não liberar imediatamente uma entrada quando o número máximo de entradas for atingido para minimizar o impacto no desempenho.

Se o algoritmo LRU ou LFU não for adequado para um determinado aplicativo, será possível redigir seus próprios evictors para criar a sua estratégia de despejo.

Utilizando Evictors Conectáveis Opcionais

Para incluir evictors conectáveis opcionais na configuração do BackingMap, é possível utilizar tanto a configuração programática quanto a configuração do XML.

Conectando um Evictor Conectável Programaticamente

Como os evictors estão associados aos BackingMaps, use a interface BackingMap para especificar o evictor conectável. O trecho de código a seguir é um exemplo de especificação de um evictor LRUEvictor para o BackingMap map1 e um evictor LFUEvictor para a instância do BackingMap map2:

Conectando um evictor programaticamente

```
import com.ibm.websphere.objectgrid.ObjectGridManagerFactory;
import com.ibm.websphere.objectgrid.ObjectGridManager;
import com.ibm.websphere.objectgrid.ObjectGrid;
import com.ibm.websphere.objectgrid.BackingMap;
import com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor;
import com.ibm.websphere.objectgrid.plugins.builtins.LFUEvictor;
ObjectGridManager ogManager = ObjectGridManagerFactory.getObjectGridManager();
```

```
ObjectGrid og = ogManager.createObjectGrid("grid");
BackingMap bm = og.defineMap( "map1" );
LRUEvictor evictor = new LRUEvictor();
evictor.setMaxSize(1000);
evictor.setSleepTime( 15 );
evictor.setNumberOfLRUQueues( 53 );
bm.setEvictor(evictor);
bm = og.defineMap( "map2" );
LFUEvictor evictor2 = new LFUEvictor();
evictor2.setMaxSize(2000);
evictor2.setSleepTime( 15 );
evictor2.setNumberOfHeaps( 211 );
bm.setEvictor(evictor2);
```

O snippet anterior mostra um evictor LRUEvictor sendo utilizado para o map1 BackingMap com um número aproximado de entradas de 53.000 (53 * 1000). O evictor LFUEvictor é utilizado para o map2 BackingMap com um número máximo aproximado de entradas de 422.000 (211 * 2000). Os evictores LRU e LFU têm uma propriedade de tempo de suspensão que indica por quanto tempo o evictor fica suspenso antes de ser ativado e verificar se as entradas precisam ser liberadas. O tempo de suspensão é especificado em segundos. Um valor de 15 segundos é uma boa garantia entre o impacto no desempenho e a prevenção para que o BackingMap não se torne muito grande. A meta é utilizar o maior tempo de suspensão possível sem fazer o BackingMap crescer a um tamanho excessivo.

O método setNumberOfLRUQueues configura a propriedade LRUEvictor que indica quantas filas de LRU o evictor utiliza para gerenciar informações de LRU. Uma coleta de filas é utilizada para que cada entrada não mantenha informações de LRU na mesma fila. Esta abordagem pode melhorar o desempenho minimizando o número de entradas do mapa que precisam ser sincronizadas no mesmo objeto de fila. Aumentar o número de filas é uma boa maneira de minimizar o impacto que o evictor LRU pode causar no desempenho. Um bom ponto de partida é utilizar dez por cento do número máximo de entradas como o número de filas. A utilização de um número primo geralmente é melhor do que utilizar um número que não seja primo. O método setMaxSize indica quantas entradas são permitidas em cada fila. Quando uma fila alcança seu número máximo entradas, a entrada ou as entradas utilizadas menos recentemente nesta fila são despejadas na próxima vez que o evictor verifica se há alguma entrada que precisa ser despejada.

O método setNumberOfHeaps configura a propriedade LFUEvictor para determinar quantos objetos de heap binários o LFUEvictor utiliza para gerenciar informações de LFU. Mais uma vez é utilizada uma coleta para melhorar o desempenho. A utilização de dez por cento do número máximo de entrada é um bom ponto de partida e utilizar um número primo geralmente é melhor do que utilizar um número que não seja primo. O método setMaxSize indica quantas entradas são permitidas em cada heap. Quando um heap alcança seu número máximo entradas, a entrada ou as entradas utilizadas menos recentemente neste heap são despejadas na próxima vez que o evictor verifica se há alguma entrada que precisa ser despejada.

Abordagem de Configuração XML para Conectar um Evictor Conectável

Em vez de utilizar várias APIs para conectar programaticamente um evictor e configurar suas propriedades, um arquivo XML pode ser utilizado para configurar cada BackingMap conforme ilustrado na amostra a seguir:

conectando um
evictor utilizando XML

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
<objectGrids>
    <objectGrid name="grid">
        <backingMap name="map1" ttlEvictorType="NONE" pluginCollectionRef="LRU" />
        <backingMap name="map2" ttlEvictorType="NONE" pluginCollectionRef="LFU" />
    </objectGrid>
</objectGrids>
<backingMapPluginCollections>
    <backingMapPlugincollection id="LRU">
        <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
            <property name="maxSize" type="int" value="1000" description="set max size for each LRU queue" />
            <property name="sleepTime" type="int" value="15" description="evictor thread sleep time" />
            <property name="numberOfLRUQueues" type="int" value="53" description="set number of LRU queues" />
        </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="LFU">
        <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LFUEvictor">
            <property name="maxSize" type="int" value="2000" description="set max size for each LFU heap" />
            <property name="sleepTime" type="int" value="15" description="evictor thread sleep time" />
            <property name="numberOfHeaps" type="int" value="211" description="set number of LFU heaps" />
        </bean>
    </backingMapPluginCollection>
</backingMapPluginCollections>
</objectGridConfig>
```

Despejo Baseado em Memória

Todos os evictors integrados suportam despejo baseado em memória que pode ser ativado na interface BackingMap configurando o atributo evictionTriggers de BackingMap como "MEMORY_USAGE_THRESHOLD". Para obter mais informações sobre como configurar o atributo evictionTriggers no BackingMap, consulte a interface BackingMap e a referência de configuração do eXtreme Scale.

O despejo baseado em memória é baseado no limite de uso do heap. Quando o despejo baseado em memória é ativado no BackingMap e o BackingMap possui algum evictor integrado, o limite de uso é configurado com uma porcentagem padrão de memória total se o limite ainda não tiver sido configurado anteriormente.

Para alterar a porcentagem de limite de uso, configure a propriedade memoryThresholdPercentage no contêiner e o arquivo de propriedades do servidor para o processo do servidor do eXtreme Scale. Para configurar o limite de uso de destino em um processo do cliente do eXtreme Scale, é possível utilizar o MemoryPoolMXBean. Consulte também: Arquivo containerServer.props e Iniciando Processos do Servidor eXtreme.

Durante o tempo de execução, se o uso da memória exceder o limite de uso destinado, os evictors baseados em memória iniciam o despejo de entradas e tentam manter o uso da memória abaixo do limite de uso destinado. Entretanto, não há garantia de que a velocidade do despejo seja rápida o suficiente para evitar um potencial erro de falta de memória se o tempo de execução do sistema continuar a consumir memória rapidamente.

Configurando uma Estratégia de Bloqueio

É possível definir uma estratégia otimista, pessimista ou de nenhum bloqueio em cada BackingMap na configuração do WebSphere eXtreme Scale.

Sobre Esta Tarefa

É possível especificar uma estratégia de bloqueio programaticamente ou com o XML. Para obter mais informações sobre o bloqueio, consulte as informações sobre as estratégias de bloqueio no *Visão Geral do Produto*.

Procedimento

- **Configure uma estratégia de bloqueio otimista**
  - Programaticamente

    Especificar a estratégia otimista programaticamente
    ```
     import com.ibm.websphere.objectgrid.BackingMap;
    import com.ibm.websphere.objectgrid.LockStrategy;
    import com.ibm.websphere.objectgrid.ObjectGrid;
    import com.ibm.websphere.objectgrid.ObjectGridManagerFactory;
    ...
    ObjectGrid og =
     ObjectGridManagerFactory.getObjectGridManager().createObjectGrid("test");
    BackingMap bm = og.defineMap("optimisticMap");
    bm.setLockStrategy( LockStrategy.OPTIMISTIC );
    ```
  - Usando XML

    Especificar estratégia otimista usando XML
    ```
    <?xml version="1.0" encoding="UTF-8"?>
    <objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
        xmlns="http://ibm.com/ws/objectgrid/config">
        <objectGrids>
            <objectGrid name="test">
                <backingMap name="optimisticMap"
                    lockStrategy="OPTIMISTIC"/>
            </objectGrid>
        </objectGrids>
    </objectGridConfig>
    ```
- **Configure uma estratégia de bloqueio pessimista**
  - Programaticamente

    especificar programaticamente estratégia pessimista
    ```
    import com.ibm.websphere.objectgrid.BackingMap;
    import com.ibm.websphere.objectgrid.LockStrategy;
    import com.ibm.websphere.objectgrid.ObjectGrid;
    import com.ibm.websphere.objectgrid.ObjectGridManagerFactory;
    ...
    ObjectGrid og =
     ObjectGridManagerFactory.getObjectGridManager().createObjectGrid("test");
    BackingMap bm = og.defineMap("pessimisticMap");
    bm.setLockStrategy( LockStrategy.PESSIMISTIC );
    ```
  - Usando XML

    Especificar estratégia pessimista usando XML
    ```
    <?xml version="1.0" encoding="UTF-8"?>
    <objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
        xmlns="http://ibm.com/ws/objectgrid/config">

        <objectGrids>
            <objectGrid name="test">
                <backingMap name="pessimisticMap"
                    lockStrategy="PESSIMISTIC"/>
            </objectGrid>
        </objectGrids>
    </objectGridConfig>
    ```
- **Configure uma estratégia de ausência de bloqueio**
  - Programaticamente

    Especificar uma estratégia de ausência de bloqueio programaticamente

    ```
    import com.ibm.websphere.objectgrid.BackingMap;
    import com.ibm.websphere.objectgrid.LockStrategy;
    import com.ibm.websphere.objectgrid.ObjectGrid;
    import com.ibm.websphere.objectgrid.ObjectGridManagerFactory;
    ...
    ```

```
ObjectGrid og =
 ObjectGridManagerFactory.getObjectGridManager().createObjectGrid("test");
BackingMap bm = og.defineMap("noLockingMap");
bm.setLockStrategy( LockStrategy.NONE );
```

– Usando XML

Especificar uma estratégia de ausência de bloqueio com XML

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
    xmlns="http://ibm.com/ws/objectgrid/config">

    <objectGrids>
        <objectGrid name="test">
            <backingMap name="noLockingMap"
                lockStrategy="NONE"/>
        </objectGrid>
    </objectGrids>
</objectGridConfig>
```

O que Fazer Depois

Para evitar uma exceção java.lang.IllegalStateException, é necessário chamar o método setLockStrategy antes de chamar os métodos initialize ou getSession na instância ObjectGrid.

Configurando Utilitários de Carga

A implementação de um utilitário de carga requer configuração para vários atributos.

Considerações sobre Pré-carregamento

Os utilitários de carga são plug-ins de mapa de apoio que são chamados quando são feitas alterações no mapa de apoio ou quando o mapa de apoio não pode atender a um pedido de dados (um erro de cache). Para obter uma visão geral de como o eXtreme Scale interage com um utilitário de carga, consulte as informações sobre cenários de armazenamento em cache sequencial no *Visão Geral do Produto*

Cada mapa de apoio tem um atributo preloadMode booleano que é configurado para indicar se o pré-carregamento de um mapa é executado assincronamente. Por padrão, o atributo preloadMode está configurado como false, o que indica que a inicialização do mapa de suporte não será concluída até que o pré-carregamento do mapa esteja concluído. Por exemplo, a inicialização do mapa de suporte não será concluída até que o método preloadMap seja retornado. Se o método preloadMap ler uma grande quantidade de dados no seu back end e carregá-los para o mapa, o tempo de conclusão desse procedimento pode ser relativamente longo. Neste caso, é possível configurar um mapa de suporte para utilizar o pré-carregamento assíncrono do mapa, configurando o atributo preloadMode como true. Essa configuração faz o código de inicialização do mapa de apoio iniciar um encadeamento que invoca o método preloadMap, permitindo que a inicialização de um mapa de apoio seja concluída enquanto o pré-carregamento do mapa ainda está em andamento.

Em um cenário eXtreme Scale distribuído, um dos padrões de pré-carregamento é o pré-carregamento de cliente. No pré-carregamento de cliente padrão, um cliente eXtreme Scale é responsável por recuperar dados do backend e inserir os dados no servidor eXtreme Scale distribuído utilizando agentes DataGrid. Além disso, o pré-carregamento de cliente poderia ser executado no método Loader.preloadMap em uma, e apenas uma, partição específica. Nesse caso, o carregamento assíncrono

de dados para a grade se tornaria muito importante. Se o pré-carregamento do cliente fosse executado no mesmo encadeamento, o mapa de apoio nunca seria inicializado, assim, a partição na qual ele reside nunca ficaria ON-LINE. Portanto, o cliente eXtreme Scale não poderia enviar o pedido para a partição e, eventualmente, isso causaria uma exceção.

Se um cliente do eXtreme Scale for utilizado no método preloadMap, você deverá configurar o atributo preloadMode como true. A alternativa é iniciar um encadeamento no código de pré-carregamento do cliente.

O trecho de código a seguir ilustra como o atributo preloadMode é configurado para ativar o pré-carregamento assíncrono:

```
BackingMap bm = og.defineMap( "map1" );
bm.setPreloadMode( true );
```

O atributo preloadMode também pode ser configurado utilizando um arquivo XML conforme ilustrado no seguinte exemplo:

```
<backingMap name="map1" preloadMode="true" pluginCollectionRef="map1"
 lockStrategy="OPTIMISTIC" />
```

TxID e Utilização da Interface TransactionCallback

O método get e os métodos batchUpdate da interface do Utilitário de Carga são transmitidos para um objeto TxID que representa a transação do objeto Session que requer que a operação get ou batchUpdate seja executada. É possível que os métodos get e batchUpdate sejam chamados mais de uma vez por transação. Portanto, os objetos com escopo definido pela transação requeridos pelo Loader geralmente são mantidos em um slot do objeto TxID. Um utilitário de carga JDBC (Java Database Connectivity) é usado para ilustrar como um utilitário de carga usa as interfaces TxID e TransactionCallback.

Também é possível que vários mapas do ObjectGrid sejam armazenados no mesmo banco de dados. Cada mapa possui seu próprio Loader e cada Loader pode precisar conectar-se ao mesmo banco de dados. Ao conectar-se ao mesmo banco de dados, cada Loader deseja utilizar a mesma conexão JDBC para que as alterações em cada tabela sejam confirmadas como parte da transação do mesmo banco de dados. Geralmente, a mesma pessoa que grava a implementação do Loader também grava a implementação do TransactionCallback. O melhor método é quando a interface TransactionCallback é estendida de modo a incluir os métodos que Utilitário de Carga precisa para obter uma conexão com o banco de dados e para armazenar em cache as instruções preparadas. O motivo para esta metodologia torna-se aparente à medida que você visualiza como as interfaces TransactionCallback e TxID são utilizadas pelo utilitário de carga.

Como um exemplo, o utilitário de carga pode precisar da interface TransactionCallback para ser estendido conforme a seguir:

```
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import com.ibm.websphere.objectgrid.TxID;
public interface MyTransactionCallback extends TransactionCallback
{
    Connection getAutoCommitConnection(TxID tx, String databaseName) throws SQLException;
    Connection getConnection(TxID tx, String databaseName, int isolationLevel ) throws SQLException;
    PreparedStatement getPreparedStatement(TxID tx, Connection conn, String tableName, String sql)
   throws SQLException;
    Collection getPreparedStatementCollection( TxID tx, Connection conn,  String tableName );
}
```

Com tais novos métodos, os métodos get e batchUpdate do Utilitário de Carga podem obter uma conexão da seguinte forma:

```
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import com.ibm.websphere.objectgrid.TxID;
private Connection getConnection(TxID tx, int isolationLevel)
{
    Connection conn = ivTcb.getConnection(tx, databaseName, isolationLevel );
    return conn;
}
```

No exemplo anterior e nos exemplos que seguem, vTcb e ivOcb são variáveis da instância do Carregador que foram inicializadas conforme descrito na seção Considerações de Pré-carregamento. A variável ivTcb é uma referência à instância MyTransactionCallback e ivOcb é uma referência à instância MyOptimisticCallback. A variável databaseName é uma variável da instância do Utilitário de Carga que foi configurada como uma propriedade do Utilitário de Carga durante a inicialização do mapa de suporte. O argumento isolationLevel é uma das constantes da Conexão JDBC que estão definidas para os diversos níveis de isolamento suportados pelo JDBC. Se o Utilitário de Carga estiver utilizando uma implementação otimista, o método get geralmente utilizará uma conexão de autoconfirmação JDBC para buscar os dados do banco de dados. Nesse caso, o Utilitário de Carga pode ter um método getAutoCommitConnection que seja implementado da seguinte forma:

```
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import com.ibm.websphere.objectgrid.TxID;
private Connection getAutoCommitConnection(TxID tx)
{
    Connection conn = ivTcb.getAutoCommitConnection(tx, databaseName);
    return conn;
}
```

Lembre-se de que o método batchUpdate possui a seguinte instrução switch:

```
switch ( logElement.getType().getCode() )
{
    case LogElement.CODE_INSERT:
        buildBatchSQLInsert( tx, key, value, conn );
        break;
    case LogElement.CODE_UPDATE:
        buildBatchSQLUpdate( tx, key, value, conn );
        break;
    case LogElement.CODE_DELETE:
        buildBatchSQLDelete( tx, key, conn );
        break;
}
```

Cada um dos métodos buildBatchSQL utiliza a interface MyTransactionCallback para obter uma instrução preparada. Este é um trecho de código que mostra o método buildBatchSQLUpdate construindo uma instrução SQL update para atualizar uma entrada EmployeeRecord e incluindo-a na atualização de batch:

```
private void buildBatchSQLUpdate( TxID tx, Object key, Object value, Connection conn )
throws SQLException, LoaderException
{
    String sql = "update EMPLOYEE set LASTNAME = ?, FIRSTNAME = ?, DEPTNO = ?,
    SEQNO = ?, MGRNO = ? where EMPNO = ?";
    PreparedStatement sqlUpdate = ivTcb.getPreparedStatement( tx, conn,
   "employee", sql );
    EmployeeRecord emp = (EmployeeRecord) value;
```

```
    sqlUpdate.setString(1, emp.getLastName());
    sqlUpdate.setString(2, emp.getFirstName());
    sqlUpdate.setString(3, emp.getDepartmentName());
    sqlUpdate.setLong(4, emp.getSequenceNumber());
    sqlUpdate.setInt(5, emp.getManagerNumber());
    sqlUpdate.setInt(6, key);
    sqlUpdate.addBatch();
}
```

Quando o loop batchUpdate tiver construído todas as instruções preparadas, ele chamará o método getPreparedStatementCollection. Esse método é implementado como segue:

```
private Collection getPreparedStatementCollection( TxID tx, Connection conn )
{
    return ( ivTcb.getPreparedStatementCollection( tx, conn, "employee" ) );
}
```

Quando o aplicativo chama o método commit no objeto Session, o código do Session chamará o método commit, no método TransactionCallback, depois de enviar todas as alterações feitas pela transação fora do Utilitário de Carga para cada mapa alterado pela transação. Como todos os Loaders utilizaram o método MyTransactionCallback para obter todas as conexões e instruções preparadas necessárias, o método TransactionCallback sabe qual conexão utilizar para solicitar que o back end confirme as alterações. Portanto, estender a interface TransactionCallback com métodos requeridos por cada um dos Loaders tem as seguintes vantagens:

- O objeto TransactionCallback encapsula a utilização de slots TxID para dados com escopo definido pela transação e o Loader não requer informações sobre os slots TxID. O Loader apenas precisa saber sobre os métodos que foram incluídos no TransactionCallback utilizando a interface MyTransactionCallback para as funções de suporte requeridas pelo Loader.
- O objeto TransactionCallback pode assegurar que o compartilhamento da conexão ocorra entre cada Loader que se conecta ao mesmo backend para que um protocolo de confirmação de duas fases seja evitado.
- O objeto TransactionCallback pode assegurar que a conexão com o backend seja orientada para conclusão por meio de uma confirmação ou rollback chamado na conexão quando apropriado.
- O TransactionCallback garante a limpeza dos recursos do banco de dados após a conclusão de uma transação.
- O TransactionCallback fica oculto se ele estiver obtendo uma conexão gerenciada a partir de um ambiente gerenciado como WebSphere Application Server ou algum outro servidor de aplicativos compatível com Java 2 Platform, Enterprise Edition (J2EE). Esta vantagem permite que o mesmo código do Loader seja utilizado em ambientes gerenciados e não gerenciados. Apenas o plug-in TransactionCallback deve ser alterado.
- Para obter informações detalhadas sobre como a implementação do TransactionCallback utiliza os slots TxID para dados dentro do escopo da transação, consulte Plug-in do TransactionCallback.

OptimisticCallback

Conforme mencionado anteriormente, o Utilitário de Carga pode utilizar uma abordagem otimista para controle de simultaneidade. Nesse caso, o exemplo do método buildBatchSQLUpdate precisará ser modificado ligeiramente para implementar uma abordagem otimista. Existem várias maneiras possíveis para utilizar uma abordagem otimista. Uma maneira típica é ter uma coluna de time

stamp ou uma coluna do contador de números de seqüência para o controle de versões de cada atualização da linha. Suponha que a tabela de funcionários tenha uma coluna de números de seqüência que aumenta sempre que a linha é atualizada. Em seguida, você modifica a assinatura do método buildBatchSQLUpdate para que ela seja transmitida para o objeto LogElement em vez do par chave e valor. Ele também precisa utilizar o objeto OptimisticCallback que está conectado ao mapa de suporte para obter o objeto da versão inicial e para atualizar o objeto da versão. Este é um exemplo de método buildBatchSQLUpdate modificado que utiliza a variável da instância ivOcb inicializada conforme descrito na seção preloadMap:

exemplo de código do método batch-update modificado

```
private void buildBatchSQLUpdate( TxID tx, LogElement le, Connection conn )
 throws SQLException, LoaderException
{
    // Get the initial version object when this map entry was last read
    // or updated in the database.
    Employee emp = (Employee) le.getCurrentValue();
    long initialVersion = ((Long) le.getVersionedValue()).longValue();
    // Get the version object from the updated Employee for the SQL update
    //operation.
    Long currentVersion = (Long)ivOcb.getVersionedObjectForValue( emp );
    long nextVersion = currentVersion.longValue();
    // Now build SQL update that includes the version object in where clause
    // for optimistic checking.
    String sql = "update EMPLOYEE set LASTNAME = ?, FIRSTNAME = ?,
    DEPTNO = ?,SEQNO = ?, MGRNO = ? where EMPNO = ? and SEQNO = ?";
    PreparedStatement sqlUpdate = ivTcb.getPreparedStatement( tx, conn,
   "employee", sql );
    sqlUpdate.setString(1, emp.getLastName());
    sqlUpdate.setString(2, emp.getFirstName());
    sqlUpdate.setString(3, emp.getDepartmentName());
    sqlUpdate.setLong(4, nextVersion );
    sqlUpdate.setInt(5, emp.getManagerNumber());
    sqlUpdate.setInt(6, key);
    sqlUpdate.setLong(7, initialVersion);
    sqlUpdate.addBatch();
}
```

O exemplo mostra que o LogElement é utilizado para obter o valor de versão inicial. Quando a transação acessa pela primeira vez a entrada do mapa, é criado um LogElement com o objeto Employee inicial obtido do mapa. O objeto Employee inicial também é transmitido para o método getVersionedObjectForValue na interface OptimisticCallback e o resultado é salvo no LogElement. Este processamento ocorre antes de um aplicativo receber uma referência ao objeto Employee inicial e de chamar algum método que altere o estado do objeto Employee inicial.

O exemplo mostra que o Loader utiliza o método getVersiondObjectForValue para obter o objeto de versão para o objeto Employee atual atualizado. Antes de chamar o método batchUpdate na interface do utilitário de carga, eXtreme Scale chama o método updateVersionedObjectForValue na interface OptimisticCallback para gerar um novo objeto de versão a ser gerado para o objeto Employee atualizado. Quando o método batchUpdate retornar ao ObjectGrid, o LogElement será atualizado com o objeto de versão atual e se tornará o novo objeto de versão inicial. Esta etapa é necessária porque o aplicativo pode ter chamado o método flush no mapa em vez do método commit na Session. É possível que o Loader seja chamado várias vezes por uma única transação para a mesma chave. Por este motivo, o eXtreme Scale assegura que o LogElement seja atualizado com o novo objeto de versão toda vez que a linha for atualizada na tabela de funcionários.

Agora que o Utilitário de Carga tem o objeto de versão inicial e o próximo objeto de versão, ele pode executar uma instrução SQL update que configura a coluna SEQNO para o próximo valor do objeto de versão e utiliza o valor do objeto de versão inicial na cláusula where. Essa abordagem, às vezes, é referida como uma instrução update super qualificada. A utilização da instrução update super qualificada permite que o banco de dados relacional verifique se a linha não foi alterada por alguma outra transação entre o tempo de leitura do banco de dados por parte da transação e o momento em que esta o atualizou. Se outra transação modificou a linha, a matriz de contagem retornada pela atualização de batch indica que zero linhas foram atualizadas para esta chave. O Utilitário de Carga é responsável por verificar se a operação SQL update atualizou a linha. Se isso não ocorreu, ele exibirá uma exceção com.ibm.websphere.objectgrid.plugins.OptimisticCollisionException para informar o objeto Session que o método batchUpdate falhou porque mais de uma transação simultânea está tentando atualizar a mesma linha na tabela de banco de dados. Esta exceção faz a Session efetuar rollback e o aplicativo deve tentar novamente a transação inteira. O fundamento lógico é que a nova tentativa será bem-sucedida, motivo pelo qual esta abordagem é chamada de otimista. A abordagem otimista apresenta um desempenho melhor quando os dados não sofrem alterações freqüentes ou transações simultâneas raramente tentam atualizar a mesma linha.

É importante que o Utilitário de Carga utilize o parâmetro key do construtor OptimisticCollisionException para identificar qual chave ou conjunto de chaves causou a falha do método batchUpdate otimista. O parâmetro de chave pode ser o próprio objeto de chave ou uma matriz de objetos de chave se mais de uma chave resultar em uma falha de atualização otimista. E o eXtreme Scale usa o método getKey do construtor OptimisticCollisionException para determinar quais entradas de mapa contêm dados desatualizados e causaram a exceção. Parte do processamento de rollback é liberar cada entrada do mapa stale do mapa. A liberação de entradas stale é necessária para que qualquer transação subseqüente que acessa a mesma chave ou chaves resulte no método get da interface do Loader que está sendo chamada para atualizar as entradas do mapa com os dados atuais do banco de dados.

Outras maneiras para um Loader implementar uma abordagem otimista incluem:

- Não existe nenhuma coluna de time stamp ou de número de seqüência. Neste caso, o método getVersionObjectForValue na interface OptimisticCallback apenas retorna o próprio objeto de valor como a versão. Com essa abordagem, o Utilitário de Carga precisa construir uma cláusula WHERE que inclua cada um dos campos do objeto de versão inicial. Essa abordagem não é eficiente e nem todos os tipos de colunas estão qualificados para serem utilizados na cláusula WHERE de uma instrução SQL update super qualificada. Esta abordagem geralmente não é utilizada.
- Não existe nenhuma coluna de time stamp ou de número de seqüência. No entanto, diferente da abordagem anterior, a cláusula WHERE contém apenas os campos de valores modificados pela transação. Um método para detectar quais campos foram modificados é configurar o modo de cópia no mapa de suporte como CopyMode.COPY_ON_WRITE. Esse modo de cópia requer que uma interface de valor seja transmitida para o método setCopyMode na interface BackingMap. O BackingMap cria objetos de proxy dinâmicos que implementam a interface de valor fornecida. Com este modo de cópia, o Loader pode lançar cada valor em um objeto com.ibm.websphere.objectgrid.plugins.ValueProxyInfo. A interface ValueProxyInfo possui um método que permite que o Loader obtenha a Lista de nomes de atributos que foram alterados pela transação. Esse método permite que o Utilitário de Carga chame os métodos get na interface de

valor para os nomes de atributos a fim de obter os dados alterados e construa uma instrução SQL update que configure apenas os atributos alterados. A cláusula WHERE agora pode ser construída de modo a ter a coluna-chave primária e cada uma das colunas de atributos alteradas. Esta abordagem é mais eficiente do que a abordagem anterior, mas requer que mais código seja gravado no Loader e gera a possibilidade de que o cache de instrução preparado precise ser maior para manipular as diferentes permutações. No entanto, se as transações geralmente modificarem apenas alguns dos atributos, esta limitação pode não ser um problema.

- Alguns bancos de dados relacionais podem ter uma API para ajudar na manutenção automática de dados da coluna que são úteis para o controle de versões otimista. Consulte a documentação do banco de dados para determinar se existe esta possibilidade.

Configurando o Suporte do Carregador Write-behind

É possível ativar o suporte write-behind usando o arquivo XML descritor do ObjectGrid ou programaticamente usando a interface BackingMap.

Use o arquivo XML descritor do ObjectGrid para ativar o suporte write-behind ou programaticamente usando a interface BackingMap.

Arquivo XML descritor do ObjectGrid

Ao configurar um ObjectGrid usando o arquivo XML descritor do ObjectGrid, o utilitário de carga write-behind é ativado configurando-se o atributo writeBehind na tag backingMap. Este é um exemplo:

```
<objectGrid name="library" >
     <backingMap name="book" writeBehind="T300;C900" pluginCollectionRef="bookPlugins"/>
```

No exemplo anterior, o suporte para write-behind do mapa de apoio book está ativado com o parâmetro T300;C900. O atributo write-behind especifica a duração da atualização máxima e/ou uma contagem máxima de atualização de chave. O formato do parâmetro write-behind é:

```
::= <defaults> | <update time> | <update key count> | <update time> ";"
<update key count>::= "T" <positive integer>::= "C" <positive integer>::= ""
```

- atributo write-behind
- duração da atualização
- quantidade de chaves atualizadas
- defaults

Ocorrem atualizações no utilitário de carga quando um dos seguintes eventos ocorre:

1. O tempo máximo de atualização em segundos decorreu desde a última atualização.
2. O número de chaves atualizadas no mapa de fila alcançou a contagem de chaves de atualização.

Estes parâmetros são apenas dicas. A contagem de atualização real e a duração da atualização estarão dentro do intervalo próximo dos parâmetros. Entretanto, não garantimos que a contagem de atualização real ou a duração da atualização sejam as mesmas que as definidas nos parâmetros. Além disso, a primeira atualização behind pode ocorrer após até o dobro da duração da atualização. Isto ocorre porque ObjectGrid escolhe aleatoriamente o momento de início da atualização para que todas as partições não cheguem no banco de dados simultaneamente.

No exemplo anterior T300;C900, o utilitário de carga grava os dados no backend quando 300 segundos decorreram desde a última atualização ou quando 900 chaves estão pendentes para serem atualizadas. A duração da atualização padrão é 300 segundos e a contagem de chaves de atualização padrão é de 1000.

Armazenamento em Cache Write-behind

É possível utilizar armazenamento em cache write-behind para reduzir o gasto adicional que ocorre durante a atualização de um banco de dados que você está utilizando como back end.

Apresentação

O armazenamento em cache write-behind enfileira assincronamente as atualizações no plug-in do Utilitário de Carga. É possível melhorar o desempenho desconectando atualizações, inserções e remoções para um mapa, a sobrecarga de atualização do banco de dados de backend. A atualização assíncrona é executada após um atraso baseado em tempo (por exemplo, cinco minutos) ou um atraso baseado em entradas (1000 entradas).

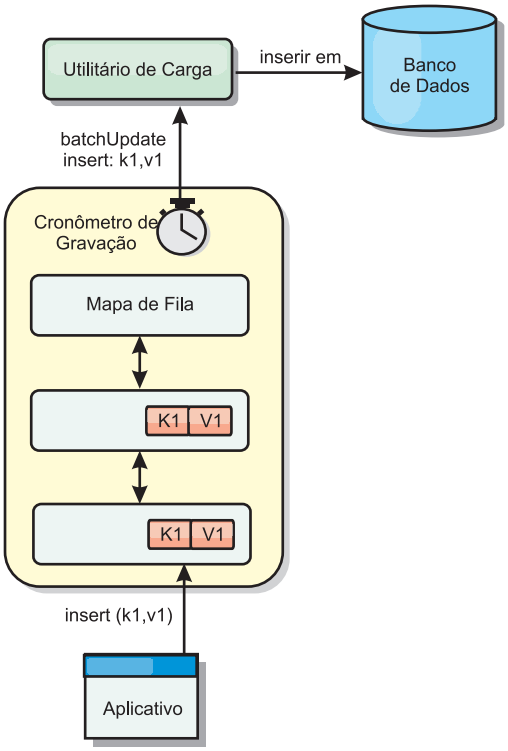


Figura 7. Armazenamento em Cache Write-behind

A configuração write-behind em um BackingMap cria um encadeamento entre o utilitário de carga e o mapa. O utilitário de carga então delega pedidos de dados por meio do encadeamento de acordo com as definições da configuração no método BackingMap.setWriteBehind. Quando uma transação do eXtreme Scale insere, atualiza ou remove uma entrada de um mapa, um objeto LogElement é criado para cada um destes registros. Estes elementos são enviados para o utilitário de carga write-behind e enfileirados em um ObjectMap especial denominado mapa de fila. Cada mapa de apoio com a configuração write-behind ativada possui seus próprios mapas de fila. Um encadeamento write-behind remove periodicamente os dados enfileirados dos mapas de fila e executa o push deles para o utilitário de carga de backend real.

O utilitário de carga write-behind enviará apenas os tipos insert, update e delete dos objetos LogElement para o utilitário de carga real. Todos os outros tipos de objetos LogElement, por exemplo, o tipo EVICT, são ignorados.

Benefícios

Ativar o suporte write-behind possui os seguintes benefícios:

- **Isolamento de falha de backend:** O armazenamento em cache write-behind fornece uma camada de isolamento das falhas de backend. Quando o banco de dados de backend falha, as atualizações são enfileiradas no mapa de fila. Os aplicativos podem continuar a conduzir transações para o eXtreme Scale. Quando o backend se recupera, os dados no mapa de fila são enviados para o backend.
- **Carga de backend reduzida:** O utilitário de carga write-behind mescla as atualizações em uma base de chave, portanto, apenas uma atualização mesclada por chave existe no mapa de fila. Esta mesclagem diminui o número de atualizações no backend.
- **Desempenho de transação aprimorado:** Tempos de transação do eXtreme Scale individuais são reduzidos porque a transação não precisa aguardar até que os dados sejam sincronizados com o backend.

Considerações de Design do Aplicativo

Ativar o suporte write-behind é simples, mas o design de um aplicativo para trabalhar com o suporte write-behind precisa de consideração cuidadosa. Sem o suporte de write-behind, a transação de ObjectGrid engloba a transação de backend. A transação do ObjectGrid inicia antes da transação de backend iniciar e termina após a transação de backend terminar.

Com suporte write-behind ativado, a transação do ObjectGrid é concluída antes que a transação de backend inicie. A transação do ObjectGrid e a transação de backend não estão acopladas.

Limitadores de Integridade Referencial

Cada mapa de apoio que é configurado com suporte write-behind possui seu próprio encadeamento write-behind para enviar os dados para o backend. Portanto, os dados que são atualizados em diferentes mapas em uma transação do ObjectGrid são atualizadas no backend em diferentes transações de backend. Por exemplo, a transação T1 atualiza a chave key1 no mapa Map1 e a chave key2 no mapa Map2. A atualização da key1 para o mapa Map1 é atualizada no backend em uma transação de backend e a key2 atualizada para o mapa Map2 é atualizada no backend em outra transação de backend por encadeamentos write-behind diferentes. Se os dados armazenados no Map1 e Map2 possuírem relações, tais como limitadores de chave estrangeira no backend, as atualizações podem falhar.

Ao projetar os limitadores de integridade referencial em seu banco de dados de backend, certifique-se de que atualizações fora de ordem sejam permitidas.

Comportamento do Bloqueio de Mapa de Fila

Outra grande diferença de comportamento da transação é o comportamento do bloqueio. O ObjectGrid suporta três diferentes estratégias de bloqueio: PESSIMISTIC, OPTIMISITIC e NONE. Os mapas de fila write-behind utilizam a estratégia de bloqueio pessimista não importando qual estratégia de bloqueio está

configurada para seu mapa de apoio. Existem dois diferentes tipos de operações que adquirem um bloqueio no mapa de fila:

- Quando uma transação do ObjectGrid é confirmada ou um flush (flush de mapa ou flush de sessão) acontece, a transação lê a chave no mapa de fila e coloca um bloqueio S na chave.
- Quando uma transação do ObjectGrid é confirmada, a transação tenta atualizar o bloqueio S para o bloqueio X na chave.

Devido a este comportamento do mapa de fila extra, é possível visualizar algumas diferenças de comportamento de bloqueio.

- Se o mapa do usuário for configurado como a estratégia de bloqueio PESSIMISTIC, não há muita diferença no comportamento de bloqueio. Sempre que um flush ou commit é chamado, um bloqueio S é colocado na mesma chave no mapa de fila. Durante o momento do commit, um bloqueio X não é adquirido apenas para a chave no mapa do usuário, ele também é adquirido para a chave no mapa de fila.
- Se o mapa do usuário for configurado com a estratégia de bloqueio OPTIMISTIC ou NONE, a transação do usuário seguirá o padrão de estratégia de bloqueio PESSIMISTIC. Sempre que um flush ou commit é chamado, um bloqueio S é adquirido para a mesma chave no mapa de fila. Durante o momento do commit, um bloqueio X é adquirido para a chave no mapa de fila utilizando a mesma transação.

Novas Tentativas de Transações do Utilitário de Carga

O ObjectGrid não suporta transações 2-phase ou XA. O encadeamento write-behind remove registros do mapa de fila e atualiza os registros no backend. Se o servidor falhar no meio da transação, algumas atualizações de backend podem ser perdidas.

O utilitário de carga write-behind automaticamente tentará gravar novamente transações falhas e enviará uma LogSequence duvidosa para o backend para evitar a perda de dados. Esta ação requer que o utilitário de carga seja idempotente, o que significa que o Loader.batchUpdate(TxId, LogSequence) é chamado duas vezes com o mesmo valor, ele fornece o mesmo resultado como se tivesse sido aplicado uma vez. As implementações do utilitário de carga devem implementar a interface RetryableLoader para ativar este recurso. Consulte a documentação da API para obter mais detalhes.

Falha do Utilitário de Carga

O plug-in do utilitário de carga pode falhar quando não consegue se comunicar com o back end do banco de dados. Isto pode acontecer se o servidor de banco de dados ou a conexão de rede estiver inativa. O utilitário de carga write-behind irá enfileirar as atualizações e tentará executar o push das alterações de dados para o utilitário de carga periodicamente. O utilitário de carga deve notificar o tempo de execução do ObjectGrid que há um problema de conectividade do banco de dados lançando uma exceção LoaderNotAvailableException.

Portanto, a implementação do Utilitário de Carga deve poder distinguir uma falha de dados ou uma falha de utilitário de carga físico. A falha de dados deve ser lançada ou relançada como uma LoaderException ou uma OptimisticCollisionException, mas uma falha de utilitário de carga físico deve ser lançada ou relançada como uma LoaderNotAvailableException. O ObjectGrid manipula estas exceções de maneira diferente:

- Se uma LoaderException for capturada pelo utilitário de carga write-behind, o utilitário de carga write-behind a considerará falha devido a alguma falha de dados, tal como uma falha de chave duplicada. O utilitário de carga write-behind irá remover a atualização do lote e tentará atualizar um registro em um momento para isolar a falha de dados. Se uma {{LoaderException}} for capturada durante uma atualização de registro, um registro de atualização falho é criado e registrado no mapa de atualização falho.
- Se uma LoaderNotAvailableException for capturada pelo utilitário de carga write-behind, o utilitário de carga write-behind a considerará falha porque não pode se conectar ao final do banco de dados, por exemplo, porque o backend do banco de dados estiver inativo, uma conexão com o banco de dados não estiver disponível ou a rede estiver inativa. O utilitário de carga write-behind aguardará por 15 segundo e, em seguida, tentará novamente executar uma atualização de lote no banco de dados.

O erro comum é lançar uma LoaderException enquanto uma LoaderNotAvailableException deve ser lançada. Todos os registros enfileirados no utilitário de carga write-behind se tornarão atualizações de registro falhas, o que frustra o propósito do isolamento de falha do backend.

Considerações sobre Desempenho

O suporte ao armazenamento em cache write-behind aumenta o tempo de resposta removendo a atualização do utilitário de carga da transação. Ele também aumenta o rendimento do banco de dados já que as atualizações de banco de dados são combinadas. É importante compreender o gasto adicional introduzido pelo encadeamento write-behind, que executa o pull dos dados da mapa de fila e executa o push para o utilitário de carga.

A contagem máxima de atualização ou o tempo máximo de atualização necessário a ser ajustado com base nos padrões de uso e no ambiente esperados. Se o valor da contagem máxima de atualização ou o tempo máximo de atualização for muito pequeno, o gasto adicional do encadeamento write-behind pode exceder os benefícios. Configurar um valor maior para estes dois parâmetros também pode aumentar o uso da memória para enfileirar os dados e aumentar o tempo de envelhecimento dos registros do banco de dados.

Para obter um melhor desempenho, ajuste os parâmetros write-behind com base nos seguintes fatores:

- Proporção de transações de leitura e gravação
- Mesma frequência de atualização de registro
- Latência de atualização de banco de dados.

Suporte de Armazenamento em Cache Write-behind

É possível usar o armazenamento em cache write-behind para reduzir o gasto adicional que ocorre na atualização de um banco de dados backend. As filas de armazenamento em cache write-behind são atualizadas para o plug-in do utilitário de carga.

Apresentação

O armazenamento em cache write-behind enfileira assincronamente as atualizações no plug-in do Utilitário de Carga. É possível melhorar o desempenho desconectando atualizações, inserções e remoções para um mapa, a sobrecarga de

atualização do banco de dados de backend. A atualização assíncrona é executada após um atraso baseado em tempo (por exemplo, cinco minutos) ou um atraso baseado em entradas (1000 entradas).

Ao definir a configuração write-behind em um mapa de apoio, um encadeamento write-behind é criado e oculta o utilitário de carga configurado. Quando uma transação do eXtreme Scale insere, atualiza ou remove uma entrada do mapa do eXtreme Scale, um objeto LogElement é criado para cada um destes registros. Estes elementos são enviados para o utilitário de carga write-behind e enfileirados em um ObjectMap especial denominado mapa de fila. Cada mapa de apoio com a configuração write-behind ativada possui seus próprios mapas de fila. Um encadeamento write-behind remove periodicamente os dados enfileirados dos mapas de fila e executa o push deles para o utilitário de carga de backend real.

O utilitário de carga write-behind enviará apenas os tipos insert, update e delete dos objetos LogElement para o utilitário de carga real. Todos os outros tipos de objetos LogElement, por exemplo, o tipo EVICT, são ignorados.

O suporte write-behind *é* uma extensão do plug-in do Carregador, que você usa para integrar o eXtreme Scale ao banco de dados. Por exemplo, consulte as informações do "Configurando Utilitários de Carga do JPA" na página 230 sobre como configurar um carregador JPA.

Benefícios

Ativar o suporte write-behind possui os seguintes benefícios:

- Isolamento de falha do backend: O armazenamento em cache write-behind fornece uma camada de isolamento a partir de falhas backend. Quando o banco de dados de backend falha, as atualizações são enfileiradas no mapa de fila. Os aplicativos podem continuar a conduzir transações para o eXtreme Scale. Quando o backend se recupera, os dados no mapa de fila são enviados para o backend.
- Carga de backend reduzida: O utilitário de carga write-behind mescla as atualizações em uma base de chave, portanto, apenas uma atualização mesclada por chave existe no mapa de fila. Esta mesclagem diminui o número de atualizações no backend.
- Desempenho de transação aprimorado: Tempos de transação do eXtreme Scale individuais são reduzidos porque a transação não precisa aguardar até que os dados sejam sincronizados com o backend.

XML Descritor do ObjectGrid

Ao configurar um eXtreme Scale utilizando um arquivo XML descritor doeXtreme Scale, o utilitário de carga write-behind é ativado configurando o atributo writeBehind na tag backingMap. Este é um exemplo:

```
<objectGrid name="library" >
     <backingMap name="book" writeBehind="T300;C900" pluginCollectionRef="bookPlugins"/>
```

No exemplo anterior, o suporte write-behind do mapa de apoio "book" é ativado com o parâmetro "T300;C900".

O atributo write-behind especifica a duração da atualização máxima e/ou uma contagem máxima de atualização de chave. O formato do parâmetro write-behind é:

```
write-behind attribute  ::= <defaults> | <update time> | <update key count> | <update time> ";" <update key count>
update time  ::= "T" <positive integer>
update key count  ::= "C" <positive integer>
defaults  ::= "" {table}
```

Ocorrem atualizações no utilitário de carga quando um dos seguintes eventos ocorre:

1. O tempo máximo de atualização em segundos decorreu desde a última atualização.
2. O número de chaves atualizadas no mapa de fila alcançou a contagem de chaves de atualização.

Estes parâmetros são apenas dicas. A contagem de atualização real e a duração da atualização estarão dentro do intervalo próximo dos parâmetros. Entretanto, não é garantido que a contagem de atualização real ou a duração da atualização sejam as mesmas que as definidas nos parâmetros. Além disso, a primeira atualização behind pode ocorrer após até o dobro da duração da atualização. Isto é porque o eXtreme Scale escolhe a esmo o horário de início para que todas as partições não acessem o banco de dados simultaneamente.

No exemplo anterior T300;C900, o utilitário de carga grava os dados no backend quando decorrem 300 segundos da última atualização ou quando 900 chaves estão pendentes de atualização.

A duração da atualização padrão é 300 segundos e a contagem de chaves de atualização padrão é de 1000.

A seguinte tabela lista alguns exemplos do atributo write-behind.

Nota: Se você configurar o utilitário de carga write-behind como uma cadeia vazia: `writeBehind=""`, o utilitário de carga write-behind é ativado usando os valores padrão. Portanto, não especifique o atributo writeBehind se não desejar que o suporte write-behind seja ativado.

Tabela 8. Algumas Opções de write-behind

Valor do Atributo	Tempo
T100	A duração da atualização é 100 segundos e a contagem de chaves de atualização é de 1000 (o valor padrão)
C2000	A duração da atualização é de 300 segundos (o valor padrão) e a contagem de chaves de atualização é de 2000.
T300;C900	A duração da atualização é de 300 segundos e a contagem de chaves de atualização é de 900.
""	A duração da atualização é de 300 segundos (o valor padrão) e a contagem de chaves de atualização é de 1000 (o valor padrão).

Ativando o Suporte Write-behind Programaticamente

Quando você cria um mapa de apoio programaticamente para um eXtreme Scale em memória local, é possível usar o método a seguir na interface BackingMap para ativar e desativar o suporte para write-behind.

```
public void setWriteBehind(String writeBehindParam);
```

Para obter mais detalhes sobre como usar o método setWriteBehind, consulte as informações sobre a interface BackingMap no *Guia de Programação*.

Considerações de Design do Aplicativo

Ativar o suporte write-behind é simples, mas o design de um aplicativo para trabalhar com o suporte write-behind precisa de consideração cuidadosa. Sem o

suporte write-behind, a transação do eXtreme Scale engloba a transação de backend. A transação do eXtreme Scale inicia antes do início da transação de backend e termina após a transação de backend terminar.

Com o suporte write-behind ativado, a transação do eXtreme Scale termina antes que a transação de backend inicie. A transação do eXtreme Scale e a transação de backend não estão acopladas.

Limitadores de Integridade Referencial

Cada mapa de apoio que é configurado com suporte write-behind possui seu próprio encadeamento write-behind para enviar os dados para o backend. Portanto, os dados que são atualizados em mapas diferentes em uma transação do eXtreme Scale são atualizados no backend em diferentes transações de backend. Por exemplo, a transação T1 atualiza a chave key1 no mapa Map1 e a chave key2 no mapa Map2. A atualização da key1 para o mapa Map1 é atualizada no backend em uma transação de backend e a key2 atualizada para o mapa Map2 é atualizada no backend em outra transação de backend por encadeamentos write-behind diferentes. Se os dados armazenados no Map1 e Map2 possuírem relações, tais como limitadores de chave estrangeira no backend, as atualizações podem falhar.

Ao projetar os limitadores de integridade referencial em seu banco de dados de backend, certifique-se de que atualizações fora de ordem sejam permitidas.

Atualizações Falhas

Como a transação do eXtreme Scale termina antes de a transação de backend iniciar, é possível que ocorra um falso sucesso da transação. Por exemplo, se você tentar inserir uma entrada em uma transação do eXtreme Scale que não exista no mapa de apoio mais que exista no backend, causando uma chave duplicada, a transação do eXtreme Scale não será bem-sucedida. Entretanto, a transação na qual o encadeamento write-behind insere tal objeto no backend falha com uma exceção de chave duplicada.

Consulte o “Manipulando Atualizações Write-Behind Falhas” na página 129 para saber como manipular essas falhas.

Comportamento do Bloqueio de Mapa de Fila

Outra grande diferença de comportamento da transação é o comportamento de bloqueio. O eXtreme Scale suporta três diferentes estratégias de bloqueio: pessimistic, optimistic e none. Os mapas de fila write-behind utilizam a estratégia de bloqueio pessimista não importando qual estratégia de bloqueio está configurada para seu mapa de apoio. Existem dois diferentes tipos de operações que adquirem um bloqueio no mapa de fila:

- Quando ocorre o commit de uma transação do eXtreme Scale ou acontece um flush (flush de mapa ou flush de sessão), a transação lê a chave no mapa de fila e coloca um bloqueio S na chave.
- Quando ocorre um commit na transação do eXtreme Scale, a transação tenta atualizar o bloqueio S para um bloqueio X na chave.

Devido a este comportamento do mapa de fila extra, é possível visualizar algumas diferenças de comportamento de bloqueio.

- Se o mapa do usuário for configurado como a estratégia de bloqueio PESSIMISTIC, não há muita diferença no comportamento de bloqueio. Sempre

que um flush ou commit é chamado, um bloqueio S é colocado na mesma chave no mapa de fila. Durante o momento do commit, um bloqueio X não é adquirido apenas para a chave no mapa do usuário, ele também é adquirido para a chave no mapa de fila.

- Se o mapa do usuário for configurado com a estratégia de bloqueio OPTIMISTIC ou NONE, a transação do usuário seguirá o padrão de estratégia de bloqueio PESSIMISTIC. Sempre que um flush ou commit é chamado, um bloqueio S é adquirido para a mesma chave no mapa de fila. Durante o momento do commit, um bloqueio X é adquirido para a chave no mapa de fila utilizando a mesma transação.

Novas Tentativas de Transações do Utilitário de Carga

O WebSphere eXtreme Scale não suporta transações de 2 fases ou XA. O encadeamento write-behind remove registros do mapa de fila e atualiza os registros no backend. Se o servidor falhar no meio da transação, algumas atualizações de backend podem ser perdidas.

O utilitário de carga write-behind automaticamente tenta gravar novamente as transações falhas e envia um LogSequence em dúvida para o backend para evitar a perda de dados. Esta ação exige que o utilitário de carga seja idempotente, o que significa que, quando o método Loader.batchUpdate(TxId, LogSequence) for chamado duas vezes com o mesmo valor, ele fornecerá o mesmo resultado como se fosse aplicado uma vez. As implementações do utilitário de carga devem implementar a interface RetryableLoader para ativar este recurso. Consulte na documentação da API para obter detalhes adicionais.

Falha do Utilitário de Carga

O plug-in do utilitário de carga pode falhar quando não consegue se comunicar com o back end do banco de dados. Isto pode acontecer se o servidor de banco de dados ou a conexão de rede estiver inativa. O utilitário de carga write-behind irá enfileirar as atualizações e tentará executar o push das alterações de dados para o utilitário de carga periodicamente. O utilitário de carga deve notificar o tempo de execução do WebSphere eXtreme Scale de que existe um problema de conectividade com o banco de dados gerando uma exceção LoaderNotAvailableException.

Portanto, a implementação do Utilitário de Carga deve poder distinguir uma falha de dados ou uma falha de utilitário de carga físico. A falha de dados deve ser gerada ou gerada novamente como uma exceção LoaderException ou OptimisticCollisionException, mas a falha física do utilitário de carga deve ser gerada ou gerada novamene como uma exceção LoaderNotAvailableException. O WebSphere eXtreme Scale trata estas duas exceções diferentemente:

- Se uma LoaderException for capturada pelo utilitário de carga write-behind, o utilitário de carga write-behind a considerará falha devido a alguma falha de dados, tal como uma falha de chave duplicada. O utilitário de carga write-behind irá remover a atualização do lote e tentará atualizar um registro em um momento para isolar a falha de dados. Se uma exceção LoaderException for capturada novamente durante a atualização de um registro, um registro de atualização falho é criado e registrado no mapa de atualização falho.
- Se uma LoaderNotAvailableException for capturada pelo utilitário de carga write-behind, o utilitário de carga write-behind a considerará falha porque não pode se conectar ao final do banco de dados, por exemplo, porque o backend do banco de dados estiver inativo, uma conexão com o banco de dados não estiver

disponível ou a rede estiver inativa. O utilitário de carga write-behind aguardará por 15 segundo e, em seguida, tentará novamente executar uma atualização de lote no banco de dados.

O erro comum é lançar uma LoaderException enquanto uma LoaderNotAvailableException deve ser lançada. Todos os registros enfileirados no utilitário de carga write-behind se tornarão atualizações de registro falhas, o que frustra o propósito do isolamento de falha do backend. Esse erro provavelmente ocorrerá se você gravar um utilitário de carga genérico para se comunicar com os bancos de dados.

O eXtreme Scale fornece o JPALoader como um exemplo. O JPALoader usa a API do JPA para interagir com os backends de banco de dados. Quando a rede falhar, o JPALoader obtém javax.persistence.PersitenceException, mas não reconhece a importância da falha a menos que o estado do SQL e o código de erro do SQL da SQLException em cadeia sejam verificados. O fato de que o JPALoader foi projetado para trabalhar com todos os tipos de banco de dados, complica ainda mais o problema já que os estados e os códigos de erro do SQL são diferentes quando ocorrer uma queda de rede. Para resolver isso, o WebSphere eXtreme Scale fornece uma API ExceptionMapper para permitir que usuários conectem uma implementação para mapear uma Exceção para uma exceção mais consumível. Por exemplo, os usuários podem mapear um javax.persistence.PersitenceException genérica para um LoaderNotAvailableException se o estado ou o código de erro SQL indicar uma queda de rede.

Considerações sobre Desempenho

O suporte ao armazenamento em cache write-behind aumenta o tempo de resposta removendo a atualização do utilitário de carga da transação. Ele também aumenta o rendimento do banco de dados já que as atualizações de banco de dados são combinadas. É importante compreender a sobrecarga introduzida pelo encadeamento write-behind, que executa o pull dos dados do mapa de fila e executa o push para o utilitário de carga.

A contagem máxima de atualização ou o tempo máximo de atualização necessário a ser ajustado com base nos padrões de uso e no ambiente esperados. Se o valor da contagem máxima de atualização ou o tempo máximo de atualização for muito pequeno, o gasto adicional do encadeamento write-behind pode exceder os benefícios. Configurar um valor maior para estes dois parâmetros também pode aumentar o uso da memória para enfileirar os dados e aumentar o tempo de envelhecimento dos registros do banco de dados.

Para obter um melhor desempenho, ajuste os parâmetros write-behind com base nos seguintes fatores:

- Proporção de transações de leitura e gravação
- Mesma frequência de atualização de registro
- Latência de atualização de banco de dados.

Manipulando Atualizações Write-Behind Falhas

Como a transação do WebSphere eXtreme Scale é concluída antes da transação de backend iniciar, é possível ter um falso sucesso de transação.

Por exemplo, se você tentar inserir uma entrada em uma transação do eXtreme Scale que não existe no mapa de apoio mas existe no backend, causando uma

chave duplicada, a transação do eXtreme Scale será bem-sucedida. Entretanto, a transação na qual o encadeamento write-behind insere tal objeto no backend falha com uma exceção de chave duplicada.

Manipulando atualizações write-behind falhas: lado do cliente

Uma atualização deste tipo, ou qualquer outra atualização de backend falha, é uma atualização write-behind falha. Atualizações write-behind falhas são armazenadas em um mapa de atualização write-behind falho. Este mapa funciona como uma fila de eventos para atualizações falhas. A chave da atualização é um objeto Integer exclusivo e o valor é uma instância do FailedUpdateElement. O mapa de atualização write-behind com falha é configurado com um evictor, que despeja os registros 1 hora depois de ter sido inserido. Assim, os registros de atualização com falha serão perdidos se eles não forem recuperados dentro de 1 hora.

A API do ObjectMap pode ser utilizada para recuperar as entradas do mapa de atualização write-behind falho. O nome do mapa de atualização write-behind falho é: IBM_WB_FAILED_UPDATES_<nome do mapa>. Consulte a documentação da API do WriteBehindLoaderConstants para os nomes de prefixo de cada um dos mapas do sistema write-behind. A seguir há um exemplo.

processo com falha - código de exemplo

```
ObjectMap failedMap = session.getMap(
    WriteBehindLoaderConstants.WRITE_BEHIND_FAILED_UPDATES_MAP_PREFIX + "Employee");
Object key = null;

session.begin();
while(key = failedMap.getNextKey(ObjectMap.QUEUE_TIMEOUT_NONE)) {
    FailedUpdateElement element = (FailedUpdateElement) failedMap .get(key);
    Throwable throwable = element.getThrowable();
    Object failedKey = element.getKey();
    Object failedValue = element.getAfterImage();
    failedMap .remove(key);
    // Do something interesting with the key, value, or exception.

}
session.commit();
```

Uma chamada getNextKey funciona com uma partição específica para cada transação eXtreme Scale. Em um ambiente distribuído, para obter chaves de todas as partições, você deve iniciar múltiplas transações, como mostrado no exemplo a seguir:

obtendo chaves de todas as partições - código de exemplo

```
ObjectMap failedMap = session.getMap(
    WriteBehindLoaderConstants.WRITE_BEHIND_FAILED_UPDATES_MAP_PREFIX + "Employee");
while (true) {
    session.begin();
    Object key = null;
    while(( key = failedMap.getNextKey(5000) )!= null ) {
        FailedUpdateElement element = (FailedUpdateElement) failedMap .get(key);
    Throwable throwable = element.getThrowable();
        Object failedKey = element.getKey();
        Object failedValue = element.getAfterImage();
        failedMap .remove(key);
        // Do something interesting with the key, value, or exception.

    }
    Session.commit();
}
```

Nota: O mapa de atualização falho fornece uma maneira de monitorar o funcionamento do aplicativo. Se um sistema produzir muitos registros no mapa de

atualização falho, isto é um sinal de que o aplicativo ou a arquitetura deve ser reavaliado ou revisado para utilizar o suporte write-behind. A partir da 6.1.0.5, é possível utilizar o script xsadmin para visualizar o tamanho da entrada do mapa de atualização falho.

Manipulando atualizações write-behind falhas: listener do shard

É importante detectar e registrar quando uma transação write-behind falha. Todo aplicativo utilizando write-behind precisa implementar um watcher para manipular atualizações write-behind falhas. Isto evita potencialmente ficar sem memória já os registros no Mapa de atualização inválido não são despejados porque o aplicativo é esperado para manipulá-los.

O código a seguir mostra como conectar um watcher, ou "dumper", que deve ser incluído no XML do descritor de ObjectGrid no fragmento.

```
<objectGrid name="Grid">
 <bean id="ObjectGridEventListener" className="utils.WriteBehindDumper"/>
```

É possível visualizar o bean ObjectGridEventListener que foi incluído, que é o watcher write-behind referido acima. O watcher interage nos Mapas para todos os shards principais em um JVM procurando por aqueles com write-behind ativado. Se ele localizar um, então, ele tenta registrar até 100 atualizações inválidas. Ele continua observando um shard principal até que o shard seja movido para um JVM diferente. Todos os aplicativos utilizando write-behind *devem* utilizar um watcher semelhante a este. Caso contrário, o Java Virtual Machines fica sem memória porque este mapa de erro nunca é despejado.

Consulte Código de amostra de classe do dumper write-behind para obter informações adicionais.

Código de Amostra da Classe do Dumper Write-behind

Essa amostra de código de origem mostra como gravar um watcher (dumper) para manipular atualizações write-behind com falhas.

```
//
//This sample program is provided AS IS and may be used, executed, copied and
//modified without royalty payment by customer (a) for its own instruction and
//study, (b) in order to develop applications designed to run with an IBM
//WebSphere product, either for customer's own internal use or for redistribution
//by customer, as part of such an application, in customer's own products. "
//
//5724-J34 (C) COPYRIGHT International Business Machines Corp. 2009
//All Rights Reserved * Licensed Materials - Property of IBM
//
package utils;

import java.util.Collection;
import java.util.Iterator;
import java.util.concurrent.Callable;
import java.util.concurrent.ScheduledExecutorService;
import java.util.concurrent.ScheduledFuture;
import java.util.concurrent.ScheduledThreadPoolExecutor;
import java.util.concurrent.TimeUnit;
import java.util.logging.Logger;

import com.ibm.websphere.objectgrid.BackingMap;
import com.ibm.websphere.objectgrid.ObjectGrid;
import com.ibm.websphere.objectgrid.ObjectGridException;
import com.ibm.websphere.objectgrid.ObjectGridRuntimeException;
import com.ibm.websphere.objectgrid.ObjectMap;
import com.ibm.websphere.objectgrid.Session;
import com.ibm.websphere.objectgrid.UndefinedMapException;
import com.ibm.websphere.objectgrid.plugins.ObjectGridEventGroup;
import com.ibm.websphere.objectgrid.plugins.ObjectGridEventListener;
import com.ibm.websphere.objectgrid.writebehind.FailedUpdateElement;
import com.ibm.websphere.objectgrid.writebehind.WriteBehindLoaderConstants;

/**
 * Write behind expects transactions to the Loader to succeed. If a transaction for a key fails then
 * it inserts an entry in a Map called PREFIX + mapName. The application should be checking this
 * map for entries to dump out write behind transaction failures. The application is responsible for
```

```
 * analyzing and then removing these entries. These entries can be large as they include the key, before
 * and after images of the value and the exception itself. Exceptions can easily be 20k on their own.
 *
 * The class is registered with the grid and an instance is created per primary shard in a JVM. Isso cria
 * um único encadeamento
 * e depois esse encadeamento verifica cada mapa de erro write behind
para o shard, imprime o problema e
 * depois remove a entrada.
 *
 * Isso significa que existirá um encadeamento por shard. Se o shard for
movido para outra JVM, o método deactivate
 * irá parar o encadeamento.
 * method stops the thread.
 * @author bnewport
 *
 */
public class WriteBehindDumper implements ObjectGridEventListener, ObjectGridEventGroup.ShardEvents, Callable<Boolean>
{
 static Logger logger = Logger.getLogger(WriteBehindDumper.class.getName());

 ObjectGrid grid;

 /**
  * Thread pool to handle table checkers. If the application has it's own pool
  * then change this to reuse the existing pool
  */
 static ScheduledExecutorService pool = new ScheduledThreadPoolExecutor(2); // two threads to dump records

 // the future for this shard
 ScheduledFuture<Boolean> future;

 // true if this shard is active
 volatile boolean isShardActive;

 /**
  * Normal time between checking Maps for write behind errors
  */
 final long BLOCKTIME_SECS = 20L;

 /**
  * An allocated session for this shard. No point in allocating them again and again
  */
 Session session;
 /**
  * When a primary shard is activated then schedule the checks to periodically check
  * the write behind error maps and print out any problems
  */
 public void shardActivated(ObjectGrid grid)
 {
  try
  {
   this.grid = grid;
   session = grid.getSession();

   isShardActive = true;
   future = pool.schedule(this, BLOCKTIME_SECS, TimeUnit.SECONDS); // check every BLOCKTIME_SECS seconds initially
  }
  catch(ObjectGridException e)
  {
   throw new ObjectGridRuntimeException("Exception activating write dumper", e);
  }
 }

 /**
  * Mark shard as inactive and then cancel the checker
  */
 public void shardDeactivate(ObjectGrid arg0)
 {
  isShardActive = false;
  // if it's cancelled then cancel returns true
  if(future.cancel(false) == false)
  {
   // otherwise just block until the checker completes
   while(future.isDone() == false) // wait for the task to finish one way or the other
   {
    try
    {
     Thread.sleep(1000L); // check every second
    }
    catch (InterruptedException e)
    {
    }
   }
  }
 }

 /**
  * Simple test to see if the map has write behind enabled and if so then return
  * the name of the error map for it.
  * @param mapName The map to test
  * @return The name of the write behind error map if it exists otherwise null
```

```
 */
static public String getWriteBehindNameIfPossible(ObjectGrid grid, String mapName)
{
 BackingMap map = grid.getMap(mapName);
 if(map != null && map.getWriteBehind() != null)
 {
  return WriteBehindLoaderConstants.WRITE_BEHIND_FAILED_UPDATES_MAP_PREFIX + mapName;
 }
 else
  return null;
}

/**
 * This runs for each shard. It checks if each map has write behind enabled and if it does
  * then it prints out any write behind
 * transaction errors and then removes the record.
 */
public Boolean call()
{
 logger.fine("Called for " + grid.toString());
 try
 {
  // while the primary shard is present in this JVM
  // only user defined maps are returned here, no system maps like write behind maps are in
  // this list.
  Iterator<String> iter = grid.getListOfMapNames().iterator();
  boolean foundErrors = false;
  // iterate over all the current Maps
  while(iter.hasNext() && isShardActive)
  {
   String origName = iter.next();

   // if it's a write behind error map
   String name = getWriteBehindNameIfPossible(grid, origName);
   if(name != null)
   {
    // try to remove blocks of N errors at a time
    ObjectMap errorMap = null;
    try
    {
     errorMap = session.getMap(name);
    }
    catch(UndefinedMapException e)
    {
     // at startup, the error maps may not exist yet, patience...
     continue;
    }
    // try to dump out up to N records at once
    session.begin();
    for(int counter = 0; counter < 100; ++counter)
    {
     Integer seqKey = (Integer)errorMap.getNextKey(1L);
     if(seqKey != null)
     {
      foundErrors = true;
      FailedUpdateElement elem = (FailedUpdateElement)errorMap.get(seqKey);
      //
      // Your application should log the problem here
      logger.info("WriteBehindDumper ( " + origName + ") for key (" + elem.getKey() + ") Exception: " +
       elem.getThrowable().toString());
      //
      //
      errorMap.remove(seqKey);
     }
     else
      break;
    }
    session.commit();
   }
  } // do next map
  // loop faster if there are errors
  if(isShardActive)
  {
   // reschedule after one second if there were bad records
   // otherwise, wait 20 seconds.
   if(foundErrors)
    future = pool.schedule(this, 1L, TimeUnit.SECONDS);
   else
    future = pool.schedule(this, BLOCKTIME_SECS, TimeUnit.SECONDS);
  }
 }
 catch(ObjectGridException e)
 {
  logger.fine("Exception in WriteBehindDumper" + e.toString());
  e.printStackTrace();

  //don't leave a transaction on the session.
  if(session.isTransactionActive())
  {
   try { session.rollback(); } catch(Exception e2) {}
  }
```

```
 }
 return true;
}

public void destroy() {
 // TODO Auto-generated method stub

}

public void initialize(Session arg0) {
 // TODO Auto-generated method stub

}

public void transactionBegin(String arg0, boolean arg1) {
 // TODO Auto-generated method stub

}

public void transactionEnd(String arg0, boolean arg1, boolean arg2,
  Collection arg3) {
 // TODO Auto-generated method stub

}
}
```

Configurando Replicação Ponto a Ponto com o JMS

Mecanismo de replicação ponto a ponto baseado no Java Message Service (JMS) é usado em ambiente local e distribuído do WebSphere eXtreme Scale. O JMS é um processo de replicação núcleo a núcleo e permite que as atualizações de dados fluam entre os ObjectGrids e os ObjectGrids distribuídos. Por exemplo, com esse mecanismo é possível mover as atualizações de dados de uma grade distribuída do eXtreme Scale para uma grade local do eXtreme Scale ou de uma grade para outra grade em um domínio de sistema diferente.

Antes de Iniciar

O mecanismo de replicação ponto a ponto baseado em JMS é baseado no ObjectGridEventListener baseado em JMS integrado, com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener. Para obter informações detalhadas sobre a ativação do mecanismo de replicação ponto a ponto, consulte “Listener de Eventos da JMS” na página 138.

Consulte “Ativando o Mecanismo de Invalidação do Cliente” na página 209 para obter mais informações.

A seguir há um exemplo de uma configuração XML para ativar um mecanismo de replicação ponto a ponto em uma configuração do eXtreme Scale:

configuração da
replicação ponto a ponto - exemplo de XML

```
<bean id="ObjectGridEventListener"
className="com.ibm.websphere.objectgrid.plugins.JMSObjectGridEventListener">
 <property name="replicationRole" type="java.lang.String" value="DUAL_ROLES" description="" />
    <property name="replicationStrategy" type="java.lang.String" value="PUSH" description="" />
      <property name="jms_topicConnectionFactoryJndiName" type="java.lang.String"
  value="defaultTCF" description="" />
  <property name="jms_topicJndiName" type="java.lang.String" value="defaultTopic" description="" />
  <property name="jms_topicName" type="java.lang.String" value="defaultTopic" description="" />
  <property name="jms_userid" type="java.lang.String" value="" description="" />
  <property name="jms_password" type="java.lang.String" value="" description="" />
  <property name="jndi_properties" type="java.lang.String"
 value="java.naming.factory.initial=org.apache.activemq.jndi.ActiveMQInitialContextFactory;
 java.naming.provider.url=tcp://localhost:61616;connectionFactoryNames=defaultTCF;
 topic.defaultTopic=defaultTopic"
  description="jndi properties" />
    </bean>
```

Distribuição de Alterações Entre Java Virtual Machines Peer

Os objetos LogSequence e LogElement distribuem mudanças entre JVMs peer e comunicam as mudanças que ocorreram em uma transação do eXtreme Scale com um plug-in do ObjectGridEventListener.

Para obter mais informações sobre o como o Java Message Service (JMS) pode ser usado para distribuir alterações transacionais, consulte as informações sobre o uso da JMS para distribuir alterações de transação no *Visão Geral do Produto*.

Um pré-requisito é que a instância do ObjectGrid deve ser armazenada em cache pelo ObjectGridManager. Consulte Métodos createObjectGrid para obter mais informações. O valor booleano cacheInstance deve ser configurado como true.

Não é necessário que você implemente esse mecanismo. Há um mecanismo de replicação ponto a ponto integrado disponível para utilizar esta função. Consulte as informações sobre a configuração de replicação ponto a ponto com JMS no *Guia de Administração*.

Os objetos fornecem um meio para que um aplicativo publique facilmente alterações que ocorreram em um ObjectGrid utilizando um transporte de mensagens para ObjectGrids peer nas Java Virtual Machines remotas e, então, aplica estas alterações em tais JVM. A classe LogSequenceTransformer é importante para ativar este suporte. Este artigo examina como escrever um listener usando um sistema de mensagens JVM (Java Message Service) para propagação das mensagens. Para este fim, o eXtreme Scale suporta a transmissão de LogSequences que resultam de uma consolidação transacional de um eXtreme Scale por meio dos membros do cluster do WebSphere Application Server com um plug-in fornecido pela IBM. Esta função não é ativada por padrão, mas pode ser configurada para ser operacional. Porém, quando o consumidor ou produtor não for um WebSphere Application Server, o uso de um sistema de mensagens JMS externo pode ser necessário.

Implementando o Mecanismo

A classe LogSequenceTransformer e as APIs ObjectGridEventListener, LogSequence e LogElement permitem que qualquer publicação-e-assinatura confiável seja utilizada para distribuir as alterações e filtrar os mapas que você deseja distribuir. Os snippets neste tópico mostram como utilizar estas APIs com o JMS para criar um ObjectGrid ponto a ponto compartilhado por aplicativos que estão hospedados em um conjunto diverso de plataforma compartilhando um transporte de mensagens comum.

Inicialize o plug-in

O ObjectGrid chama o método de inicialização do plug-in, parte do contrato da interface ObjectGridEventListener, quando o ObjectGrid inicia. O método de inicialização deve obter seus recursos JMS, incluindo conexões, sessões, e publicadores, e inicia o encadeamento que é o listener JMS.

Os exemplos a seguir mostram o método de inicialização:

```
exemplo do método de inicializar
public void initialize(Session session) {
        mySession = session;
        myGrid = session.getObjectGrid();
        try {
            if (mode == null) {
                throw new ObjectGridRuntimeException("No mode specified");
```

```
            }
            if (userid != null) {
                connection = topicConnectionFactory.createTopicConnection(userid, password);
            } else
                connection = topicConnectionFactory.createTopicConnection();

            // need to start the connection to receive messages.
            connection.start();

            // the jms session is not transactional (false).
            jmsSession = connection.createTopicSession(false,
        javax.jms.Session.AUTO_ACKNOWLEDGE);
            if(topic == null)
                if (topicName == null) {
                    throw new ObjectGridRuntimeException("Topic not specified");
                } else {
                    topic = jmsSession.createTopic(topicName);
                }
            publisher = jmsSession.createPublisher(topic);
            // start the listener thread.
            listenerRunning = true;
            listenerThread = new Thread(this);
            listenerThread.start();
         } catch (Throwable e) {
            throw new ObjectGridRuntimeException("Cannot initialize", e);
         }
    }
```

O código para iniciar o encadeamento usa um encadeamento Java 2 Platform, Standard Edition (Java SE). Se você estiver executando um WebSphere Application Server Versão 6.x ou um servidor enterprise WebSphere Application Server Version 5.x, use a interface de programação de aplicativos (API) do bean assíncrono para iniciar este encadeamento de daemon. Também é possível utilizar as APIs comuns. A seguir está um snippet de substituição de exemplo que mostra a mesma ação utilizando um gerenciador de trabalho:

```
// start the listener thread.
listenerRunning = true;
workManager.startWork(this, true);
```

O plug-in também deve implementar a interface Work em vez da interface Runnable. Também é necessário incluir um método de liberação para configurar a variável listenerRunning como false. O plug-in deve ser fornecido com uma instância do gerenciador de trabalho em seu construtor ou por injeção, se estiver utilizando um contêiner IoC (Inversion of Control).

Transmita as alterações

A seguir, está um método transactionEnd de amostra para a publicação das alterações locais que são feitas em um ObjectGrid. Esta amostra utiliza o JMS, embora seja possível utilizar qualquer transporte de mensagens que seja capaz de emitir mensagens de publicação-e-assinatura confiáveis.

Exemplo do método transactionEnd

```
// This method is synchronized to make sure the
    // messages are published in the order the transaction
    // were committed. If we started publishing the messages
    // in parallel then the receivers could corrupt the Map
    // as deletes may arrive before inserts etc.
    public synchronized void transactionEnd(String txid, boolean isWriteThroughEnabled, boolean committed,
            Collection changes) {
        try {
            // must be write through and commited.
            if (isWriteThroughEnabled && committed) {
                // write the sequences to a byte []
                ByteArrayOutputStream bos = new ByteArrayOutputStream();
```

```
            ObjectOutputStream oos = new ObjectOutputStream(bos);
            if (publishMaps.isEmpty()) {
                // serialize the whole collection
                LogSequenceTransformer.serialize(changes, oos, this, mode);
            } else {
                // filter LogSequences based on publishMaps contents
                Collection publishChanges = new ArrayList();
                Iterator iter = changes.iterator();
                while (iter.hasNext()) {
                    LogSequence ls = (LogSequence) iter.next();
                    if (publishMaps.contains(ls.getMapName())) {
                        publishChanges.add(ls);
                    }
                }
                LogSequenceTransformer.serialize(publishChanges, oos, this, mode);
            }
            // make an object message for the changes
            oos.flush();
            ObjectMessage om = jmsSession.createObjectMessage(bos.toByteArray());
            // set properties
            om.setStringProperty(PROP_TX, txid);
            om.setStringProperty(PROP_GRIDNAME, myGrid.getName());
            // transmit it.
            publisher.publish(om);
        }
    } catch (Throwable e) {
        throw new ObjectGridRuntimeException("Cannot push changes", e);
    }
}
```

Este método utiliza diversas variáveis de instância:

- Variável `jmsSession`: Uma sessão JMS utilizada para publicar mensagens. É criada quando o plug-in inicializa.
- Variável `mode`: O modo de distribuição.
- Variável `publishMaps`: Um conjunto que contém o nome de cada mapa com alterações para publicar. Se a variável está vazia, então, todos os mapas são publicados.
- Variável `publisher`: Um objeto TopicPublisher que é criado durante o método de inicialização de plug-in

Receber e aplicar mensagens de atualização

A seguir está o método de execução. Este método é executado em um loop até que o aplicativo pare o loop. Cada iteração do loop tenta receber uma mensagem JMS e aplicá-la ao ObjectGrid.

Exemplo do método de execução de mensagem JMS

```
private synchronized boolean isListenerRunning() {
        return listenerRunning;
    }

    public void run () {
        try {
            System.out.println("Listener starting");
            // get a jms session for receiving the messages.
            // Non transactional.
            TopicSession myTopicSession;
            myTopicSession = connection.createTopicSession(false, javax.jms.
       Session.AUTO_ACKNOWLEDGE);

            // get a subscriber for the topic, true indicates don't receive
            // messages transmitted using publishers
            // on this connection. Otherwise, we'd receive our own updates.
            TopicSubscriber subscriber = myTopicSession.createSubscriber(topic,
       null, true);
            System.out.println("Listener started");
            while (isListenerRunning()) {
                ObjectMessage om = (ObjectMessage)subscriber.receive(2000);
```

```
                if (om != null) {
                    // Use Session that was passed in on the initialize...
                    // very important to use no write through here
                    mySession.beginNoWriteThrough();
                    byte[] raw = (byte[])om.getObject();
                    ByteArrayInputStream bis = new ByteArrayInputStream(raw);
                    ObjectInputStream ois = new ObjectInputStream(bis);
                    // inflate the LogSequences
                    Collection collection = LogSequenceTransformer.inflate(ois,
           myGrid);
                    Iterator iter = collection.iterator();
                    while (iter.hasNext()) {
                        // process each Maps changes according to the mode when
                        // the LogSequence was serialized
                        LogSequence seq = (LogSequence)iter.next();
                        mySession.processLogSequence(seq);
                    }
                    mySession.commit();
                } // if there was a message
              } // loop while
              // stop the connection
              connection.close();
          } catch (IOException e) {
              System.out.println("IO Exception: " + e);
          } catch (JMSException e) {
              System.out.println("JMS Exception: " + e);
          } catch (ObjectGridException e) {
              System.out.println("ObjectGrid exception: " + e);
              System.out.println("Caused by: " + e.getCause());
          } catch (Throwable e) {
              System.out.println("Exception : " + e);
          }
          System.out.println("Listener stopped");
      }
```

Listener de Eventos da JMS

O JMSObjectGridEventListener foi projetado para suportar invalidação de cache próximo do lado do cliente e um mecanismo de replicação ponto a ponto. Ele é uma implementação Java Message Service (JMS) da interface do ObjectGridEventListener.

O mecanismo de invalidação do cliente pode ser usado em um ambiente distribuído doeXtreme Scale para garantir que os dados de cache próximos do cliente sejam sincronizados com os servidores ou outros cliente. Sem essa função, o cache perto do cliente pode deixar os dados obsoletos. Entretanto, mesmo com essa invalidação de cliente baseada em JMS, é necessário levar em consideração a janela de sincronização para atualizar um cache perto do cliente devido ao atraso do tempo de execução ao publicar as atualizações.

O mecanismo de replicação ponto a ponto pode ser usado em ambientes local e distribuído do eXtreme Scale. Ele é um processo de replicação de núcleo para núcleo do ObjectGrid e permite o fluxo de atualizações de dados entre os ObjectGrids locais e distribuídos. Por exemplo, com esse mecanismo, é possível mover as atualizações de dados de uma grade distribuída para um ObjectGrid local ou de qualquer grade para outra grade em um domínio de sistema diferente.

O JMSObjectGridEventListener requer que o usuário configure informações de JMS e Java Naming and Directory Interface (JNDI) para obter os recursos do JMS necessários. Além disso, as propriedades relacionadas à replicação devem ser configuradas corretamente. Em um ambiente JEE, o JNDI deve estar disponível em

ambos os contêineres da Web e do Enterprise JavaBean (EJB). Nesse caso, a propriedade JNDI é opcional, a menos que você deseje obter os recursos JMS externos.

Esse listener de evento possui propriedades que podem ser configuradas com o XML ou com abordagens programáticas, que podem ser usadas apenas para invalidação do cliente, apenas replicação ponto a ponto, ou ambos. A maioria das propriedades é opcional para customizar o comportamento para obter a funcionalidade necessária.

.

Para obter mais informações, consulte a API do JMSObjectGridEventListener.

Estendendo o Plug-in do JMSObjectGridEventListener

O plug-in do JMSObjectGridEventListener permite que as instâncias equivalentes do ObjectGrid recebam atualizações quando os dados na grade forem alterados ou despejados. Também permite que os clientes sejam notificados quando as entradas forem atualizadas ou despejadas de uma grade do eXtreme Scale. Esse tópico descreve como estender o plug-in do JMSObjectGridEventListener para permitir que os aplicativos sejam notificados quando uma mensagem JMS for recebida. Isso é mais útil ao usar a configuração do CLIENT_SERVER_MODEL para invalidação do cliente.

Durante a execução na função de receptor, o método JMSObjectGridEventListener.onMessage substituído é automaticamente chamado pelo tempo de execução do eXtreme Scale quando a instância do JMSObjectGridEventListener recebe atualizações de mensagem JMS da grade. Essas mensagens agrupam uma coleta de Objetos LogSequence . Os objetos LogSequence são transmitidos para o método onMessage e o aplicativo usa o LogSequence para identificar quais entradas de cache foram inseridas, excluídas, atualizadas ou invalidadas.

Para usar o ponto de extensão onMessage, os aplicativos executam as seguintes etapas.

1. Criar uma nova classe, estendendo a classe JMSObjectGridEventListener, substituindo o método onMessage.
2. Configurar o JMSObjectGridEventListener estendido da mesma forma que o ObjectGridEventListener para ObjectGrid.

A classe JMSObjectGridEventListener estendida é uma classe-filha da classe JMSObjectGridEventListener e só pode substituir dois métodos: o initialize (opcional) e o onMessage. Se uma classe-filha da classe JMSObjectGridEventListener precisar utilizar quaisquer artefatos de ObjectGrid como ObjectGrid ou Session no método onMessage, ela pegará esses artefatos no método de inicialização e os armazenará em cache como variáveis de instância. Além disso, no método onMessage, artefatos de ObjectGrid armazenados em cache podem ser utilizados para processar uma coleta de LogSequences transmitida.

Nota: O método de inicialização substituído deve invocar o método super.initialize para inicializar o JMSObjectGridEventListener pai corretamente.

A seguir há uma amostra de uma classe JMSObjectGridEventListener estendida.

```
package com.ibm.websphere.samples.objectgrid.jms.price;

import java.util.*;
import com.ibm.websphere.objectgrid.*;
import com.ibm.websphere.objectgrid.plugins.LogElement;
import com.ibm.websphere.objectgrid.plugins.LogSequence;
import com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener;

public class ExtendedJMSObjectGridEventListener extends JMSObjectGridEventListener{
 protected static boolean debug = true;

    /**
     * Essa é a grade associada a esse listener.
     */
    ObjectGrid grid;

    /**
     * Essa é a sessão associada a esse listener.
     */
    Session session;

    String objectGridType;

    public List receivedLogSequenceList = new ArrayList();


 /* (não-Javadoc)
  * @see com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener
    #initialize(com.ibm.websphere.objectgrid.Session)
  */
 public void initialize(Session session) {
  // Note: if need to use any ObjectGrid artifact, this class need to get ObjectGrid
  //  from the passed Session instance and get ObjectMap from session instance
  // for any transactional ObjectGrid map operation.

  super.initialize(session);  // must invoke super's initialize method.
  this.session = session; // cache the session instance, in case need to
  //   use it to perform map operation.
  this.grid = session.getObjectGrid(); // get ObjectGrid, in case need
  //   to get ObjectGrid information.

  if (grid.getObjectGridType() == ObjectGrid.CLIENT)
   objectGridType = "CLIENT";
  else if (grid.getObjectGridType() == ObjectGrid.SERVER)
   objectGridType = "Server";

  if (debug)
   System.out.println("ExtendedJMSObjectGridEventListener[" +
     objectGridType + "].initialize() : grid = " + this.grid);
 }

 /* (não-Javadoc)
  * @see com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener
    #onMessage(java.util.Collection)
  */
 protected void onMessage(Collection logSequences) {
  System.out.println("ExtendedJMSObjectGridEventListener[" +
    objectGridType + "].onMessage(): ");

  Iterator iter = logSequences.iterator();

  while(iter.hasNext()) {
            LogSequence seq = (LogSequence)iter.next();

      StringBuffer buffer = new StringBuffer();
      String mapName = seq.getMapName();
      int size = seq.size();
      buffer.append("\nLogSequence[mapName=" + mapName + ", size=" + size + ",
     objectGridType=" + objectGridType
        + "]: ");

      Iterator logElementIter = seq.getAllChanges();
      for (int i = seq.size() - 1; i >= 0; --i) {
       LogElement le = (LogElement) logElementIter.next();
       buffer.append(le.getType() + " -> key=" + le.getCacheEntry().getKey() + ", ");
      }
      buffer.append("\n");

      receivedLogSequenceList.add(buffer.toString());
```

```
      if (debug) {
       System.out.println("ExtendedJMSObjectGridEventListener["
      + objectGridType + "].onMessage(): " + buffer.toString());
      }
  }
 }

 public String dumpReceivedLogSequenceList() {
  String result = "";
  int size = receivedLogSequenceList.size();
  result = result + "\nExtendedJMSObjectGridEventListener[" + objectGridType
    + "]: receivedLogSequenceList size = " + size + "\n";
  for (int i = 0; i < size; i++) {
   result = result + receivedLogSequenceList.get(i) + "\n";
  }
  return result;
 }

 public String toString() {
  return "ExtendedJMSObjectGridEventListener["
    + objectGridType + " - " + this.grid + "]";
 }
}
```

Configuração

A classe JMSObjectGridEventListener estendida deve ser configurada da mesma forma para o mecanismo de invalidação de cliente e de replicação ponto a ponto. A seguir há um exemplo de configuração XML.

```
<objectGrid name="PRICEGRID">
   <bean id="ObjectGridEventListener"
    className="com.ibm.websphere.samples.objectgrid.jms.
      price.ExtendedJMSObjectGridEventListener">
    <property name="invalidationModel" type="java.lang.String"
value="CLIENT_SERVER_MODEL" description="" />
    <property name="invalidationStrategy" type="java.lang.String"
     value="INVALIDATE" description="" />
    <property name="jms_topicConnectionFactoryJndiName" type="java.lang.String"
     value="jms/TCF" description="" />
    <property name="jms_topicJndiName" type="java.lang.String"
     value="GRID.PRICEGRID" description="" />
    <property name="jms_topicName" type="java.lang.String"
     value="GRID.PRICEGRID" description="" />
    <property name="jms_userid" type="java.lang.String" value="" description="" />
    <property name="jms_password" type="java.lang.String" value="" description="" />
   </bean>
   <backingMap name="PRICE" pluginCollectionRef="PRICE"></backingMap>
      </objectGrid>
```

Nota: ClassName do bean ObjectGridEventListener é configurado com a classe JMSObjectGridEventListener estendida com as mesmas propriedades que o JMSObjectGridEventListener genérico.

Arquivo XML descritor do ObjectGrid

Para configurar o WebSphere eXtreme Scale, utilize um arquivo XML descritor do ObjectGrid e a API do ObjectGrid.

Nas seções a seguir, os arquivos XML de amostra são fornecidos para ilustrar diversas configurações. Cada elemento e atributo do arquivo XML está definido. Use o esquema de descritor XML do ObjectGrid para criar o arquivo descritor XML. Consulte o “Arquivo objectGrid.xsd” na página 158 para obter um exemplo do esquema do descritor XML do ObjectGrid.

Uma versão modificada do arquivo `companyGrid.xml` original é utilizada. O arquivo `companyGridSingleMap.xml` a seguir é como o arquivo `companyGrid.xml`. O

arquivo companyGridSingleMap.xml tem um mapa e o arquivo companyGrid.xml tem quatro mapas. Os elementos e os atributos do arquivo são descritos em detalhes no exemplo a seguir.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
  xmlns="http://ibm.com/ws/objectgrid/config">

  <objectGrids>
    <objectGrid name="CompanyGrid">
      <backingMap name="Customer"/>
    </objectGrid>
  </objectGrids>
</objectGridConfig>
```

Elemento objectGridConfig

O elemento objectGridConfig é o elemento de nível superior do arquivo XML. Grave este elemento em seu documento XML do eXtreme Scale, conforme mostrado no exemplo anterior. Este elemento configura o espaço de nomes do arquivo e o local do esquema. O esquema está definido no arquivo objectGrid.xsd.

- Número de ocorrências: Uma
- Elemento-filho: Elemento objectGrids e elemento backingMapPluginCollections

Elemento objectGrids

O elemento objectGrids é um contêiner para todos os elementos objectGrid. No arquivo companyGridSingleMap.xml, o elemento objectGrids contém um objectGrid, o CompanyGrid objectGrid.

- Número de ocorrências: Uma ou mais
- Elemento-filho: Elemento objectGrid

Elemento objectGrid

Utilize o elemento objectGrid para definir um ObjectGrid. Cada um dos atributos no elemento objectGrid corresponde a um método na interface ObjectGrid.

- Número de ocorrências: Uma para muitas
- Elemento-filho: Elemento bean, elemento backingMap, elemento querySchema e elemento streamQuerySet

Atributos

name
Especifica o nome que é designado ao ObjectGrid. A validação de XML falhará se esse atributo estiver faltando. (Necessário)

securityEnabled
Ativa a segurança no nível do ObjectGrid, o que possibilita as autorizações de acesso aos dados no mapa, quando você configura o atributo como true. O padrão é true. (Opcional)

authorizationMechanism
Configura o mecanismo de autorização para o elemento. É possível configurar o atributo com um dos dois valores: AUTHORIZATION_MECHANISM_JAAS ou AUTHORIZATION_MECHANISM_CUSTOM. O padrão é AUTHORIZATION_MECHANISM_JAAS. Configurado como AUTHORIZATION_MECHANISM_CUSTOM quando você estiver

utilizando um plug-in MapAuthorization customizado. É necessário configurar o atributo securityEnabled como `true` para o atributo authorizationMechanism tomar efeito. (Opcional)

permissionCheckPeriod
Especifica um valor de número inteiro em segundos que indica com que frequência verificar a permissão que é utilizada para permitir um acesso do cliente. O padrão é `0`. Ao configurar o valor de atributo `0`, cada chamada de método get, put, update, remove ou evict solicita que o mecanismo de autorização, seja a autorização Java Authentication and Authorization Service (JAAS) ou a autorização customizada, verifique se o usuário atual possui permissão. Um valor maior do que 0 indica o número de segundos para armazenar em cache um conjunto de permissões antes de retornar ao mecanismo de autorização para atualizar. É necessário configurar o atributo securityEnabled como `true` para o atributo permissionCheckPeriod tomar efeito. (Opcional)

txTimeout
Especifica a quantidade de tempo em segundos permitida para que uma transação seja concluída. Se uma transação não for concluída nesse período de tempo, ela será marcada para retrocesso e o resultado será uma exceção TransactionTimeoutException. Se você configurar o valor como `0`, as transações nunca expirarão. (Opcional)

entityMetadataXMLFile
Especifica o caminho relativo para o arquivo XML descritor da entidade. O caminho é referente ao local do arquivo descritor do Objectgrid. Utilize este atributo para definir um esquema de entidade utilizando um arquivo XML. As entidades devem ser definidas antes de iniciar o eXtreme Scale de forma que cada entidade possa vincular-se a um BackingMap. (Opcional)

```
<objectGrid
(1)   name="objectGridName"
(2)   securityEnabled="true" | "false"
(3)   authorizationMechanism="AUTHORIZATION_MECHANISM_JASS" | "AUTHORIZATION_MECHANISM_CUSTOM"
(4)    permissionCheckPeriod="permission_check_period"
(5)   txTimeout="seconds"
(6)   entityMetadataXMLFile="URL"
/>
```

No exemplo a seguir, o arquivo `companyGridObjectGridAttr.xml` demonstra uma maneira de configurar os atributos de um elemento objectGrid. A segurança é ativada, o mecanismo de autorização é configurado como JAAS e o período de verificação da permissão é configurado para 45 segundos. O arquivo também registra entidades especificando um atributo entityMetadataXMLFile.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlnc:xsi="http:www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid" securityEnabled="true"
   authorizationMechanism="AUTHORIZATION_MECHANISM_JASS"
   permissionCheckPeriod="45"
   entityMetadataXMLFile="companyGridEntities.xml">
   <backingMap name="Customer"/>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

A amostra de código a seguir demonstra a abordagem programática para obter a mesma configuração que o arquivo `companyGridObjectGridAttr.xml` no exemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);

companyGrid.setSecurityEnabled();
companyGrid.setAuthorizationMechanism(SecurityConstants.AUTHORIZATION_MECHANISM_JAAS);
companyGrid.setPermissionCheckPeriod(45);
companyGrid.registerEntities(new URL("file:companyGridEntities.xml"));
```

Elemento backingMap

O elemento backingMap é utilizado para definir uma instância BackingMap em um ObjectGrid. Cada um dos atributos no elemento backingMap corresponde a um método na interface BackingMap. Para obter detalhes, consulte as informações sobre a interface BackingMap no *Guia de Programação*.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Elemento timeBasedDBUpdate

Atributos

name
Especifica o nome que é designado à instância do backingMap. Se este atributo estiver ausente, a validação de XML falhará. (Necessário)

readOnly
Configura uma instância do BackingMap como leitura-gravação quando você especifica o atributo como `false`. Quando você especifica o atributo como `true`, a instância do BackingMap é somente leitura. As chamadas para Session.getMap(String) resultam na criação de um mapa dinâmico se o nome transmitido para o método corresponder à expressão regular especificada no atributo name desse backingMap. O valor padrão é `false`. (Opcional)

template
Especifica se os mapas dinâmicos podem ser usados. Configure esse valor para `true` se o mapa BackingMap for um mapa de modelo. Os mapas de modelo podem ser utilizados para criar dinamicamente os mapas após ObjectGrid ser iniciado. As chamadas de Session.getMap(String) resultarão na criação de mapas dinâmicos se o nome transmitido para o método corresponder à expressão regular especificada no atributo de nome desse backingMap. O valor padrão é `false`. (Opcional)

pluginCollectionRef
Especifica uma referência para um plug-in do backingMapPluginCollection. O valor desse atributo deve corresponder ao atributo ID de um plug-in backingMapCollection. A validação falhará se não existir nenhum ID correspondente. Configure o atributo para reutilizar plug-ins do BackingMap. (Opcional)

numberOfBuckets
Especifica o número de depósitos para a instância do BackingMap utilizar. A instância do BackingMap utiliza um mapa hash para implementação. Se existirem muitas entradas no BackingMap, mais depósitos resultarão em melhor desempenho, porque o risco de colisões é menor à medida que o número de depósitos aumenta. Mais depósitos também resultam em maior simultaneidade. Especifique um valor de `0` para desativar o cache local em um cliente ao comunicar-se remotamente com o eXtreme Scale. Ao configurar o valor como `0` para um cliente, configure o valor apenas no arquivo descritor XML do ObjectGrid da substituição do cliente. (Opcional)

preloadMode
Configura o modo preload se um plug-in do utilitário de carga for configurado

para esta instância do BackingMap. O valor padrão é `false`. Se o atributo estiver configurado como `true`, o método Loader.preloadMap(Session, BackingMap) será chamado de maneira assíncrona. Caso contrário, a execução do método é bloqueada ao carregar dados de forma que o cache fique indisponível até que o pré-carregamento seja concluído. O pré-carregamento ocorre durante a inicialização. (Opcional)

lockStrategy
Especifica se o gerenciador de bloqueio interno é utilizado sempre que uma entrada de mapa é acessada por uma transação. Configure esse atributo com um dos três valores: `OPTIMISTIC`, `PESSIMISTIC` ou `NONE`. O valor padrão é `OPTIMISTIC`. (Opcional)

A estratégia de bloqueio otimista é normalmente utilizada quando um mapa que não possui um plug-in do utilitário de carga for mais lido do que gravado ou atualizado e quando o bloqueio não for fornecido pelo gerenciador de persistência usando o eXtreme Scale como um cache secundário ou pelo aplicativo. Um bloqueio exclusivo é obtido em uma entrada do mapa que é inserido, atualizado ou removido no momento do commit. O bloqueio assegura que as informações de versão não poderão ser alteradas por outra transação enquanto a transação que está sendo confirmada estiver executando uma verificação de versão otimista.

A estratégia de bloqueio pessimista é normalmente utilizada para um mapa que não possui um plug-in do utilitário de carga, e o bloqueio não é fornecido pelo gerenciador de persistência utilizando o eXtreme Scale como um cache secundário ou pelo aplicativo. A estratégia de bloqueio pessimista é utilizada quando a estratégia de bloqueio otimista falha com muita frequência porque as transações de atualização frequentemente colidem na mesma entrada do mapa.

A estratégia de ausência de bloqueio indica que o LockManager interno não é necessário. O controle de simultaneidade é fornecido fora do 3eXtreme Scale, seja pelo gerenciador de persistência usando o eXtreme Scale como um cache ou aplicativo secundário ou pelo plug-in do utilitário de carga que usa os bloqueios do banco de dados para controlar a simultaneidade.

Para obter mais informações, consulte as informações sobre o bloqueio de entrada de mapa no *Guia de Programação*.

numberOfLockBuckets
Configura o número de depósitos de bloquei que são utilizados pelo gerenciador de bloqueios para a instância do BackingMap. Configure o atributo lockStrategy como `OPTIMISTIC` ou `PESSIMISTIC` para criar um gerenciador de bloqueios para a instância do BackingMap. O gerenciador de bloqueios utiliza um mapa hash para controlar entradas bloqueadas por uma ou mais transações. Se existirem muitas entradas, mais depósitos resultarão em melhor desempenho, porque o risco de colisões é menor à medida que o número de depósitos aumenta. Mais depósitos de bloqueios também resultam em maior simultaneidade. Configure o atributo lockStrategy como `NONE` para especificar que a instância do BackingMap não utilize nenhum gerenciador de bloqueios. (Opcional)

lockTimeout
Configura o tempo limite do bloqueio que é utilizado pelo gerenciador de bloqueios para a instância do BackingMap. Configure o atributo lockStrategy como `OPTIMISTIC` ou `PESSIMISTIC` para criar um gerenciador de bloqueios para a instância do BackingMap. Para evitar a ocorrência de conflitos, o gerenciador de bloqueios possui um valor de tempo limite de 15 segundos. Se o tempo limite for excedido, ocorre uma exceção LockTimeoutException. O valor padrão

de 15 segundos é suficiente para a maioria dos aplicativos, mas em um sistema com grande extremamente carregado, um tempo limite pode ocorrer quando não existir nenhum conflito. Utilize o atributo lockTimeout para aumentar o valor do padrão para evitar a ocorrência de falsas exceções de tempo limite. Configure o atributo lockStrategy como NONE para especificar que a instância do BackingMap não utilize nenhum gerenciador de bloqueios. (Opcional)

CopyMode
Especifica se uma operação get de uma entrada na instância BackingMap retorna o valor real, uma cópia do valor ou um proxy para o valor. Configure o atributo CopyMode para um dos cinco valores:

COPY_ON_READ_AND_COMMIT
O valor padrão é COPY_ON_READ_AND_COMMIT. Configure o valor como COPY_ON_READ_AND_COMMIT para garantir que um aplicativo nunca faça uma referência ao objeto de valor que está na instância BackingMap. Em vez disso, o aplicativo trabalha sempre com uma cópia do valor que está na instância de BackingMap. (Opcional)

COPY_ON_READ
Configure o valor como COPY_ON_READ para melhorar o desempenho sobre o valor COPY_ON_READ_AND_COMMIT eliminando a cópia que ocorre quando uma transação é confirmada. Para preservar a integridade dos dados do BackingMap, o aplicativo é confirmado para excluir cada referência para uma entrada após a transação ser confirmada. A configuração deste valor resulta em um método ObjectMap.get retornando uma cópia do valor ao invés de uma referência ao valor, o que garante que as alterações que são feitas pelo aplicativo no valor não afetem o elemento BackingMap até que a transação seja confirmada.

COPY_ON_WRITE
Configure o valor como COPY_ON_WRITE para melhorar o desempenho sobre o valor COPY_ON_READ_AND_COMMIT eliminando a cópia que ocorre quando o método ObjectMap.get é chamado pela primeira vez por uma transação para uma determinada chave. Em vez disso, o método ObjectMap.get retorna um proxy para o valor em vez de uma referência direta ao objeto de valor. O proxy assegura que não seja feita uma cópia do valor, a menos que o aplicativo chame um método set na interface do valor.

NO_COPY
Configure o valor como NO_COPY para permitir que um aplicativo nunca modifique um objeto de valor que é obtido utilizando um método ObjectMap.get na troca de aprimoramentos de desempenho. Configure o valor como NO_COPY para mapas associados com as entidades da API do EntityManager.

COPY_TO_BYTES
Configure o valor para COPY_TO_BYTES para melhorar a área de cobertura da memória para tipos complexos do Object e para melhorar o desempenho quando a cópia de um Object depender da serialização para ser feita. Se um Object não for Clonável ou um ObjectTransformer com um método copyValue eficiente não for fornecido, o mecanismo de cópia padrão deverá serializar e aumentar o objeto para fazer uma cópia. Com a configuração COPY_TO_BYTES, o aumento é executado apenas durante a leitura e a serialização é executada apenas durante a confirmação.

Para obter mais informações sobre essas configurações, consulte as informações sobre as boas práticas do CopyMode no *Guia de Programação*..

valueInterfaceClassName
Especifica uma classe que é necessária ao configurar o atributo CopyMode como `COPY_ON_WRITE`. Esse atributo é ignorado para todos os outros modos. O valor `COPY_ON_WRITE` utiliza um proxy quando são feitas chamadas do método ObjectMap.get. O proxy assegura que não seja feita uma cópia do valor, a menos que o aplicativo chame um método set na classe especificada como o atributo valueInterfaceClassName. (Opcional)

copyKey
Especifica se uma cópia da chave é necessária quando uma entrada de mapa é criada. A cópia do objeto de chave permite que o aplicativo utilize o mesmo objeto de chave para cada operação de ObjectMap. Configure o valor como `true` para copiar o objeto de chave quando uma entrada de mapa é criada. O valor padrão é `false`. (Opcional)

nullValuesSupported
Configure o valor como `true` para suportar valores nulos no ObjectMap. Quando valores nulos são suportados, uma operação get que retorna null pode significar que o valor é nulo ou que o mapa não contém a chave que é passada para o método. O valor padrão é `true`. (Opcional)

ttlEvictorType
Especifica como o prazo de expiração de uma entrada do BackingMap é computado. Configure este atributo para um desses valores: `CREATION_TIME`, `LAST_ACCESS_TIME`, `LAST_UPDATE_TIME` ou `NONE`. O valor `CREATION_TIME` indica que um prazo de expiração de entrada é a soma do hora da criação da entrada mais o valor de atributo timeToLive. O valor `LAST_ACCESS_TIME` indica que um prazo de expiração de entrada é a soma do horário do último acesso da entrada (se a entrada foi atualizada ou simplesmente lida) mais o valor do atributo timeToLive. O valor `LAST_UPDATE_TIME` indica que um prazo de expiração de entrada é a soma do horário da última atualização da entrada mais o valor do atributo timeToLive. O valor `NONE`, que é o padrão, indica que uma entrada não possui prazo de expiração e está presente na instância do BackingMap até que o aplicativo explicitamente remova ou invalide a entrada. (Opcional)

timeToLive
Especifica, em segundos, quanto tempo cada entrada do mapa fica presente. O valor padrão de `0` significa que a entrada do mapa está presente para sempre, ou até que o aplicativo explicitamente remova ou invalide a entrada. Caso contrário, o evictor TTL despeja a entrada de mapa com base nesse valor. (Opcional)

streamRef
Especifica que o backingMap é um mapa de origem do fluxo. Qualquer inserção ou atualização no backingMap é convertida em um evento de fluxo para o mecanismo de consulta de fluxo. Esse atributo deve fazer referência a um nome do fluxo válido em um streamQuerySet. (Opcional)

viewRef
Especifica que o backingMap é um mapa de visualização. A saída da visualização do mecanismo de consulta de fluxo é convertida no formato de tupla do eXtreme Scale e colocada no mapa. (Opcional)

writeBehind
Especifica que o suporte write-behind está ativado com parâmetros write-behind (Opcional). Os parâmetros write-behind consistem em uma

duração da atualização máxima e uma contagem de atualização de chaves. O formato do parâmetro write-behind é `"[T(time)][;][C(count)]"`. O banco de dados é atualizado quando um dos seguintes eventos ocorre:

- A duração da atualização máxima, especificada em segundos, passou desde a última atualização.
- O número de atualizações disponíveis no mapa de fila alcançou a contagem de atualização máxima.

Para obter informações adicionais, consulte "Suporte de Armazenamento em Cache Write-behind" na página 124.

O suporte Write-behind é uma extensão do plug-in do Carregador, que você usa para integrar o eXtreme Scale ao banco de dados. Por exemplo, consulte as informações do "Configurando Utilitários de Carga do JPA" na página 230 sobre como configurar um carregador JPA.

evictionTriggers
Configura os tipos de acionadores de despejo adicionais a utilizar. Todos os evictors para o mapa de apoio utilizam esta lista de acionadores adicionais. Para evitar uma IllegalStateException, esse atributo deve ser chamado antes do método ObjectGrid.initialize(). Além disso, observe que o método ObjectGrid.getSession() chama implicitamente o método ObjectGrid.initialize() se o método já foi chamado pelo aplicativo. As entradas na lista de acionadores são separadas por ponto-e-vírgulas. Os Current Eviction Triggers incluem MEMORY_USAGE_THRESHOLD. (Opcional)

```
<backingMap
(1)   name="objectGridName"
(2)   readOnly="true" | "false"
(3)      template="true" | "false"
(4)   pluginCollectionRef="reference to backingMapPluginCollection"
(5)   numberOfBuckets="number of buckets"
(6)   preloadMode="true" | "false"
(7)   lockStrategy="OPTIMISTIC" | "PESSIMISTIC" | "NONE"
(8)   numberOfLockBuckets="number of lock buckets"
(9)   lockTimeout="lock timeout"
(10)   copyMode="COPY_ON_READ_AND_COMMIT" | "COPY_ON_READ" | "COPY_ON_WRITE"
       | "NO_COPY" | "COPY_TO_BYTES"
(11)   valueInterfaceClassName="value interface class name"
(12)   copyKey="true" | "false"
(13)   nullValuesSupported="true" | "false"
(14)   ttlEvictorType="CREATION_TIME" | "LAST_ACCESS_TIME" | "LAST_UPDATE_TIME" | NONE"
(15)   timeToLive="time to live"
(16)   streamRef="reference to a stream"
(17)   viewRef="reference to a view"
(18)   writeBehind="write-behind parameters"
(19)   evictionTriggers="MEMORY_USAGE_THRESHOLD"
/>
```

No exemplo a seguir, o arquivo `companyGridBackingMapAttr.xml` é utilizado para demonstrar uma configuração do backingMap de amostra.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
    <backingMap name="Customer" readOnly="true"
     numberOfBuckets="641" preloadMode="false"
     lockStrategy="OPTIMISTIC" numberOfLockBuckets="409"
     lockTimeout="30" copyMode="COPY_ON_WRITE"
     valueInterfaceClassName="com.ibm.websphere.samples.objectgrid.CounterValueInterface"
     copyKey="true" nullValuesSupported="false"
     ttlEvictorType="LAST_ACCESS_TIME" timeToLive="3000"/>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

O código de amostra a seguir demonstra a abordagem programática para a obtenção da mesma configuração que o arquivo `companyGridBackingMapAttr.xml` no exemplo anterior:

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);

BackingMap customerMap = companyGrid.defineMap("Customer");
customerMap.setReadOnly(true);
customerMap.setNumberOfBuckets(641);
customerMap.setPreloadMode(false);
customerMap.setLockStrategy(LockStrategy.OPTIMISTIC);
customerMap.setNumberOfLockBuckets(409);
customerMap.setLockTimeout(30);

// when setting copy mode to COPY_ON_WRITE, a valueInterface class is required
customerMap.setCopyMode(CopyMode.COPY_ON_WRITE,
 com.ibm.websphere.samples.objectgrid.CounterValueInterface.class);
customerMap.setCopyKey(true);
customerMap.setNullValuesSupported(false);
customerMap.setTtlEvictorType(TTLType.LAST_ACCESS_TIME);
customerMap.setTimeToLive(3000); // set time to live to 50 minutes
```

Elemento bean

Utilize o elemento bean para definir plug-ins. É possível conectar plug-ins aos elementos objectGrid e BackingMap.

- Número de ocorrências no elemento objectGrid: Zero para muitas
- Número de ocorrências no elemento backingMapPluginCollection: Zero para muitas
- Elemento-filho: Elemento property

Atributos

id Especifica o tipo de plug-in a criar. (Necessário)

Os plug-ins para um bean que é um elemento-filho do elemento objectGrid são incluídos na lista a seguir:

- Plug-in TransactionCallback
- Plug-in ObjectGridEventListener
- Plug-in SubjectSource
- Plug-in MapAuthorization
- Plug-in SubjectValidation

Os plug-ins válidos para um bean que é um elemento-filho do elemento backingMapPluginCollection são incluídos na lista a seguir:

- Plug-in do Utilitário de Carga
- Plug-in ObjectTransformer
- Plug-in OptimisticCallback
- Plug-in Evictor
- Plug-in MapEventListener
- Plug-in MapIndex

className
Especifica o nome da classe ou o Spring bean para instanciar para criar o plug-in. A classe deve implementar a interface do tipo de plug-in. Por exemplo, se você especificar `ObjectGridEventListener` como o valor do atributo do ID, o valor do atributo className deve fazer referência a uma classe que implemente a interface ObjectGridEventListener. (Necessário)

```
<bean
(1)   id="TransactionCallback" | "ObjectGridEventListener" |"SubjectSource" |
    "MapAuthorization" | "SubjectValidation" | "Loader" | "ObjectTransformer" |
    "OptimisticCallback" | "Evictor" | "MapEventListener" | "MapIndexPlugin"
(2)   className="class name" | "(spring)bean name"
/>
```

No exemplo a seguir, o arquivo companyGridBean.xml é utilizado para demonstrar como configurar plug-ins utilizando o elemento bean. Um plug-in do ObjectGridEventListener é incluído no CompanyGrid ObjectGrid. O atributo className para este bean é a classe com.ibm.websphere.objectgrid.plugins.builtins.TranPropListener. Esta classe implementa a interface com.ibm.websphere.objectgrid.plugins.ObjectGridEventListener conforme necessário.

Um plug-in BackingMap também está definido no arquivo companyGridBean.xml. Um plug-in evictor está incluído na instância Customer BackingMap. Como o ID do bean é Evictor, o atributo className deve especificar uma classe que implemente a interface com.ibm.websphere.objectgrid.plugins.Evictor. A classe com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor implementa esta interface. O backingMap faz referência aos seus plug-ins utilizando o atributo pluginCollectionRef.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   bean id="ObjectGridEventListener"
    className="com.ibm.websphere.objectgrid.plugins.builtins.TranPropListener"/>
   <backingMap name="Customer"
    pluginCollectionRef="customerPlugins"/>
  </objectGrid>
 </objectGrids>
 <backingMapPluginCollections>
  <backingMapPluginCollection id="customerPlugins">
   <bean id="Evictor"
     className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor/>
  </backingMapPluginCollection>
 </backingMapPluginCollections>
</objectGridConfig>
```

A amostra de código a seguir demonstra a abordagem programática para obter a mesma configuração que o arquivo companyGridBean.xml no exemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
TranPropListener tranPropListener = new TranPropListener();
companyGrid.addEventListener(tranPropListener);

BackingMap customerMap = companyGrid.defineMap("Customer");
Evictor lruEvictor = new com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor();
customerMap.setEvictor(lruEvictor);
```

Elemento property

Utilize o elemento property para incluir propriedades nos plug-ins. O nome da propriedade deve corresponder a um método configurado na classe referenciada pelo bean de conteúdo.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

name
Especifica o nome da propriedade. O valor atribuído a esse atributo deve corresponder a um método set na classe que é fornecida como atributo className no bean que contém o atributo. Por exemplo, se você configurar o atributo className do bean como `com.ibm.MyPlugin` e o nome da propriedade que é fornecido for `size`, a classe `com.ibm.MyPlugin` deve ter um método setSize. (Necessário)

type
Especifica o tipo da propriedade. O tipo é passado para o método configurado que é identificado pelo atributo name. Os valores válidos são os primitivos Java, os correspondentes java.lang e o java.lang.String. Os atributos name e type devem corresponder a uma assinatura de método no atributo className do bean. Por exemplo, se você configurar o nome como `size` e o tipo como `int`, um método setSize(int) deve existir na classe que é especificada como o atributo className para o bean. (Necessário)

value
Especifica o valor da propriedade. Este valor é convertido no tipo especificado pelo atributo type e, em seguida, é utilizado como um parâmetro na chamada para o método set identificado pelos atributos name e type. O valor deste atributo não é validado de nenhuma maneira. (Necessário)

description
Descreve a propriedade. (Opcional)

```
<bean
(1)   name="name"
(2)   type="java.lang.String" | "boolean" | "java.lang.Boolean" | "int" |
    "java.lang.Integer" | "double" | "java.lang.Double" | "byte" |
    "java.lang.Byte" | "short" | "java.lang.Short" | "long" |
    "java.lang.Long" | "float" | "java.lang.Float" | "char" |
    "java.lang.Character"
(3)   value="value"
(4)   description="description"
/>
```

No exemplo a seguir, o arquivo `companyGridProperty.xml` é utilizado para demonstrar como incluir um elemento property em um bean. Neste exemplo, uma propriedade com o nome maxSize e o tipo int são incluídos em um evictor. O Evictor com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor tem uma assinatura de método que corresponde ao método setMaxSize(int). Um valor de número inteiro de 499 é passado para o método setMaxSize(int) na classe com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Customer"
    pluginCollectionRef="customerPlugins"/>
  </objectGrid>
 </objectGrids>
 <backingMapPluginCollections>
  <backingMapPluginCollection id="customerPlugins">
   <bean id="Evictor"
     className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor>
     <property name="maxSize" type="int" value="449"
       description="The maximum size of the LRU Evictor"/>
   </bean>
  </backingMapPluginCollection>
 </backingMapPluginCollections>
</objectGridConfig>
```

A amostra de código a seguir demonstra a abordagem programática para obter a mesma configuração do arquivo companyGridProperty.xml no exemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);

BackingMap customerMap = companyGrid.defineMap("Customer");

LRUEvictor lruEvictor = new com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor();
// if the XML file is used instead,
// the property that was added would cause the following call to occur
lruEvictor.setMaxSize(449);
customerMap.setEvictor(lruEvictor);
```

Elemento backingMapPluginsCollections

O elemento backingMapPluginsCollections é um contêiner para todos os elementos backingMapPluginCollection. No arquivo companyGridProperty.xml na seção anterior, o elemento backingMapPluginCollections contém um elemento backingMapPluginCollection com o ID customerPlugins.

- Número de ocorrências: Zero para uma
- Elemento-filho: Elemento backingMapPluginCollection

Elemento backingMapPluginCollection

O elemento backingMapPluginCollection define os plug-ins do BackingMap e é identificado pelo atributo **id**. Especifique o atributo pluginCollectionRef para referenciar os plug-ins. Ao configurar vários plug-ins do BackingMaps de maneira semelhante, cada BackingMap pode referenciar o mesmo elemento backingMapPluginCollection.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Elemento bean

Atributos

id Identifica a backingMapPluginCollection e é referenciado pelo atributo pluginCollectionRef do elemento backingMap. Cada ID deve ser exclusivo. Se o valor de um atributo pluginCollectionRef não corresponder ao ID de um elemento backingMapPluginCollection, a validação XML falha. Qualquer número de elementos backingMap pode fazer referência a um único elemento backingMapPluginCollection. (Necessário)

```
<backingMapPluginCollection
(1)   id="id"
/>
```

No exemplo a seguir, o arquivo companyGridCollection.xml é utilizado para demonstrar como utilizar o elemento backingMapPluginCollection. Neste arquivo, o BackingMap do Cliente utilizar o customerPlugins backingMapPluginCollection para configurar o BackingMap do Cliente com um LRUEvictor. O Item e OrderLine BackingMaps referenciam o collection2 backingMapPluginCollection. Cada um destes BackingMaps possuem um conjunto de LFUEvictor.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Customer"
    pluginCollectionRef="customerPlugins"/>
   <backingMap name="Item" pluginCollectionRef="collection2"/>
   <backingMap name="OrderLine"
    pluginCollectionRef="collection2"/>
```

```
   <backingMap name="Order"/>
  </objectGrid>
 </objectGrids>
 <backingMapPluginCollections>
  <backingMapPluginCollection id="customerPlugins">
   <bean id="Evictor"
     className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor/>
  </backingMapPluginCollection>
  <backingMapPluginCollection id="collection2">
   <bean id="Evictor"
     className="com.ibm.websphere.objectgrid.plugins.builtins.LFUEvictor/>
   <bean id="OptimisticCallback"
     className="com.ibm.websphere.samples.objectgrid.EmployeeOptimisticCallBackImpl"/>
  </backingMapPluginCollection>
  </backingMapPluginCollections>
</objectGridConfig>
```

A amostra de código a seguir demonstra a abordagem programática para obter a mesma configuração que o arquivo `companyGridCollection.xml` no exemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();

ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
BackingMap customerMap = companyGrid.defineMap("Customer");
LRUEvictor customerEvictor = new LRUEvictor();
customerMap.setEvictor(customerEvictor);

BackingMap itemMap = companyGrid.defineMap("Item");
LFUEvictor itemEvictor = new LFUEvictor();
itemMap.setEvictor(itemEvictor);

BackingMap orderLineMap = companyGrid.defineMap("OrderLine");
LFUEvictor orderLineEvictor = new LFUEvictor();
orderLineMap.setEvictor(orderLineEvictor);

BackingMap orderMap = companyGrid.defineMap("Order");
```

Elemento querySchema

O elemento querySchema define relacionamentos entre BackingMaps e identifica o tipo de objeto em cada mapa. Estas informações são utilizadas pelo ObjectQuery para converter cadeias de linguagem de consulta em chamadas de acesso de mapa.

- Número de ocorrências: Zero para uma
- Elemento-filho: Elemento mapSchemas, elemento relationships

Elemento mapSchemas

Cada elemento querySchema tem um elemento mapSchemas que contém um ou mais elementos mapSchema.

- Número de ocorrências: Uma
- Elemento-filho: Elemento mapSchema

Elemento mapSchema

Um elemento mapSchema define o tipo de objeto que é armazenado em um BackingMap e instruções sobre como acessar os dados.

- Número de ocorrências: Uma ou mais
- Elemento-filho: Nenhum

Atributos

mapName
Especifica o nome do BackingMap a incluir no esquema. (Necessário)

valueClass
Especifica o tipo de objeto que é armazenado na parte do valor do BackingMap. (Necessário)

primaryKeyField
Especifica o nome do atributo-chave principal no atributo valueClass. A chave primária também deve ser armazenada na parte da chave do BackingMap. (Opcional)

accessType
Identifica como o mecanismo de consulta examina e acessa os dados persistentes nas instâncias do objeto valueClass. Se você configurar o valor como `FIELD`, os campos de classe são examinados e incluídos no esquema. Se o valor for `PROPERTY`, os atributos que estão associados com os métodos get e is são utilizados. O valor padrão é `PROPERTY`. (Opcional)

```
<mapSchema
(1)   mapName="backingMapName"
(2)   valueClass="com.mycompany.OrderBean"
(3)   primaryKeyField="orderId"
(4)   accessType="PROPERTY" | "FIELD"
/>
```

No exemplo a seguir, o arquivo `companyGridQuerySchemaAttr.xml` é utilizado para demonstrar uma configuração do mapSchema de amostra.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Order"/>
   <backingMap name="Customer"/>

   <querySchema>
    <mapSchemas>
     <mapSchema mapName="Order"
      valueClass="com.mycompany.OrderBean"
      primaryKeyField="orderNumber"
      accessType="FIELD"/>
     <mapSchema mapName="Customer"
      valueClass="com.mycompany.CustomerBean"
      primaryKeyField="id"
      accessType="FIELD"/>
    </mapSchemas>
   </querySchema>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

A amostra de código a seguir demonstra a abordagem programática para obter a mesma configuração do arquivo `companyGridQuerySchemaAttr.xml` no exemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
companyGrid.defineMap("Order");
companyGrid.defineMap("Customer");

// Define the schema
QueryConfig queryCfg = new QueryConfig();
queryCfg.addQueryMapping(new QueryMapping(
    "Order", OrderBean.class.getName(), "orderNumber", QueryMapping.FIELD_ACCESS));
queryCfg.addQueryMapping(new QueryMapping(
    "Customer", CustomerBean.class.getName(), "id", QueryMapping.FIELD_ACCESS));
companyGrid.setQueryConfig(queryCfg);
```

Elemento relationships

Cada elemento querySchema tem zero ou um elemento relationships que contém um ou mais elementos relationship.

- Número de ocorrências: Zero ou uma
- Elemento-filho: Elemento relationship

Elemento relationship

Um elemento relationship define o relacionamento entre dois BackingMaps e os atributos no atributo valueClass que liga o relacionamento.

- Número de ocorrências: Uma ou mais
- Elemento-filho: Nenhum

Atributos

source
Especifica o nome da valueClass do lado da origem de um relacionamento. (Necessário)

target
Especifica o nome da valueClass do lado do destino de um relacionamento. (Necessário)

relationField
Especifica o nome do atributo na valueClass de origem que faz referência ao destino. (Necessário)

invRelationField
Especifica o nome do atributo na valueClass de destino que faz referência à origem. Se este atributo não for especificado, o relacionamento é unidirecional. (Opcional)

```
<mapSchema
(1)   source="com.mycompany.OrderBean"
(2)   target="com.mycompany.CustomerBean"
(3)   relationField="customer"
(4)   invRelationField="orders"
/>
```

No exemplo a seguir, o arquivo `companyGridQuerySchemaWithRelationshipAttr.xml` é utilizado para demonstrar uma configuração de mapSchema de amostra que inclui um relacionamento bidirecional.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Order"/>
   <backingMap name="Customer"/>

   <querySchema>
    <mapSchemas>
     <mapSchema mapName="Order"
      valueClass="com.mycompany.OrderBean"
      primaryKeyField="orderNumber"
      accessType="FIELD"/>
     <mapSchema mapName="Customer"
      valueClass="com.mycompany.CustomerBean"
      primaryKeyField="id"
      accessType="FIELD"/>
    </mapSchemas>
    <relationships>
     <relationship
```

```
      source="com.mycompany.OrderBean"
      target="com.mycompany.CustomerBean"
      relationField="customer"/>
      invRelationField="orders"/>
    </relationships>
   </querySchema>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

A amostra de código a seguir demonstra a abordagem programática para obter a mesma configuração que o arquivo `companyGridQuerySchemaWithRelationshipAttr.xml` no exemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
companyGrid.defineMap("Order");
companyGrid.defineMap("Customer");

// Define the schema
QueryConfig queryCfg = new QueryConfig();
queryCfg.addQueryMapping(new QueryMapping(
    "Order", OrderBean.class.getName(), "orderNumber", QueryMapping.FIELD_ACCESS));
queryCfg.addQueryMapping(new QueryMapping(
    "Customer", CustomerBean.class.getName(), "id", QueryMapping.FIELD_ACCESS));
queryCfg.addQueryRelationship(new QueryRelationship(
     OrderBean.class.getName(), CustomerBean.class.getName(), "customer", "orders"));
companyGrid.setQueryConfig(queryCfg);
```

Elemento streamQuerySet

O elemento streamQuerySet é o elemento de nível superior para definir um conjunto de consultas de fluxo.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Elemento stream, elemento view

Elemento stream

O elemento stream representa um fluxo para o mecanismo de consulta do fluxo. Cada atributo do elemento stream corresponde a um método da interface StreamMetadata.

- Número de ocorrências: Uma para muitas
- Elemento-filho: Elemento basic

Atributos

name
Especifica o nome do fluxo. A validação falha se este atributo não for especificado. (Necessário)

valueClass
Especifica o tipo de classe do valor que é armazenado no ObjectMap do fluxo. O tipo de classe é utilizado para converter o objeto para os eventos de fluxo e para gerar uma instrução SQL se a instrução não for fornecida. (Necessário)

sql
Especifica a instrução SQL do fluxo. Se essa propriedade não for fornecida, um SQL de fluxo será gerado, refletindo os atributos ou métodos do acessador no atributo valueClass ou utilizando os atributos tuple do metadados da entidade. (Opcional)

access
Especifica o tipo para acessar os atributos da classe de valor. Se você configurar o valor para `FIELD`, os atributos serão recuperados diretamente a

partir dos campos usando o reflexo Java. Caso contrário, os métodos do acessador serão utilizados para ler os atributos. O valor padrão é PROPERTY. (Opcional)

```
<stream
(1)   name="streamName"
(2)   valueClass="streamMapClassType"
(3)   sql="streamSQL create stream stockQuote
     keyed by t ( transactionvolume INTEGER, price DECIMAL (9,2), issue VARCHAR(100) );"
(4)   access="PROPERTY" | "FIELD"
/>
```

Elemento view

O elemento view representa uma visualização de consulta de fluxo. Cada elemento stream corresponde a um método na interface ViewMetadata.

- Número de ocorrências: Uma para muitas
- Elemento-filho: Elemento basic, elemento id

Atributos

name
Especifica o nome da visualização. A validação falha se este atributo não for especificado. (Necessário)

sql
Especifica o SQL do fluxo, que define a transformação da visualização. A validação falha se este atributo não for especificado. (Necessário)

valueClass
Especifica o tipo de classe do valor que é armazenado nesta visualização do ObjectMap. O tipo de classe é utilizado para converter eventos de visualização no formato de tupla correto que é compatível com este tipo de classe. Se o tipo de classe não for fornecido, um formato padrão após as definições da coluna em SPTSQL (Stream Processing Technology Structured Query Language) será utilizado. Se os metadados de uma entidade forem definidos para esse mapa de visualização, não utilize o atributo valueClass. (Opcional)

access
Especifica o tipo para acessar os atributos da classe de valor. Se você configurar o tipo de acesso para FIELD, os valores de coluna serão configurados diretamente para os campos usando o reflexo Java. Caso contrário, os métodos do acessador são utilizados para configurar os atributos. O valor padrão é PROPERTY. (Opcional)

```
<view
(1)   name="viewName"
(2)   valueClass="viewMapValueClass"
(3)   sql="viewSQL CREATE VIEW last5MinuteAvgPrice AS
      SELECT issue, avg(price) as totalVolume
        FROM (SELECT * FROM stockQuote FETCH LATEST 5 MINUTES) group by issue;"/>
(4)   access="PROPERTY" | "FIELD"
/>
```

Elemento basic

O elemento basic é utilizado para definir um mapeamento a partir do nome do atributo na classe de valor ou metadados da entidade para a coluna que é definida no SPTSQL.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

```
<basic
(1)   name="attributeName"
(2)   column="columnName"
/>
```

Elemento ID

O elemento id é utilizado para um mapeamento de atributo-chave.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

```
<id
(1)   name="idName"
(2)   column="columnName"
/>
```

No exemplo a seguir, o arquivo `StreamQueryApp2.xml` é utilizado para demonstrar como configurar os atributos de um streamQuerySet. O conjunto de consultas de fluxo _stockQuoteSQS_ possui um fluxo e uma visualização. O fluxo e a visualização definem seu nome, valueClass, sql e tipo de acesso. O fluxo também define um elemento básico, que especifica que o atributo volume na classe StockQuote é mapeado para o volume de transação da coluna SQL que é definida na instrução SQL.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
 <objectGrids>
  <objectGrid name="og1">
   <backingMap name="stockQuote" readOnly="false" copyKey="true" streamRef="stockQuote"/>
   <backingMap name="last5MinuteAvgPrice" readOnly="false" copyKey="false" viewRef="last5MinuteAvgPrice"/>

   <streamQuerySet name="stockQuoteSQS">
    <stream
     name="stockQuote"
     valueClass="com.ibm.ws.objectgrid.streamquery.sample.guide.StockQuote"
     sql="create stream stockQuote
         keyed by t ( transactionvolume INTEGER, price DECIMAL (9,2), issue VARCHAR(100) );"
     access="FIELD">
     <basic name="volume" column="transactionvolume"/>
    </stream>

    <view
     name="last5MinuteAvgPrice"
     valueClass="com.ibm.ws.objectgrid.streamquery.sample.guide.AveragePrice"
     sql="CREATE VIEW last5MinuteAvgPrice AS SELECT issue, avg(price) as avgPrice
       FROM (SELECT * FROM stockQuote FETCH LATEST 5 MINUTES) group by issue;"
     access="FIELD"
    </view>
   </streamQuerySet>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

Arquivo `objectGrid.xsd`

Use o esquema XML do descritor do ObjectGrid para configurar o WebSphere eXtreme Scale.

Consulte o “Arquivo XML descritor do ObjectGrid” na página 141 para obter descrições dos elementos e atributos definidos no arquivo `objectGrid.xsd`. Para obter mais informações sobre o arquivo `objectgrid.xsd` Spring, consulte “Arquivo XML descritor do Spring” na página 273.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:cc="http://ibm.com/ws/objectgrid/config"
xmlns:dgc="http://ibm.com/ws/objectgrid/config"
elementFormDefault="qualified"
```

```
targetNamespace="http://ibm.com/ws/objectgrid/config">

 <xsd:element name="objectGridConfig">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element maxOccurs="1" minOccurs="1" name="objectGrids" type="dgc:objectGrids">
     <xsd:unique name="objectGridNameUnique">
      <xsd:selector xpath="dgc:objectGrid"/>
      <xsd:field xpath="@name" />
     </xsd:unique>
    </xsd:element>
    <xsd:element maxOccurs="1" minOccurs="0" name="backingMapPluginCollections"
       type="dgc:backingMapPluginCollections"/>
   </xsd:sequence>
  </xsd:complexType>

  <xsd:key name="backingMapPluginCollectionId">
   <xsd:selector xpath="dgc:backingMapPluginCollections/dgc:backingMapPluginCollection"/>
   <xsd:field xpath="@id"/>
  </xsd:key>

  <xsd:keyref name="pluginCollectionRef" refer="dgc:backingMapPluginCollectionId">
   <xsd:selector xpath="dgc:objectGrids/dgc:objectGrid/dgc:backingMap"/>
   <xsd:field xpath="@pluginCollectionRef"/>
  </xsd:keyref>

  <xsd:key name="streamName">
   <xsd:selector xpath="dgc:objectGrids/dgc:objectGrid/dgc:streamQuerySet/dgc:stream"/>
   <xsd:field xpath="@name" />
  </xsd:key>

  <xsd:keyref name="streamRef" refer="dgc:streamName">
   <xsd:selector xpath="dgc:objectGrids/dgc:objectGrid/dgc:backingMap"/>
   <xsd:field xpath="@streamRef"/>
  </xsd:keyref>

  <xsd:key name="viewName">
   <xsd:selector xpath="dgc:objectGrids/dgc:objectGrid/dgc:streamQuerySet/dgc:view"/>
   <xsd:field xpath="@name" />
  </xsd:key>

  <xsd:keyref name="viewRef" refer="dgc:viewName">
   <xsd:selector xpath="dgc:objectGrids/dgc:objectGrid/dgc:backingMap"/>
   <xsd:field xpath="@viewRef"/>
  </xsd:keyref>
 </xsd:element>

 <xsd:complexType name="objectGrids">
  <xsd:sequence>
   <xsd:element maxOccurs="unbounded" minOccurs="1" name="objectGrid" type="dgc:objectGrid">
    <xsd:unique name="backingMapNameUnique">
     <xsd:selector xpath="dgc:backingMap"/>
     <xsd:field xpath="@name" />
    </xsd:unique>
    <xsd:unique name="streamQuerySetNameUnique">
     <xsd:selector xpath="dgc:streamQuerySet"/>
     <xsd:field xpath="@name" />
    </xsd:unique>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="backingMapPluginCollections">
  <xsd:sequence>
   <xsd:element maxOccurs="unbounded" minOccurs="0" name="backingMapPluginCollection"
type="dgc:backingMapPluginCollection"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="objectGrid">
  <xsd:sequence>
   <xsd:element maxOccurs="unbounded" minOccurs="0" name="bean" type="dgc:bean"/>
   <xsd:element maxOccurs="unbounded" minOccurs="0" name="backingMap" type="dgc:backingMap"/>
   <xsd:element maxOccurs="1" minOccurs="0" name="querySchema" type="dgc:querySchema"/>
   <xsd:element maxOccurs="unbounded" minOccurs="0" name="streamQuerySet"
      type="dgc:streamQuerySet">
    <xsd:unique name="stream">
     <xsd:selector xpath="dgc:stream"/>
     <xsd:field xpath="@name" />
    </xsd:unique>
```

```
    <xsd:unique name="view">
     <xsd:selector xpath="dgc:view"/>
     <xsd:field xpath="@name" />
    </xsd:unique>
   </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="name" type="xsd:string" use="required" />
  <xsd:attribute name="authorizationMechanism"
type="dgc:authorizationMechanism"
    use="optional" />
  <xsd:attribute name="accessByCreatorOnlyMode"
type="dgc:accessByCreatorOnlyMode"
    use="optional" />
  <xsd:attribute name="securityEnabled" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="txTimeout" type="xsd:int" use="optional"/>
  <xsd:attribute name="permissionCheckPeriod" type="xsd:int" use="optional"/>
  <xsd:attribute name="entityMetadataXMLFile" type="xsd:string" use="optional"/>
  <xsd:attribute name="initialState" type="dgc:initialState" use="optional"/>
 </xsd:complexType>


 <xsd:complexType name="backingMap">
  <xsd:sequence>
   <xsd:element maxOccurs="1" minOccurs="0" name="timeBasedDBUpdate" type="dgc:timeBasedDBUpdate"/>
  </xsd:sequence>
  <xsd:attribute name="name" type="xsd:string" use="required" />
  <xsd:attribute name="readOnly" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="pluginCollectionRef" type="xsd:string" use="optional"/>
  <xsd:attribute name="preloadMode" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="lockStrategy" type="dgc:lockStrategy" use="optional"/>
  <xsd:attribute name="copyMode" type="dgc:copyMode" use="optional"/>
  <xsd:attribute name="valueInterfaceClassName" type="xsd:string" use="optional"/>
  <xsd:attribute name="numberOfBuckets" type="xsd:int" use="optional"/>
  <xsd:attribute name="nullValuesSupported" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="lockTimeout" type="xsd:int" use="optional"/>
  <xsd:attribute name="numberOfLockBuckets" type="xsd:int" use="optional"/>
  <xsd:attribute name="copyKey" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="timeToLive" type="xsd:int" use="optional"/>
  <xsd:attribute name="ttlEvictoryType" type="dgc:ttlEvictorType" use="optional"/>
  <xsd:attribute name="streamRef" type="xsd:string" use="optional"/>
  <xsd:attribute name="viewRef" type="xsd:string" use="optional"/>
  <xsd:attribute name="writeBehind" type="xsd:string" use="optional"/>
  <xsd:attribute name="evictionTriggers" type="xsd:string" use="optional"/>
  <xsd:attribute name="template" type="xsd:boolean" use="optional"/>
 </xsd:complexType>

 <xsd:complexType name="bean">
  <xsd:sequence>
   <xsd:element maxOccurs="unbounded" minOccurs="0" name="property" type="dgc:property"/>
  </xsd:sequence>
  <xsd:attribute name="className" type="xsd:string" use="required" />
  <xsd:attribute name="id" type="dgc:beanId" use="required"/>
 </xsd:complexType>

 <xsd:complexType name="backingMapPluginCollection">
  <xsd:sequence>
   <xsd:element maxOccurs="unbounded" minOccurs="0" name="bean" type="dgc:bean"/>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:string" use="required"/>
 </xsd:complexType>

 <xsd:complexType name="property">
  <xsd:attribute name="name" type="xsd:string" use="required" />
  <xsd:attribute name="value" type="xsd:string" use="required" />
  <xsd:attribute name="type" type="dgc:propertyType" use="required"/>
  <xsd:attribute name="description" type="xsd:string" use="optional" />
 </xsd:complexType>

 <xsd:simpleType name="propertyType">
  <xsd:restriction base="xsd:string">
   <xsd:enumeration value="java.lang.Boolean"/>
   <xsd:enumeration value="boolean"/>
   <xsd:enumeration value="java.lang.String"/>
   <xsd:enumeration value="java.lang.Integer"/>
   <xsd:enumeration value="int"/>
   <xsd:enumeration value="java.lang.Double"/>
   <xsd:enumeration value="double"/>
   <xsd:enumeration value="java.lang.Byte"/>
   <xsd:enumeration value="byte"/>
   <xsd:enumeration value="java.lang.Short"/>
```

```
  <xsd:enumeration value="short"/>
  <xsd:enumeration value="java.lang.Long"/>
  <xsd:enumeration value="long"/>
  <xsd:enumeration value="java.lang.Float"/>
  <xsd:enumeration value="float"/>
  <xsd:enumeration value="java.lang.Character"/>
  <xsd:enumeration value="char"/>
 </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="beanId">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="TransactionCallBack"/>
  <xsd:enumeration value="ObjectGridEventListener"/>
  <xsd:enumeration value="SubjectSource"/>
  <xsd:enumeration value="MapAuthorization"/>
  <xsd:enumeration value="SubjectValidation"/>
  <xsd:enumeration value="ObjectGridAuthorization"/>

  <xsd:enumeration value="Loader"/>
  <xsd:enumeration value="ObjectTransformer"/>
  <xsd:enumeration value="OptimisticCallback"/>
  <xsd:enumeration value="Evictor"/>
  <xsd:enumeration value="MapEventListener"/>
  <xsd:enumeration value="MapIndexPlugin"/>

 </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="copyMode">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="COPY_ON_READ_AND_COMMIT"/>
  <xsd:enumeration value="COPY_ON_READ"/>
  <xsd:enumeration value="COPY_ON_WRITE"/>
  <xsd:enumeration value="NO_COPY"/>
  <xsd:enumeration value="COPY_TO_BYTES"/>
 </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="lockStrategy">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="OPTIMISTIC"/>
  <xsd:enumeration value="PESSIMISTIC"/>
  <xsd:enumeration value="NONE"/>
 </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="ttlEvictorType">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="CREATION_TIME"/>
  <xsd:enumeration value="LAST_ACCESS_TIME"/>
      <xsd:enumeration value="LAST_UPDATE_TIME"/>
  <xsd:enumeration value="NONE"/>
 </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="authorizationMechanism">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="AUTHORIZATION_MECHANISM_JAAS" />
  <xsd:enumeration value="AUTHORIZATION_MECHANISM_CUSTOM" />
 </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="accessByCreatorOnlyMode">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="disabled"/>
  <xsd:enumeration value="complement"/>
  <xsd:enumeration value="supersede"/>
 </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="streamQuerySet">
 <xsd:sequence>
  <xsd:element maxOccurs="unbounded" minOccurs="0" name="stream" type="dgc:stream">
   <xsd:unique name="streamBasicColumnUnique">
    <xsd:selector xpath="dgc:basic"/>
    <xsd:field xpath="@column"/>
   </xsd:unique>
  </xsd:element>
  <xsd:element maxOccurs="unbounded" minOccurs="0" name="view" type="dgc:view">
```

```
    <xsd:unique name="viewBasicColumnUnique">
     <xsd:selector xpath="dgc:basic"/>
     <xsd:field xpath="@column"/>
    </xsd:unique>
   </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="name" type="xsd:string" use="required" />
  <xsd:attribute name="viewResultsToListenersOnly" type="xsd:boolean" default="false" use="optional"/>
  <xsd:attribute name="deployInPrimaryOnly" type="xsd:boolean"
default="true"
     use="optional" />
 </xsd:complexType>

 <xsd:complexType name="stream">
  <xsd:sequence>
   <xsd:element maxOccurs="unbounded" minOccurs="0" name="basic" type="dgc:basic"/>
  </xsd:sequence>
  <xsd:attribute name="valueClass" type="xsd:string" use="required"/>
  <xsd:attribute name="name" type="xsd:string" use="required" />
  <xsd:attribute name="sql" type="xsd:string" use="optional"/>
  <xsd:attribute name="access" type="cc:accessType" use="optional"/>
 </xsd:complexType>

 <xsd:complexType name="view">
  <xsd:sequence>
   <xsd:element maxOccurs="unbounded" minOccurs="0" name="id" type="dgc:basic"/>
   <xsd:element element maxOccurs="unbounded" minOccurs="0" name="basic"
     type="dgc:basic"/>
  </xsd:sequence>
  <xsd:attribute name="name" type="xsd:string" use="required" />
  <xsd:attribute name="sql" type="xsd:string" use="optional"/>
  <xsd:attribute name="valueClass" type="xsd:string" use="optional"/>
  <xsd:attribute name="access" type="cc:accessType" use="optional"/>
 </xsd:complexType>

 <xsd:complexType name="basic">
  <xsd:attribute name="name" type="xsd:string" use="required" />
  <xsd:attribute name="column" type="xsd:string" use="required"/>
 </xsd:complexType>

 <xsd:complexType name="id">
  <xsd:attribute name="name" type="xsd:string" use="required" />
  <xsd:attribute name="column" type="xsd:string" use="required"/>
 </xsd:complexType>

 <xsd:complexType name="timeBasedDBUpdate">
  <xsd:attribute name="persistenceUnitName" type="xsd:string" use="optional"/>
  <xsd:attribute name="mode" type="cc:dbUpdateMode" use="optional"/>
  <xsd:attribute name="timestampField" type="xsd:string" use="optional"/>
  <xsd:attribute name="entityClass" type="xsd:string" use="required"/>
  <xsd:attribute name="jpaPropertyFactory" type="xsd:string" use="optional"/>
 </xsd:complexType>

 <xsd:simpleType name="dbUpdateMode">
  <xsd:restriction base="xsd:string"/>
   <xsd:enumeration value="INVALIDATE_ONLY"/>
   <xsd:enumeration value="UPDATE_ONLY"/>
   <xsd:enumeration value="INSERT_UPDATE"/>
  </xsd:restriction>
 </xsd:simpleType>

 <xsd:complexType name="querySchema">
  <xsd:sequence>
   <xsd:element maxOccurs="1" minOccurs="1" name="mapSchemas" type="dgc:mapSchemas">
    <xsd:unique name="mapNameUnique">
     <xsd:selector xpath="dgc:mapSchema"/>
     <xsd:field xpath="@mapName"/>
    </xsd:unique>
   </xsd:element>
   <xsd:element maxOccurs="1" minOccurs="0" name="relationships" type="dgc:relationships"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="mapSchemas">
  <xsd:sequence>
   <xsd:element maxOccurs="unbounded" minOccurs="1" name="mapSchema" type="dgc:mapSchema"/>
  </xsd:sequence>
 </xsd:complexType>

 <xsd:complexType name="relationships">
```

```
 <xsd:sequence>
  <xsd:element maxOccurs="unbounded" minOccurs="1" name="relationship" type="dgc:relationship"/>
 </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="mapSchema">
 <xsd:attribute name="mapName" type="xsd:string" use="required"/>
 <xsd:attribute name="valueClass" type="xsd:string" use="required"/>
 <xsd:attribute name="primaryKeyField" type="xsd:string" use="optional"/>
 <xsd:attribute name="accessType" type="cc:accessType" use="optional"/>
</xsd:complexType>

<xsd:complexType name="relationship">
 <xsd:attribute name="source" type="xsd:string" use="required"/>
 <xsd:attribute name="target" type="xsd:string" use="required"/>
 <xsd:attribute name="relationField" type="xsd:string" use="required"/>
 <xsd:attribute name="invRelationField" type="xsd:string" use="optional"/>
</xsd:complexType>

<xsd:simpleType name="accessType">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="PROPERTY"/>
  <xsd:enumeration value="FIELD"/>
 </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="initialState">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="OFFLINE"/>
  <xsd:enumeration value="PRELOAD"/>
  <xsd:enumeration value="ONLINE"/>
 </xsd:restriction>
</xsd:simpleType>

</xsd:schema>
```

Configurando Políticas de Implementação

Use o arquivo XML do descritor de política de implementação e o arquivo XML do descritor de objectgrid para gerenciar uma topologia distribuída. A política de implementação é codificada como um arquivo XML que é fornecido para o contêiner do eXtreme Scale. A política de implementação fornece informações sobre mapas, conjuntos de mapas, partições, réplicas e assim por diante. Ela também controla os comportamentos do posicionamento de shards.

Configurando Implementações Distribuídas

Utilize o arquivo XML do descritor de política de implementação e o arquivo XML do descritor de objectgrid para gerenciar sua topologia.

A política de implementação é codificada como um arquivo XML que é fornecido para o contêiner do eXtreme Scale. O arquivo XML especifica as seguintes informações:

- Os mapas que pertencem a cada conjunto de mapas.
- O número de partições
- O número de réplicas síncronas e assíncronas

Para obter informações sobre como iniciar servidores de contêiner, leia sobre como iniciar contêineres do eXtreme Scale automaticamente ou como iniciar processos do contêiner.

A política de implementação também controla os seguintes comportamentos de colocação.

- O número mínimo de contêineres ativos antes da ocorrência da disposição
- A substituição automática de shards perdidos

- A disposição de cada shard de uma única partição em uma máquina diferente.

Para obter mais informações sobre a configuração de política, leia sobre o arquivo XML do descritor de política de implementação.

As informações de terminal não estão pré-configuradas no ambiente dinâmico. Não há nomes de servidor ou informações de topologia física localizadas na política de implementação. Todos os shards em uma grade são automaticamente colocados em contêineres pelo serviço de catálogo. O serviço de catálogo usa as restrições que são definidas pela política de implementação para gerenciar automaticamente a colocação de shard. Essa colocação automática de shard facilita a configuração de grades maiores. Também é possível incluir servidores no seu ambiente, conforme necessário.

Restrição: Em um ambiente do WebSphere Application Server, um tamanho de grupo principal de mais de 50 membros não é suportado.

Um arquivo XML de política de implementação é passado para um contêiner eXtreme Scale durante a inicialização. Uma política de implementação deve ser usada junto com o arquivo XML de ObjectGrid. A política de implementação não é requerida para iniciar um contêiner, mas é recomendada. A política de implementação deve ser compatível com o arquivo XML do ObjectGrid que é utilizado com ela. Para cada elemento objectgridDeployment na política de implementação, você deve incluir um elemento objectGrid correspondente no arquivo XML de ObjectGrid. Os mapas no objectgridDeployment devem ser consistentes com os elementos backingMap localizados no XML do ObjectGrid. Cada backingMap deve ser referido em um, e apenas um, elemento mapSet.

No exemplo a seguir, o arquivo `companyGridDpReplication.xml` é destinado a formar um par com o arquivo `companyGrid.xml` correspondente.

companyGridDpReplication.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org./2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

 <objectgridDeployment objectgridName="CompanyGrid">
  <mapSet name="mapSet1" numberOfPartitions="11"
   minSyncReplicas="1" maxSyncReplicas="1"
   maxAsyncReplicas="0" numInitialContainers="4">
   <map ref="Customer" />
   <map ref="Item" />
   <map ref="OrderLine" />
   <map ref="Order" />
  </mapSet>
 </objectgridDeployment>

</deploymentPolicy>
```

companyGrid.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Customer"/>
   <backingMap name="Item" />
   <backingMap name="OrderLine" />
   <backingMap name="Order"/>
  </objectGrid>
 </objectGrids>

</objectGridConfig>
```

O arquivo `companyGridDpReplication.xml` tem um elemento mapSet que é dividido em 11 partições. Cada partição deve ter exatamente uma réplica síncrona. O número de réplicas síncronas é especificado pelos atributos minSyncReplicas e maxSyncReplicas. Como o atributo minSyncReplicas é configurado como 1, cada

partição no elemento mapSet deve ter pelo menos uma réplica síncrona disponível para processar transações de gravação. Como maxSyncReplicas é configurado para 1, cada partição não pode exceder uma réplica síncrona. As partições nesse elemento mapSet não têm réplicas assíncronas.

O atributo numInitialContainers instrui o serviço de catálogo para adiar o posicionamento até que quatro contêineres estejam disponíveis para suportar essa instância de ObjectGrid. O atributo numInitialContainers será ignorado depois que o número especificado de contêineres for alcançado.

Embora o arquivo `companyGridDpReplication.xml` seja um exemplo básico, uma política de implementação pode oferecer controle total sobre seu ambiente do eXtreme Scale. Leia sobre arquivo XML do descritor da política de implementação.

Topologia Distribuída

Caches consistentes distribuídos oferecem melhores desempenho, disponibilidade e escalabilidade que podem ser configurados.

O WebSphere eXtreme Scale equilibra automaticamente os servidores. É possível incluir servidores adicionais sem reiniciar o WebSphere eXtreme Scale. Incluir servidores adicionais sem precisar reiniciar o eXtreme Scale permite executar implementações simples e implementações que atingem terabytes em que milhares de servidores são necessários.

Essa topologia de implementação é flexível. Utilizando o serviço de catálogo, é possível incluir e remover servidores para utilizar melhor os recursos sem remover o cache inteiro. É possível utilizar os comandos startOgServer e stopOgServer para iniciar e parar servidores de contêiner. Ambos os comandos requerem que você especifique a opção -catalogServiceEndPoints. Todos os clientes de topologia distribuída se comunicam com o serviço de catálogo por meio do Internet Interoperability Object Protocol (IIOP). Todos os clientes utilizam a interface ObjectGrid para comunicar-se com servidores.

O recurso de configuração dinâmica do WebSphere eXtreme Scale facilita incluir recursos no sistema. Contêineres hospedam os dados, e o serviço de catálogo permite que clientes se comuniquem com a grade de contêineres. O serviço de catálogo encaminha pedidos, aloca espaço em contêineres de host e gerencia o funcionamento e a disponibilidade do sistema geral. Os clientes se conectam a um serviço de catálogo, recuperam uma descrição da topologia de contêiner-servidor e, em seguida, comunica-se diretamente a cada servidor, conforme necessário. Quando a topologia do servidor muda devido à inclusão de novos servidores, ou devido à falha de outros, o serviço de catálogo roteia automaticamente os pedidos do cliente para o servidor apropriado que hospeda os dados.

Um serviço de catálogo normalmente existe na sua própria grade do Java Virtual Machines. Um único servidor de catálogos pode gerenciar vários servidores. É possível iniciar um contêiner em uma JVM sozinho ou carregar o contêiner em uma JVM arbitrária com outros contêineres para servidores diferentes. Pode existir um cliente em alguma JVM e comunicar-se com um ou mais servidores. Um cliente também pode existir na mesma JVM como um contêiner.

Também é possível criar uma política de implementação programaticamente ao integrar um contêiner em um processo ou aplicativo Java existente. Para obter mais informações, consulte a documentação da API DeploymentPolicy do eXtreme Scale.

Configurando Zonas para a Colocação de Réplica

O suporte à zona permite configurações sofisticadas para posicionamento de réplica nos datacenters. Com esse recurso, as grades de milhares de partições podem ser facilmente gerenciadas utilizando algumas regras de localização opcionais. Um datacenter pode estar em diferentes andares de um prédio, diferentes prédios ou até mesmo cidades diferentes ou outras distinções, conforme configurado com as regras de zonas.

Flexibilidade de Zonas

É possível colocar shards em zonas. Esta função permite que você tenha mais controle sobre como o eXtreme Scale coloca shards em uma grade. A Java Virtual Machines que hospeda um servidor eXtreme Scale pode ser identificada com um identificador de zona. O arquivo de implementação agora pode incluir uma ou mais regras de zonas e estas regras de zonas estão associadas com um tipo shard. A melhor maneira de explicar isto é com alguns exemplos acompanhados por mais detalhes.

Controle de zonas de disposição de como o eXtreme Scale designa primários e réplicas para configurar topologias avançadas.

Um Java Virtual Machine pode ter vários contêineres, mas apenas 1 servidor. Um contêiner pode hospedar vários shards a partir de um único ObjectGrid.

Este recurso é útil para certificar-se de que as réplicas e os primários sejam colocados em diferentes locais ou zonas para obter uma melhor alta disponibilidade. Normalmente, o eXtreme Scale não coloca um shard primário e de réplica na Java Virtual Machines com o mesmo endereço IP. Esta regra simples normalmente evita que dois servidores eXtreme Scale sejam colocados no mesmo computador físico. Entretanto, pode ser necessário um mecanismo mais flexível. Por exemplo, é possível estar utilizando dois chassis blade e desejar que os shards primários sejam *divididos* em ambos os chassis e que a réplica para cada shard primário seja colocada no outro chassi a partir do primário.

Dividido significa que os shards primários são colocados em cada zona e a réplica para cada primário está localizada na zona oposta. O primário 0, por exemplo, poderia estar na zoneA, a a réplica síncrona 0 poderia estar na zoneB. O primário 1 poderia estar na zoneB, e a réplica síncrona 1 poderia estar na zoneA.

O nome do chassi poderia ser o nome da zona neste caso. Alternativamente, é possível chamar as zonas de acordo com os andares em um prédio e utilizar as zonas para ter certeza de que os primários e as réplicas para os mesmos dados estejam em andares diferentes. Prédios e datacenter também são possíveis. Foram feitos testes em datacenters utilizando zonas como um mecanismo para garantir que os dados sejam adequadamente replicados entre os datacenters. Se você estiver utilizando o HTTP Session Manager para eXtreme Scale, também é possível utilizar zonas. Com este recurso, é possível implementar um aplicativo da Web único em três datacenters e garantir que as sessões HTTP para usuários sejam replicas nos datacenters para que as sessões possam ser recuperadas, mesmo se um datacenters inteiro falhar.

O WebSphere eXtreme Scale está ciente da necessidade de gerenciar uma grade grande sobre vários datacenters. Isso pode assegurar que os primários e os backups da mesma partição estejam localizados em datacenters diferentes, se isto for necessário. Ele pode colocar todos os primários no datacenter 1 e todas as

réplicas no datacenter 2, ou pode fazer um round robin com os primários e as réplica em ambos os datacenters. As regras são flexíveis de modo que muitos cenários são possíveis. O eXtreme Scale também pode gerenciar milhares de servidores, que juntos com a disposição completamente automática e o reconhecimento do datacenter tornam tais grades grandes financeiramente suportáveis de um ponto de vista administrativo. Os administradores podem especificar o que desejam fazer de maneira simples e eficiente.

Como um administrador, utilize zonas de disposição para controlar onde os shards primários e de réplica são colocados, o que permite a configuração de topologias avançadas de alto desempenho e altamente disponíveis. É possível definir uma zona para qualquer agrupamento lógico de processos do eXtreme Scale, conforme observado acima: Estas zonas podem corresponder a locais de estação de trabalho físicas, tais como um datacenter, um andar de um datacenter ou um chassi blade. É possível dividir dados em zonas, o que fornece aumento de disponibilidade ou você dividir os primários e as réplicas em zonas separadas quando um hot standby é necessário.

Associando um Servidor eXtreme Scale com um Zona que não está utilizando o WebSphere Extended Deployment

Se o eXtreme Scale for utilizado com Java Standard Edition ou um servidor de aplicativos que não seja baseado no WebSphere Extended Deployment versão 6.1, um JVM que é um contêiner de shard pode ser associado a uma zona se estiver utilizando as seguintes técnicas.

Aplicativos utilizando o script startOgServer

O script startOgServer é utilizado para iniciar um aplicativo eXtreme Scale quando ele não está sendo integrado em um servidor existente. O parâmetro **-zone** é utilizado para especificar a zona a utilizar para todos os contêineres no servidores.

Especificando a zona ao iniciar um contêiner utilizando APIs

Associando Nós do WebSphere Extended Deployment com Zonas

Se estiver utilizando eXtreme Scale com aplicativos WebSphere Extended Deployment JEE, será possível alavancar os grupos de nós do WebSphere Extended Deployment para colocar os servidores em zonas específicas.

No eXtreme Scale, um JVM só pode ser membro de uma única zona. Entretanto, o WebSphere permite que um nó faça parte de vários grupos de nós. É possível utilizar essa funcionalidade de zonas do eXtreme Scale se tiver certeza de que cada um de seus nós esteja em apenas um grupo de nós da zona.

Utilize a seguinte sintaxe para nomear o grupo de nós para declará-lo em uma zona: `ReplicationZone<UniqueSuffix>`. Servidores em execução em um nó que faz parte desse grupo de nós são incluídos nas zonas especificadas pelo nome do grupo de nós. A seguir está uma descrição de uma topologia de exemplo.

Primeiro, você configura 4 nós: node1, node2, node3 e node4, cada um com 2 servidores. Em seguida, você cria um grupo de nós denominado ReplicationZoneA e um grupo de nós denominado ReplicationZoneB. Em seguida, você inclui node1 e node2 em ReplicationZoneA e inclui node3 e node4 em ReplicationZoneB.

Quando os servidores no node1 e node2 são iniciados, eles se tornam parte de ReplicationZoneA; da mesma forma, os servidores em node3 e node4 se tornarão parte de ReplicationZoneB.

Uma JVM membro da grade verifica a associação da zona apenas na inicialização. A inclusão de um novo grupo de nós ou a mudança da associação têm impacto apenas nas JVMs recém iniciadas ou reiniciadas.

Regras de Zonas

Uma partição do eXtreme Scale possui um shard primários e zero ou mais shards de réplica. Para este exemplo, considere a seguinte convenção de nomenclatura para estes shards. `P` é o shard primário, `S` é uma réplica assíncrona e `A` é uma réplica assíncrona. Uma regra de zona possui três componentes:

- Um nome de regra
- Uma lista de zonas
- Um sinalizador inclusivo ou exclusivo

O nome da zona para um contêiner pode ser especificado conforme descrito na documentação API do servidor integrada. Uma regra de zona especifica o possíveis conjuntos de zonas nos quais um shard pode ser colocado. O sinalizador inclusivo indica que após um shard ser colocado na lista, então, todos os outros shards também são colocados em tal zona. Uma configuração exclusiva indica que cada shard para uma partição é colocada em uma zona diferente na lista de zonas. Por exemplo, utilizar uma configuração exclusiva significa que se houver três shards (um primário e duas réplicas síncronas), então, a lista de zonas deve ter três zonas.

Cada shard pode ser associado com uma regra de zona. Uma regra de zona pode ser compartilhada entre dois shards. Quando uma regra é compartilhada, então o sinalizador inclusivo ou exclusivo é estendido pelos shards de todos os tipos compartilhando uma regra única.

Exemplos

A seguir, está um conjunto de exemplos mostrando diversos cenários e a configuração de implementação para implementar os cenários.

Dividindo primários e réplicas entre zonas

É possível ter três chassis blade e desejar primários distribuídos para os três, com uma réplica síncrona única colocada em um chassi diferente do primário. Defina cada chassis como uma zona com nomes de chassi `ALPHA`, `BETA` e `GAMMA`. A seguir, está um exemplo de XML de implementação:

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
 xsi:schemaLocation=
 "http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
    xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">
  <objectgridDeployment objectgridName="library">
   <mapSet name="ms1" numberOfPartitions="37" minSyncReplicas="1"
    maxSyncReplicas="1" maxAsyncReplicas="0">
   <map ref="book" />
   <zoneMetadata>
    <shardMapping shard="P" zoneRuleRef="stripeZone"/>
    <shardMapping shard="S" zoneRuleRef="stripeZone"/>
    <zoneRule name ="stripeZone" exclusivePlacement="true" >
     <zone name="ALPHA" />
     <zone name="BETA" />
     <zone name="GAMMA" />
    </zoneRule>
```

```
   </zoneMetadata>
  </mapSet>
 </objectgridDeployment>
</deploymentPolicy>
```

Este XML de implementação contém uma grade chamada library com um Mapa único denominado book. Ele utiliza quatro partições com uma única réplica síncrona. A cláusula de metadados de zona mostra a definição de uma única regra de zona e a associação das regras de zona com os shards. Os shards primários e síncronos estão ambos associados com a regra de zona "stripeZone". A regra de zona possui as três zonas e utiliza a disposição exclusiva. Esta regra significa que se o primário para a partição 0 for colocado no ALPHA, então, a réplica para a partição 0 será colocada no BETA ou GAMMA. De maneira semelhante, os primários para outras partições são colocados em outras zonas e as réplicas serão colocadas.

Réplica assíncrona em uma zona diferente da primária e réplica síncrona

Neste exemplo, existem dois prédios com uma alta conexão de latência entre eles. Você não deseja alta disponibilidade de perda de dados para todos os cenários. Entretanto, o impacto de desempenho da replicação síncrona entre os prédios o leva a um dilema. Você deseja um primário com réplica síncrona em um prédio e uma réplica assíncrona em outro prédio. Normalmente, as falhas são paralisações da JVM ou falhas de computador ao invés de problemas de larga escala. Com esta topologia, é possível sobrevier a falhas normais sem nenhuma perda de dados. A perda de um prédio é rara o suficiente que alguma perda de dados é aceitável neste caso. É possível criar duas zonas, uma para cada prédio. A seguir, está o arquivo XML de implementação:

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
  xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

 <objectgridDeployment objectgridName="library">
  <mapSet name="ms1" numberOfPartitions="13" minSyncReplicas="1"
   maxSyncReplicas="1" maxAsyncReplicas="1">
   <map ref="book" />
   <zoneMetadata>
    <shardMapping shard="P" zoneRuleRef="primarySync"/>
    <shardMapping shard="S" zoneRuleRef="primarySync"/>
    <shardMapping shard="A" zoneRuleRef="aysnc"/>
    <zoneRule name ="primarySync" exclusivePlacement="false" >
      <zone name="BldA" />
      <zone name="BldB" />
    </zoneRule>
    <zoneRule name="aysnc" exclusivePlacement="true">
      <zone name="BldA" />
      <zone name="BldB" />
    </zoneRule>
   </zoneMetadata>
  </mapSet>
 </objectgridDeployment>
</deploymentPolicy>
```

O primário e a réplica síncrona compartilham uma regra de zona primaySync com uma configuração de sinalizador exclusivo de false. Portanto, após o primário ou síncrono ser colocado em uma zona, então o outro também é colocado na mesma zona. A réplica assíncrona utiliza uma segunda regra de zona com as mesmas zonas que a regra de zona primarySync, mas ela utiliza o atributo **exclusivePlacement** configurado como true. Este atributo indica que um shard não pode ser colocado em uma zona com outro shard a partir da mesma partição. Como resultado, a réplica assíncrona não é colocada na mesma zona que o primário e as réplicas síncronas.

Colocar todos os primários em uma zona e todas réplicas em outra zona

Aqui, todos os primários estão em uma zona específica e todas as réplicas em uma zona diferente. Teremos um primário e uma réplica assíncrona única. Todas as réplicas estarão na zona A e os primários na B.

```
<?xml version="1.0" encoding="UTF-8"?>

<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation=
  "http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

 <objectgridDeployment objectgridName="library">
  <mapSet name="ms1" numberOfPartitions="13" minSyncReplicas="0"
   maxSyncReplicas="0" maxAsyncReplicas="1">
   <map ref="book" />
   <zoneMetadata>
    <shardMapping shard="P" zoneRuleRef="primaryRule"/>
    <shardMapping shard="A" zoneRuleRef="replicaRule"/>
    <zoneRule name ="primaryRule">
     <zone name="A" />
    </zoneRule>
    <zoneRule name="replicaRule">
     <zone name="B" />
      </zoneRule>
     </zoneMetadata>
    </mapSet>
  </objectgridDeployment>
</deploymentPolicy>
```

Aqui, é possível ver duas regras, uma para os primários (P) e outra para a réplica (A).

Zonas sobre Wide Area Networks (WAN)

É possível desejar implementar um eXtreme Scale único em vários prédios ou datacenters com interconexões de rede mais lentas. Conexões de rede mais lentas levam a uma largura da banda menor e mais conexões de latência. A possibilidade de partições de rede também aumenta neste modo devido ao congestionamento de rede e outros fatores. O eXtreme Scale aborda este ambiente severo das seguintes maneiras.

Pulsação limitada entre zonas

As Java Virtual Machines agrupadas em grupos principais realizam pulsação uma com a outra. Quando o serviço de catálogo organiza as Java Virtual Machines nos grupos, tais grupos não se estendem sobre zonas. Um líder neste grupo executa o push das informações de associação no serviço de catálogo. O serviço de catálogo verifica quaisquer falhas relatadas antes de tomar a ação. Ele faz isto ao tentar conectar-se às Java Virtual Machines suspeitas. Se o serviço de catálogo visualiza uma falsa detecção de falha, então, ele não executa nenhuma ação já que a partição do grupo principal irá resolver o problema em um curto período de tempo.

O serviço de catálogo também executar pulsação nos líderes do grupo principal periodicamente em uma taxa mas baixa para tratar o caso de isolamento do grupo principal.

Configurando a Detecção de Failover

É possível configurar a quantidade de tempo entre verificações do sistema para servidores com falha com a configuração do intervalo de pulsação.

Sobre Esta Tarefa

A configuração de failover varia dependendo do tipo de ambiente que você está usando. Se você estiver utilizando um ambiente independente, é possível configurar o failover com a linha de comandos. Se você estiver usando um

ambiente do WebSphere Application Server Network Deployment, é necessário configurar o failover no console administrativo do WebSphere Application Server Network Deployment.

Procedimento

- Configurar o failover para ambientes independentes.

  É possível configurar os intervalos de pulsação na linha de comandos usando o parâmetro **-heartbeat** no arquivo de script startOgServer.bat | startOgServer.sh. Configure esse parâmetro para um dos seguintes valores:

  Tabela 9. Intervalos de Pulsações

Valor	Ação	Descrição
0	Típica (padrão)	Failovers são tipicamente detectados em 30 segundos.
-1	Agressiva	Failovers são tipicamente detectados em 5 segundos.
1	Moderada	Failovers são tipicamente detectado em 180 segundos.

  Um intervalo de pulsação agressivo pode ser útil quando os processos e a rede estão estáveis. Se a rede ou os processos não são configurados de maneira ideal, as pulsações podem ser perdidas, o que pode resultar em uma falsa detecção de falhas.

- Configurar o failover para ambientes do WebSphere Application Server.

  O WebSphere Application Server Network Deployment Versão 6.0.2 e superior pode ser configurado para permitir que o WebSphere eXtreme Scale execute failover muito rapidamente. O tempo de failover padrão para falhas "hard" é de aproximadamente 200 segundos. Uma falha "hard" é um travamento de computador físico ou de servidor, uma desconexão de cabo de rede ou um erro do sistema operacional. As falhas causadas por travamentos de processo ou por falhas de software normalmente executam failover em menos de um segundo. A detecção de falha para falhas "soft" ocorre quando os soquetes de rede a partir de um processo inativo são fechados automaticamente pelo sistema operacional para o servidor que hospeda o processo.

  Configuração de pulsação do grupo principal

  O WebSphere eXtreme Scale que executa um processo do WebSphere Application Server herda as características de failover a partir das configurações de grupo principal do servidor de aplicativos. As seguintes seções descrevem como definir as configurações de pulsação do grupo principal para versões diferentes do WebSphere Application Server Network Deployment:

  – **Atualize as configurações do grupo principal para o WebSphere Application Server Network Deployment Versão 6.x e 7.x:**

    O intervalo de pulsação pode ser especificado em segundos nas versões do WebSphere Application Server da Versão 6.0 a Versão 6.1.0.12, ou em milissegundos iniciando na Versão 6.1.0.13. Também é necessário especificar o número de pulsações perdidas. Esse valor indica quantas pulsações podem estar ausentes antes que um Java Virtual Machine (JVM) equivalente seja considerado uma falha. O tempo de detecção de falha "hard" é, aproximadamente, o produto do intervalo de pulsações e o número de pulsações ausentes.

    Essas propriedades são especificadas usando as propriedades customizadas no grupo principal usando o console administrativo do WebSphere. Consulte Propriedades customizadas do grupo principal para obter detalhes da configuração. Estas propriedades devem ser especificadas para todos os grupos principais utilizados pelo aplicativo:

- O intervalo de pulsação é especificado usando a propriedade customizada IBM_CS_FD_PERIOD_SEC para segundos ou a propriedade customizada IBM_CS_FD_PERIOD_MILLIS para milissegundos (requer o V6.1.0.13 ou superior).
- O número de pulsações ausentes é especificado usando a propriedade customizada IBM_CS_FD_CONSECUTIVE_MISSED.

O valor padrão para a propriedade IBM_CS_FD_PERIOD_SEC é 20 e para a propriedade IBM_CS_FD_CONSECUTIVE_MISSED é 10. Se a propriedade IBM_CS_FD_PERIOD_MILLIS for especificada, ela substituirá qualquer conjunto de propriedades customizadas IBM_CS_FD_PERIOD_SEC. Os valores destas propriedades são valores de número inteiro positivo.

Use as seguintes configurações para alcançar um tempo de detecção de falha de 1500 milissegundos para os servidores WebSphere Application Server Network Deployment Versão 6.x:

- Configure `IBM_CS_FD_PERIOD_MILLIS = 750` (WebSphere Application Server Network Deployment V6.1.0.13 e superior)
- Configure `IBM_CS_FD_CONSECUTIVE_MISSED = 2`

– **Atualize as configurações do grupo principal para o WebSphere Application Server Network Deployment Versão 7.0:**

O WebSphere Application Server Network Deployment Versão 7.0 fornece duas configurações de grupo principal que podem ser ajustadas para aumentar ou diminuir a detecção de failover:

- **Período de transmissão de pulsação.** O padrão é 30000 milissegundos.
- **Período de tempo limite de pulsação.** O padrão é 180000 milissegundos.

Para obter mais detalhes sobre como alterar essas configurações, consulte o Centro de Informações do WebSphere Application Server Network Deployment: Configurações de Falha e Detecção de Descoberta.

Use as seguintes configurações para alcançar um tempo de detecção de falha de 1500 milissegundos para os servidores WebSphere Application Server Network Deployment Versão 7:

- Configure o período de transmissão de pulsação para 750 milissegundos.
- Configure o tempo limite da pulsação para 1500 milissegundos.

O que Fazer Depois

Quando estas configurações são modificadas para fornecer tempos de failover curtos, há alguns problemas de ajuste de sistema a considerar. Primeiro, Java não é um ambiente em tempo real. É possível que os encadeamentos sejam atrasados se a JVM estiver experimentando longos tempos de coleta de lixo. Os encadeamentos também podem ser atrasados se a máquina que hospeda o JVM estiver sobrecarregada (devido ao próprio JVM ou outros processos que são executados na máquina). Se os encadeamentos forem atrasados, as pulsações talvez não sejam enviadas a tempo. No pior dos casos, elas podem ser atrasadas pelo tempo necessário de failover. Se os encadeamentos forem atrasados, ocorrerão falsas detecções de falhas. O sistema deve ser ativado e dimensionado para garantir que falsas detecções de falhas não aconteçam na produção. O teste de carregamento adequado é a melhor maneira de garantir isto.

Nota: A versão atual do eXtreme Scale suporta o WebSphere Real Time.

Arquivo Descritor XML de Política de Implementação

Para configurar uma política de implementação, utilize um arquivo XML do descritor da política de implementação.

Nas seguintes seções, os elementos e atributos do arquivo XML do descritor da política de implementação está definido. Consulte o "Arquivo deploymentPolicy.xsd" na página 178 para obter o esquema XML da política de implementação correspondente.

Elementos no arquivo deploymentPolicy.xml

```
(1) <deploymentPolicy>
(2)  <objectgridDeployment objectGridName="blah">
(3)   <mapSet
(4)    name="mapSetName"
(5)    numberOfPartitions="numberOfPartitions"
(6)    minSyncReplicas="minimumNumber"
(7)     maxSyncReplicas="maximumNumber"
(8)    maxAsyncReplicas="maximumNumber"
(9)    replicaReadEnable="true|false"
(10)    numInitialContainers="numberOfInitialContainersBeforePlacement"
(11)    autoReplaceLostShards="true|false"
(12)    developmentMode="true|false"
(13)    placementStrategy="FIXED_PARTITION|PER_CONTAINER">
(14)    <map ref="backingMapReference" />
(15)   </mapSet>
(16)   <zoneMetadata>
(17)    <shardMapping
(18)      shard="shardName"
(19)      zoneRuleRef="zoneRuleRefName" />
(20)    <zoneRule
(21)      name="zoneRuleName"
(22)      exclusivePlacement="true|false" >
(23)      <zone name="ALPHA" />
(24)      <zone name="BETA" />
(25)      <zone name="GAMMA" />
(26)    </zoneRule>
(27)   </zoneMetadata>
(28)  </objectgridDeployment>
</deploymentPolicy>
```

Elemento deploymentPolicy (linha 1)

O elemento deploymentPolicy é o elemento de nível superior do arquivo XML de política de implementação. Este elemento configura o espaço de nomes do arquivo e o local do esquema. O esquema está definido no arquivo `deploymentPolicy.xsd`.

- **Número de Ocorrências:** Uma
- **Elemento filho:** objectgridDeployment

Elemento objectgridDeployment (linha 2)

O elemento objectgridDeployment é utilizado para referenciar uma instância do ObjectGrid a partir do arquivo XML do ObjectGrid. Dentro do elemento objectgridDeployment, é possível dividir seus mapas em conjuntos de mapas.

- **Número de ocorrências:** Uma ou mais
- **Elemento filho:** mapSet

Atributos

objectgridName
Especifica o nome da instância de ObjectGrid para implementar. Esse atributo faz referência a um elemento objectGrid que está definido no arquivo XML do ObjectGrid. (Necessário)

Por exemplo, o atributo objectgridName é configurado como `CompanyGrid` no arquivo companyGridDpReplication.xml. O atributo objectgridName faz referência ao `CompanyGrid` definido no arquivo companyGrid.xml. Leia sobre "Arquivo XML

descritor do ObjectGrid" na página 141 que você deve acoplar com o arquivo de política de implementação para cada instância de ObjectGrid.

Elemento mapSet (linha 3)

O elemento mapSet é utilizado para agrupar mapas. Os mapas dentro de um elemento mapSet são particionados e replicados da mesma maneira. Cada mapa deve pertencer a apenas um elemento mapSet.

- **Número de ocorrências:** Uma ou mais
- **Elemento filho:** map

Atributos

name
Especifica o nome do mapSet. Esse atributo deve ser exclusivo dentro do elemento objectgridDeployment. (Necessário)

numberOfPartitions
Especifica o número de partições para o elemento mapSet. O valor padrão é 1. O número deve ser apropriado para o número de contêineres que hospeda as partições. (Opcional)

minSyncReplicas
Especifica o número mínimo de réplicas assíncronas para cada partição no mapSet. O valor padrão é 0. Shards não são colocados até o domínio poder suportar o número mínimo de réplicas síncronas. Para suporte do valor minSyncReplicas, é preciso de mais um contêiner do que o valor de minSyncReplicas. Se o número de réplicas síncronas ficar abaixo do valor de minSyncReplicas, grave transações que não são mais permitidas àquela partição. (Opcional)

maxSyncReplicas
Especifica o número máximo de réplicas síncrona para cada partição no mapSet. O valor padrão é 0. Nenhuma outra réplica síncrona é colocada para uma partição após um domínio atingir esse número de réplicas síncronas para essa partição específica. A inclusão de contêineres que podem suportar esse ObjectGrid pode resultar em um aumento no número de réplicas síncronas se seu valor de maxSyncReplicas ainda não tiver sido atingido. (Opcional)

maxAsyncReplicas
Especifica o número máximo de réplicas assíncronas para cada partição no mapSet. O valor padrão é 0. Após a réplica primária e todas as réplicas síncronas serem colocadas para uma partição, as réplicas síncronas serão colocadas até o valor maxAsyncReplicas ser atendido. (Opcional)

replicaReadEnabled
Se este atributo é configurado como `true`, pedidos de leitura são distribuídos entre uma partição primária e suas réplicas. Se o atributo replicaReadEnabled for `false`, os pedidos de leitura são roteados para o primário apenas. O valor padrão é false. (Opcional)

numInitialContainers
Especifica o número de contêineres do eXtreme Scale que são necessários antes que ocorra o posicionamento inicial para os shards neste elemento mapSet. O valor padrão é 1. Esse atributo pode ajudar a economizar processo e largura de banda da rede ao tornar uma instância de ObjectGrid on-line. (Opcional)

Iniciar um contêiner do eXtreme Scale envia um evento para o serviço de catálogo. Na primeira vez em que o número de contêineres ativos for igual ao

valor numInitialContainers para um elemento mapSet, o serviço de catálogo colocará os shards a partir do mapSet, contanto que o minSyncReplicas também possa ser satisfeito. Depois que o valor numInitialContainers tiver sido atendido, cada evento iniciado por contêiner poderá acionar um reequilíbrio de shards não colocados e colocados anteriormente. Se souber aproximadamente quantos contêineres vão iniciar para esse elemento mapSet, será possível configurar o valor de numInitialContainers próximo desse número para evitar o reequilíbrio após cada início de contêiner. O posicionamento ocorre apenas quando você atinge o valor de numInitialContainers especificado no elemento mapSet.

autoReplaceLostShards
Especifica se os shards perdidos são colocados em outros contêineres. O valor padrão é verdadeiro. Quando um contêiner é parado ou falha, os shards executando no contêiner são perdidos. Um shard primário perdido faz com que um dos shards de réplica seja promovido para o shard primário para a partição correspondente. Devido a essa promoção, uma das réplicas é perdida. Se quiser que os shards perdidos permaneçam não colocados, configure o atributo autoReplaceLostShards como `false`. Essa configuração não afeta a cadeia de promoção, mas somente a substituição do último shard na cadeia. (Opcional)

developmentMode
Com este atributo, é possível influenciar onde um shard é colocado em relação a seus shards peer. O valor padrão é verdadeiro. Quando o atributo developmentMode estiver configurado como `false`, nenhum dos dois shards da mesma partição será colocado no mesmo computador. Quando o atributo developmentMode é configurado para `true`, os shards da mesma partição podem ser colocados na mesma máquina. Nos dois casos, nenhum dos dois shards da mesma partição são nem mesmo colocados no mesmo contêiner. (Opcional)

placementStrategy
Há duas estratégias de colocação. A estratégia padrão é `FIXED_PARTITION`, em que o número de shards primários que são colocados nos contêineres disponíveis é igual ao número de partições definidas, aumentado pelo número de réplicas. A estratégia alternativa é `PER_CONTAINER`, em que o número de shards primários que são colocados em cada contêiner é igual ao número de partições definidas, com um número igual de réplicas colocadas em outros contêineres. (Opcional)

Elemento de Mapa (Linha 14)

Cada mapa em um elemento mapSet faz referência a um dos elementos backingMap definido no arquivo XML de ObjectGrid correspondente. Cada mapa em um ambiente eXtreme Scale distribuído pode pertencer a apenas um elemento mapSet.

- **Número de ocorrências:** Uma ou mais
- **Elemento filho:** None

Atributos

ref
Fornece uma referência para um elemento backingMap no arquivo XML do ObjectGrid. Cada mapa em um elemento mapSet deve fazer referência a um elemento backingMap a partir do arquivo XML de ObjectGrid. O valor que é

designado ao atributo ref deve corresponder ao atributo name de um dos elementos backingMap no arquivo XML de ObjectGrid, como no fragmento de código a seguir. (Necessário)

Elemento zoneMetadata (Linha 16)

É possível colocar shards em zonas. Esta função permite maior controle sobre como o eXtreme Scale coloca shards em uma grade. As Java™ Virtual Machines que hospedam um servidor eXtreme Scale podem ser identificadas com um identificador de zona. O arquivo de implementação pode incluir uma ou mais regras de zonas e estas regras de zonas estão associadas a um tipo de shard. Para um plano de fundo adicional, consulte "Configurando Zonas para a Colocação de Réplica" na página 166.

- **Número de ocorrências:** Zero ou uma
- **Elementos filhos:**
  - shardMapping
  - zoneRule

Atributos: Nenhum

Elemento shardMapping (Linha 17)

Cada shard pode ser associado com uma regra de zona. Uma regra de zona pode ser compartilhada entre dois shards. Quando uma regra é compartilhada, então o sinalizador inclusivo ou exclusivo é estendido pelos shards de todos os tipos compartilhando uma regra única.

- **Número de ocorrências:** Zero ou uma
- **Elementos-filhos:** Nenhum

Atributos

shard
Especifique o nome de um shard ao qual associar zoneRule. (Necessário)

zoneRuleRef
Especifique o nome de um zoneRule ao qual associar o shard. (Opcional)

Elemento zoneRule (Linha 20)

Uma regra de zona especifica o possíveis conjuntos de zonas nos quais um shard pode ser colocado.

- **Número de ocorrências:** Uma ou mais
- **Elementos filhos:** zone

Atributos

name
Especifique o nome da regra de zona definida anteriormente, como zoneRuleRef em um elemento shardMapping. (Necessário)

exclusivePlacement
Uma configuração exclusiva indica que cada shard para uma partição é colocada em uma zona diferente na lista de zonas. Uma configuração inclusiva indica que após um shard ser colocado em uma zona da lista, todos os outros shards também são colocados nessa zona. Por exemplo, utilizar uma

configuração exclusiva significa que se houver três shards (um primário e duas réplicas síncronas), então, a lista de zonas deve ter três zonas. (Opcional)

Elementos da Zona (Linhas 23 a 25)

Suponha que você tenha três chassis blade e deseja que os shards primários sejam distribuídos para os três, com uma única réplica síncrona colocada em um chassi diferente do shard primário. É possível definir cada chassi como uma zona, com nomes de zonas correspondentes aos nomes de chassis: ALPHA, BETA e GAMMA.

- **Número de ocorrências:** Uma ou mais
- **Elementos-filhos:** Nenhum

Atributos

name
Especifique o nome de uma zona à qual deseja aplicar a regra de zona determinada. (Necessário)

Exemplo:

No exemplo a seguir, o elemento mapSet é utilizado para configurar uma política de implementação. O valor é configurado como `mapSet1` e é dividido em 10 partições. Cada uma dessas partições tem pelo menos uma réplica síncrona disponíveis e não mais do que duas réplicas síncronas. Cada partição também tem uma réplica assíncrona se o ambiente a suportar. Todas as réplicas síncronas são colocadas antes que quaisquer réplicas assíncronas sejam colocadas. Além disso, o serviço de catálogo não tenta colocar os shards para o elemento mapSet1 até o domínio poder suportar o valor minSyncReplicas. O suporte do valor minSyncReplicas requer dois ou mais contêineres: um para o primário e dois para a réplica síncrona.

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org./2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

 <objectgridDeployment objectgridName="CompanyGrid">
  <mapSet name="mapSet1" numberOfPartitions="10"
   minSyncReplicas="1" maxSyncReplicas="2" maxAsyncReplicas="1"
    numInitialContainers="10" autoReplaceLostShards="true"
   developmentMode="false" replicaReadEnabled="true">
   <map ref="Customer" />
   <map ref="Item" />
   <map ref="OrderLine" />
   <map ref="Order" />
  </mapSet>
 </objectgridDeployment>

</deploymentPolicy>
```

Embora apenas dois contêineres sejam necessários para satisfazer as configurações de replicação, o atributo numInitialContainers requer 10 contêineres disponíveis antes de o serviço de catálogo tentar colocar qualquer um dos shards nesse elemento mapSet. Após o domínio ter 10 contêineres capazes de suportar o CompanyGrid ObjectGrid, todos os shards no elemento mapSet1 serão colocados.

Como o atributo autoReplaceLostShards está configurado como true, qualquer shard nesse elemento mapSet que é perdido como resultado da falha do contêiner é automaticamente substituído em outro contêiner, contanto que esse contêiner esteja disponível para hospedar o shard perdido. Os shards da mesma partição não podem ser colocados na mesma máquina para o elemento mapSet1, pois o atributo developmentMode está configurado como `false`. Pedidos somente de leitura são distribuídos no shard primário e em suas réplicas para cada partição, pois o valor de replicaReadEnabled é `true`.

O arquivo companyGridDpMapSetAttr.xml utiliza o atributo ref no mapa para fazer referência a cada um dos elementos backingMap do arquivo companyGrid.xml.

Arquivo deploymentPolicy.xsd

Utilize o esquema XML de política de implementação para criar um arquivo XML descritor de implementação.

Consulte o "Arquivo Descritor XML de Política de Implementação" na página 173 para obter descrições dos elementos e atributos definidos no arquivo deploymentPolicy.xsd.

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns:dp="http://www.ibm.com/ws/objectgrid/deploymentPolicy"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 targetNameSpace="http://ibm.com/ws/objectgrid/deploymentPolicy"
 elementFormDefault="qualified">

 <xsd:element name="deploymentPolicy">
  <xsd:complexType>
   <xsd:choice>
    <xsd:element name="objectgridDeployment"
     type="dp:objectgridDeployment" minOccurs="1"
     maxOccurs="unbounded">
     <xsd:unique name="mapSetNameUnique">
      <xsd:selector xpath="dp:mapset" />
      <xsd:field xpath="@name" />
     </xsd:unique>
    </xsd:element>
   </xsd:choice>
  </xsd:complexType>
 </xsd:element>

 <xsd:complexType name="objectgridDeployment">
  <xsd:sequence>
   <xsd:element name="mapSet" type="dp:mapSet"
    maxOccurs="unbounded" minOccurs="1">
    <xsd:unique name="mapNameUnique">
     <xsd:selector xpath="dp:map" />
     <xsd:field xpath="@ref" />
    </xsd:unique>
   </xsd:element>
  </xsd:sequence>
  <xsd:attribute name="objectgridName" type="xsd:string"
   use="required" />
 </xsd:complexType>

 <xsd:complexType name="mapSet">
  <xsd:sequence>
   <xsd:element name="map" type="dp:map" maxOccurs="unbounded"
    minOccurs="1" />
   <xsd:element name="zoneMetadata" type="dp:zoneMetadata"
    maxOccurs="1" minOccurs="0">

    <xsd: key name="zoneRuleName">
     <xsd:selector xpath="dp:zoneRule" />
     <xsd:field xpath="@name" />
    </xsd:key>

    <xsd:keyref name="zoneRuleRef"
     refer="dp:zoneRuleName">
     <xsd:selector xpath="dp:shardMapping" />
     <xsd:field xpath="@zoneRuleRef" />
    </xsd:keyref>

   </xsd:element>
  </xsd:sequence>
  <xsd:attribute name=="name" type="xsd:string" use="required" />
  <xsd:attribute name="numberOfPartitions" type="xsd:int"
   use="optional" />
  <xsd:attribute name="minSyncReplicas" type="xsd:int"
   use="optional" />
  <xsd:attribute name="maxSyncReplicas" type="xsd:int"
   use="optional" />
  <xsd:attribute name="maxAsyncReplicas" type="xsd:int"
   use="optional" />
  <xsd:attribute name="replicaReadEnabled" type="xsd:boolean"
   use="optional" />
  <xsd:attribute name="numInitialContainers" type="xsd:int"
   use="optional" />
  <xsd:attribute name="autoReplaceLostShards" type="xsd:boolean"
   use="optional" />
  <xsd:attribute name="developmentMode" type="xsd:boolean"
   use="optional" />
```

```
 <xsd:attribute name="placementStrategy"
  type="dp:placementStrategy" use="optional" />
</xsd:complexType>

<xsd:simpleType name="placementStrategy">
 <xsd:restriction base="xsd:string">
  <xsd:enumeration value="FIXED_PARTITIONS" />
  <xsd:enumeration value="PER_CONTAINER" />
 </xsd:restriction>
</xsd:simpleType>


<xsd:complexType name="map">
 <xsd:attribute name="ref" use="required" />
</xsd:complexType>

<xsd:complexType name="zoneMetadata">
 <xsd:sequence>
  <xsd:element name="shardMapping" type="dp:shardMapping"
   maxOccurs="unbounded" minOccurs="1" />
  <xsd:element name="zoneRule" type="dp:zoneRule"
   maxOccurs="unbounded" minOccurs="1">

  </xsd:element>

 </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="shardMapping">
 <xsd:attribute name="shard" use="required">
  <xsd:simpleType>
   <xsd:restriction base="xsd:string">
    <xsd:enumeration value="P"></xsd:enumeration>
    <xsd:enumeration value="S"></xsd:enumeration>
    <xsd:enumeration value="A"></xsd:enumeration>
   </xsd:restriction>
  </xsd:simpleType>
 </xsd:attribute>
 <xsd:attribute name="zoneRuleRef" type="xsd:string"
  use="required" />
</xsd:complexType>

<xsd:complexType name="zoneRule">
 <xsd:sequence>
  <xsd:element name="zone" type="dp:zone"
   maxOccurs="unbounded" minOccurs="1" />
 </xsd:sequence>
 <xsd:attribute name="name" type="xsd:string" use="required" />
 <xsd:attribute name="exclusivePlacement" type="xsd:boolean" />
  use="optional" />
</xsd:complexType>

<xsd:complexType name="zone">
 <xsd:attribute name="name" type="xsd:string" use="required" />
<xsd:complextType>

</xsd:schema>
```

Configurando Servidores de Catálogo e Contêiner

Use um arquivo de propriedades para configurar servidores de catálogos e servidores de contêiner. O WebSphere eXtreme Scale possui dois tipos de servidores: servidores de catálogos e servidores de contêiner. Os servidores de catálogo controlam o posicionamento de shards e descobrem e monitoram os servidores de contêiner. Vários servidores de catálogos juntos compõem o serviço de catálogo. Os servidores de contêiner são as Java™ Virtual Machines que armazenam os dados do aplicativo para a grade.

Configurando Topologias de Replicação Multimestre

Com o recurso de replicação assíncrona multimestre, você usa links para interconectar um conjunto de domínios, então eXtreme Scale sincroniza os domínios, usando a replicação sobre os links. Como define os links entre domínios, é possível construir quase qualquer topologia. Defina links nos arquivos de propriedades dos servidores de catálogo para cada domínio ou defina links no tempo de execução usando programas JMX ou o utilitário de linha de comandos

xsadmin. No entanto, você cria links, o conjunto de links atuais para um domínio é armazenado no serviço de catálogo, permitindo que você inclua e remova links sem reiniciar o domínio (grade).

Antes de Iniciar

Topologias de replicação de grade multimestre (AP) apresenta a você as características de várias topologias de replicação multimestre. O procedimento a seguir descreve a mecânica de configurar vários links entre domínios para formar uma topologia – qualquer topologia. Após o procedimento, são fornecidos alguns exemplos para ilustrar como configurar topologias específicas, como uma formação hub e spoke.

Procedimento

1. Defina links no arquivo de propriedades para o servidor de catálogos de cada domínio na topologia, para fins de autoinicialização.

   O arquivo de propriedades será detectado automaticamente se você nomeá-lo `objectGridServer.properties` (faz distinção de maiúsculas e minúsculas em alguns sistemas) e colocá-lo no caminho de classe usado ao iniciar uma instância do serviço de catálogo. Também é possível especificar seu local na linha de comandos para o script `startOgServer.bat|sh`, usando o parâmetro *-serverProps*.

   Como os nomes de propriedades fazem distinção de maiúsculas e minúsculas, tenha cuidado com a capitalização ao atualizar o arquivo de propriedades.

   Nome de Domínio Local
   Especifique o nome "deste" domínio, como por exemplo, domínio A:

   Por exemplo:

   ```
   domainName=A
   ```

   Uma lista opcional de nomes de domínio externo

   Especifique os nomes de "outros" domínios na topologia de replicação multimestre, como por exemplo, domínio B:

   ```
   foreignDomains=B
   ```

   Uma lista opcional de terminais para os nomes de domínio externo
   Especifica as informações de conexão para os serviços de catálogo dos domínios externos, como por exemplo, domínio B:

   Por exemplo:

   ```
   B.endPoints=hostB1:2809, hostB2:2809
   ```

   Se um domínio externo tiver vários serviços de catálogo, especifique todos eles.
2. Use o utilitário do comando xsadmin ou a programação JMX para incluir ou remover links no tempo de execução.

   Os links para um domínio são mantidos no serviço de catálogo na memória replicada. Este conjunto de links pode ser alterado a qualquer momento pelo administrador sem exigir o reinício deste domínio ou de qualquer outro domínio. O utilitário de linha de comandos xsadmin inclui várias opções para trabalhar com links.

   O utilitário xsadmin conecta a um serviço de catálogo e, assim, um único domínio. Portanto, xsadmin pode ser usado para criar e destruir links entre o domínio ao qual se conecta e qualquer outro domínio.

   Use a linha de comandos para criar um novo link, por exemplo:

```
xsadmin –ch host –p 1099 –establishLink dname fdHostA:2809,fdHostB:2809
```

O comando estabelece um novo link entre o domínio 'local' e o domínio externo denominado "dname", cujo serviço de catálogo está em execução em fdHostA:2809 e fdHostB:2809. O domínio local tem uma JVM de serviço de catálogo com um endereço JMX de host:1099. Especifique todos os terminais do catálogo do domínio externo para que a conectividade de tolerância a falhas com o domínio seja possível. Não é recomendado usar um único par host:port para o serviço de catálogo do domínio externo.

Não importa qual JVM do serviço de catálogo local o xsadmin especifica usando –ch e –p. Qualquer JVM de catálogo funcionará. Se o catálogo for hospedado em um gerenciador de implementação do WebSphere Application Server, a porta será, geralmente, 9809.

As portas especificadas para o domínio externo NÃO são portas JMX. Elas são as portas usuais que você usaria para clientes do eXtreme Scale.

Depois de o comando para incluir um novo link ser emitido, o serviço de catálogo instrui todos os contêineres sob seu gerenciamento a começar a replicar para o domínio externo. Um link não é necessário em ambos os lados. Ele é necessário apenas para criar um link em um lado.

Use a linha de comandos para remover um link, por exemplo:

```
xsadmin –ch host –p 1099 –dismissLink dname
```

O comando conecta ao serviço de catálogo para um domínio e o instrui a parar de replicar para um domínio específico. Um link precisa ser descartado apenas de um lado.

Exemplo

Suponha que você queira definir uma configuração com dois domínios envolvendo os Domínios A e B.

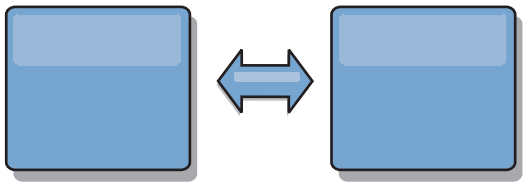


Figura 8. Link entre Domínios

Aqui é apresentado o arquivo de propriedades para o servidor de catálogos no domínio A:

```
domainName=A
foreignDomains=B
B.endPoints=hostB1:2809, hostB2:2809
```

Aqui é apresentado o arquivo de propriedades para o servidor de catálogos no domínio B. Observe a semelhança entre os dois arquivos de propriedades.

```
domainName=B
foreignDomains=A
B.endPoints=hostB1:2809, hostB2:2809
```

Depois que os dois domínios forem iniciados, quaisquer grades com as seguintes características serão replicadas entre os domínios.

- Ter um serviço de catálogo privado com um nome de domínio exclusivo

- Ter o mesmo nome de grade que outras grades no domínio
- Ter o mesmo número de partições que outras grades no domínio
- Ser uma grade FIXED_PARTITION (grades PER_CONTAINER não podem ser replicadas)
- Ter o mesmo número de partições (pode o não ter o mesmo número e tipos de réplicas)
- Ter os mesmos tipos de dados sendo replicados como outras grades no domínio
- Ter o mesmo nome do conjunto de mapas, nomes de mapas e modelos de mapas dinâmicos que outras grades no domínio

Observe que a política de replicação de um domínio será ignorada.

O exemplo anterior mostra como configurar cada domínio para ter um link para o outro domínio, mas é necessário apenas definir um link em uma direção. Este fato é especialmente útil em topologias hub e spoke, permitindo uma configuração muito mais simples. O arquivo de propriedades do hub não requer atualizações conforme os spokes são incluídos e cada arquivo do spoke precisa apenas incluir informações do hub. De forma semelhante, uma topologia em anel requer que cada domínio tenha apenas um link para o domínio anterior e o próximo no anel.

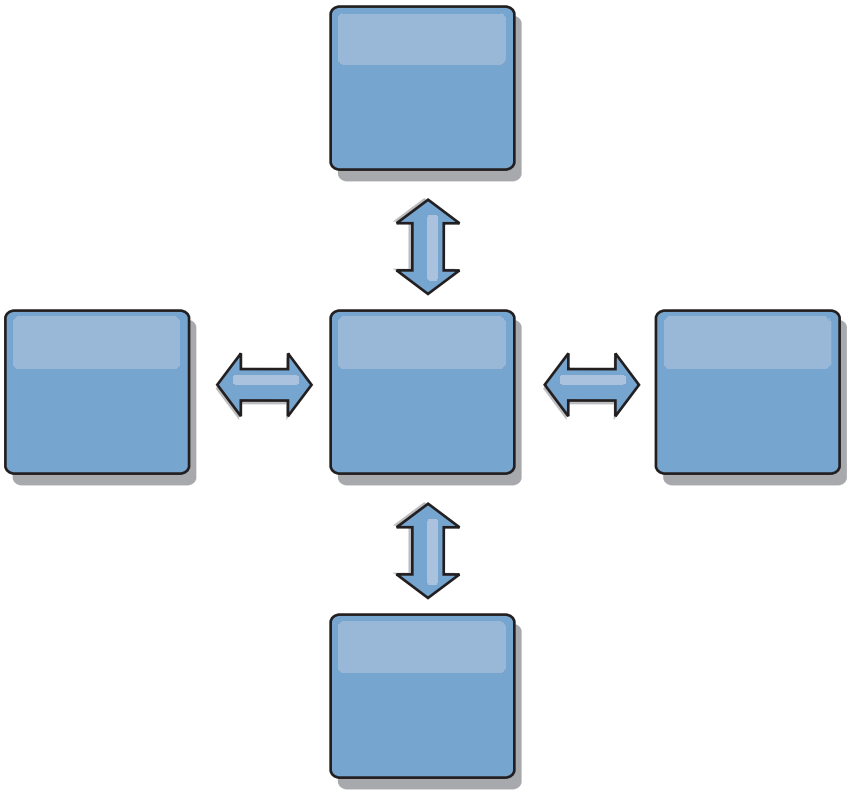


Figura 9. Topologia Hub e Spoke

O hub e quatro spokes (domínios A, B, C e D) podem ter arquivos de propriedades do servidor de catálogos como os seguintes exemplos.

```
domainName=Hub
```

O primeiro spoke pode ter as seguintes propriedades:

```
domainName=A
foreignDomains=Hub
Hub.endPoints=hostH1:2809, hostH2:2809
```

O segundo spoke pode ter as seguintes propriedades:

```
domainName=B
foreignDomains=Hub
Hub.endPoints=hostH1:2809, hostH2:2809
```

O terceiro spoke pode ter as seguintes propriedades:

```
domainName=C
foreignDomains=Hub
Hub.endPoints=hostH1:2809, hostH2:2809
```

O quarto spoke pode ter as seguintes propriedades:

```
domainName=D
foreignDomains=Hub
Hub.endPoints=hostH1:2809, hostH2:2809
```

Arquivo de Propriedades do Servidor

O arquivo de propriedades do servidor contém várias propriedades que definem configurações diferentes para o servidor, como configurações de rastreio, criação de log e configuração de segurança. O arquivo de propriedades do servidor é usado pelo serviço de catálogo e pelos servidores de contêiner.

Amostra do Arquivo de Propriedades do Servidor

É possível usar o arquivo `sampleServer.properties` que esteja no diretório *extremescale_root*`/properties` para criar seu arquivo de propriedades.

Especificando um Arquivo de Propriedades do Servidor

É possível especificar o arquivo de propriedades do servidor de uma das seguintes formas. Especificar uma configuração ao usar um dos itens recentes na lista substitui a configuração anterior. Por exemplo, se você especificar um valor de propriedade de sistema para o arquivo de propriedades do servidor, as propriedades nesse arquivo substituirão os valores no arquivo `objectGridServer.properties` que estiver no caminho de classe.

1. Como um arquivo nomeado corretamente no caminho de classe. Se você colocar esse arquivo nomeado corretamente no diretório atual, o arquivo será localizado apenas se o diretório atual estiver no caminho de classe. O nome usado é o seguinte:

   ```
   objectGridServer.properties
   ```

2. Como uma propriedade do sistema em uma configuração do WebSphere Application Server ou independente que especifica um arquivo no diretório atual do sistema. O arquivo não pode estar no caminho de classe:

   ```
   -Dobjectgrid.server.props=file_name
   ```

3. Como um parâmetro ao executar o comando startOgServer. É possível substituir essas propriedades manualmente para especificar um arquivo no diretório atual do sistema:

   ```
   -serverProps file_name
   ```

4. Como uma substituição programática usando os métodos ServerFactory.getCatalogServerProperties e ServerFactory.getCatalogServerProperties. Os dados no objeto são preenchidos com os dados a partir dos arquivos de propriedades.

Propriedades do Servidor

Propriedades Gerais

workingDirectory
Especifica o local para onde a saída do servidor de contêiner é gravada. Quando esse valor não é especificado, a saída será gravada em um diretório `log` dentro do diretório atual. Essa propriedade se aplica ao servidor de contêiner e ao serviço de catálogo.

Padrão: Nenhum valor

minThreads
Especifica o número mínimo de encadeamentos utilizados pelo conjunto de encadeamentos interno no tempo de execução para evictores integrados e operações de DataGrid.

Padrão: 10

maxThreads
Especifica o número máximo de encadeamentos utilizados pelo conjunto de encadeamentos interno no tempo de execução para evictores integrados e operações de DataGrid.

Padrão: 50

traceSpec
Ativa o rastreio e a cadeia de especificação de rastreio para o servidor de contêiner. O rastreio é desativado por padrão. Essa propriedade se aplica ao servidor de contêiner e ao serviço de catálogo.

Padrão: `*=all=disabled`

traceFile
Especifica um nome de arquivo para gravar informações de rastreio. Essa propriedade se aplica ao servidor de contêiner e ao serviço de catálogo.

systemStreamToFileEnabled
Permite que o contêiner grave SystemOut, SystemErr e a saída de rastreio em um arquivo. Se a propriedade for configurada para `false`, a saída não será gravada em um arquivo e será gravada no console.

Padrão: `true`

enableMBeans
Ativa os beans gerenciados (MBeans) do contêiner ObjectGrid. Essa propriedade se aplica ao servidor de contêiner e ao serviço de catálogo.

Padrão: `true`

serverName
Configura o nome do servidor que é usado para identificar o servidor. Essa propriedade se aplica ao servidor de contêiner e ao serviço de catálogo.

zoneName
Configura o nome da zona à qual o servidor pertence. Essa propriedade se aplica ao servidor de contêiner e ao serviço de catálogo.

haManagerPort
Sinônimo de porta de peer. Especifica o número da porta usada pelo gerenciador de alta disponibilidade. Se essa propriedade não for configurada, o serviço de catálogo gera uma porta disponível automaticamente. Essa propriedade se aplica ao servidor de contêiner e ao serviço de catálogo.

listenerHost

Especifica o nome do host para o qual o Object Request Broker (ORB) deve conectar-se. Essa propriedade se aplica ao servidor de contêiner e ao serviço de catálogo.

Se sua configuração envolver várias placas de rede, configure o host e a porta do listener para que o Object Request Broker na JVM saiba o endereço IP ao qual se ligar. Para servidores de catálogo e de contêiner, especifique o host e a porta do listener no arquivo de propriedades do servidor. A negligência ao especificar qual endereço IP usar produz sintomas como tempos de conexão esgotados, falhas de API incomuns e clientes que parecem interrompidos.

listenerPort

Especifica o número da porta para a qual o Object Request Broker (ORB) deve conectar-se. Essa propriedade se aplica ao servidor de contêiner e ao serviço de catálogo.

JMXServicePort

Especifica o número da porta na qual o servidor MBean deve atender. Essa propriedade se aplica ao servidor de contêiner e ao serviço de catálogo.

Propriedades do Servidor de Contêiner

statsSpec

Define a especificação de estatísticas para o servidor de contêiner.

Exemplo:

```
all=disabled
```

memoryThresholdPercentage

Configura o limite de memória para despejo baseado em memória. A porcentagem especifica o heap máximo que deve ser usado no Java Virtual Machine (JVM) antes de a liberação ocorrer. O valor-padrão é `-1`, que indica que o limite de memória não está configurado. Se a propriedade memoryThresholdPercentage não for configurada, o valor MemoryPoolMXBean será configurado com o valor fornecido. Consulte Interface MemoryPoolMXBean na especificação de API Java para obter mais informações. Porém, a liberação ocorre apenas se ela for ativada em um evictor. Para ativar a liberação baseada em memória, consulte as informações sobre os evictors no *Visão Geral do Produto*. Essa propriedade é aplicada apenas a um servidor de contêiner.

catalogServiceEndPoints

Especifica os pontos de extremidade para conectar ao domínio do serviço de catálogo. Este valor deve estar no formato `host:port<,host:port>` em que o valor do host é o valor listenerHost e o valor da porta é o valor listenerPort do servidor de catálogo. Essa propriedade é aplicada apenas a um servidor de contêiner.

Propriedades do Serviço de Catálogo

domainName

Especifica o nome de domínio usado para identificar exclusivamente este domínio do serviço de catálogo para clientes ao rotear para vários domínios. Essa propriedade é aplicada apenas no serviço de catálogo.

enableQuorum

Ativa o quorum para o serviço de catálogo. O quorum é usado para garantir que uma maioria do domínio do serviço de catálogo esteja disponível antes de permitir a modificação para o posicionamento de

partições nos servidores de contêiner disponíveis. Para ativar o quorum, configure o valor para `true` ou `ativado`. O valor padrão é `disabled`. Essa propriedade é aplicada apenas no serviço de catálogo.

catalogClusterEndpoints
Especifica os pontos de extremidade do domínio do serviço de catálogo para o serviço de catálogo. Esta propriedade especifica os pontos de extremidade do serviço de catálogo para iniciar o domínio do serviço de catálogo. Use o seguinte formato:

```
serverName:hostName:clientPort:peerPort<serverName:hostName:clientPort:peerPort>
```

Essa propriedade é aplicada apenas no serviço de catálogo.

heartBeatFrequencyLevel
Especifica com que frequência a pulsação ocorre. O nível de frequência de pulsação é uma troca entre o uso dos recursos e a hora da descoberta de falha. Quanto maior for a frequência das pulsações, mais recursos serão usados e as falhas serão descobertas mais rapidamente. Essa propriedade é aplicada apenas no serviço de catálogo. Utilize um dos seguintes valores:

- `0`: Especifica o nível de pulsação a uma taxa normal. Com esse valor, a detecção de failover ocorrerá a uma taxa razoável sem usar excessivamente os recursos. (Default)
- `-1`: Especifica um nível de pulsação forte. Com esse valor, as falhas serão detectadas mais rapidamente, mas também são usados um processador e recursos de rede adicionais. Esse nível pode ter pulsações ausentes quando o servidor estiver ocupado.
- `1`: Especifica um nível de pulsação livre. Com esse valor, uma frequência de pulsação diminuída aumenta o tempo para detectar falhas, mas também diminui o uso do processador e de rede.

Propriedades do Servidor de Segurança

O arquivo de propriedades do servidor também é usado para configurar a segurança do servidor eXtreme Scale. Use um arquivo de propriedades único do servidor para especificar as propriedades básicas e de segurança.

Propriedades gerais de segurança

securityEnabled
Ativa a segurança do servidor de contêiner ao configurar para `true`. O valor padrão é `false`. Esta propriedade deve corresponder à propriedade securityEnabled que é especificada no arquivo `objectGridSecurity.xml` que é fornecido para o servidor de catálogos.

credentialAuthentication
Indica se este servidor suporta a autenticação de credencial. Escolha um dos seguintes valores:

- `Nunca`: O servidor não suporta a autenticação de credencial.
- `Suportado`: O servidor suporta a autenticação de credencial se o cliente também suportar a autenticação de credencial.
- `Necessário`: O cliente requer autenticação de credencial.

Consulte o “Autenticação de Cliente do Aplicativo” na página 358 para obter detalhes sobre a autenticação de credencial.

Configurações de segurança da camada de transporte

transportType
Especifica o tipo de transporte do servidor. Utilize um dos seguintes valores:

- `TCP/IP`: Indica que o servidor suporta apenas conexões TCP/IP.
- `SSL-Suportado`: Indica que o servidor suporta ambas as conexões TCP/IP e Secure Sockets Layer (SSL). (Default)
- `SSL-Necessário`: Indica que o servidor requer as conexões SSL.

Propriedades de Configuração SSL

alias Especifica o nome do alias no armazenamento de chaves. Esta propriedade será utilizada se o armazenamento de chaves tiver vários certificados de pares de chaves e você desejar selecionar um dos certificados.

Padrão: nenhum valor

contextProvider
Especifica o nome do provedor de contexto para o serviço de confiança. Se você indicar um valor inválido, ocorrerá uma exceção de segurança que indica que o tipo de provedor de contexto está incorreto.

Valores válidos: `IBMJSSE2`, `IBMJSSE`, `IBMJSSEFIPS`, etc.

protocol
Indica o tipo de protocolo de segurança a ser usado para o cliente. Configure esse valor de protocolo baseado no provedor Java Secure Socket Extension (JSSE) que será usado. Se você indicar um valor inválido, ocorrerá uma exceção de segurança que indica que o valor de protocolo está incorreto.

Valores válidos: `SSL`, `SSLv2`, `SSLv3`, `TLS`, `TLSv1`, etc.

keyStoreType
Indica o tipo de armazenamento de chaves. Se você indicar um valor inválido, ocorrerá uma exceção de segurança no tempo de execução.

Valores válidos: `JKS`, `JCEK`, `PKCS12`, etc.

trustStoreType
Indica o tipo de armazenamento de confiança. Se você indicar um valor inválido, ocorrerá uma exceção de segurança no tempo de execução.

Valores válidos: `JKS`, `JCEK`, `PKCS12`, etc.

keyStore
Especifica um caminho completo para o arquivo de armazenamento de chaves.

Exemplo:

```
etc/test/security/client.private
```

trustStore
Especifica um caminho completo para o arquivo de armazenamento de confiança.

Exemplo:

```
etc/test/security/server.public
```

keyStorePassword
Especifica a senha da cadeia para o armazenamento de chaves. É possível codificar este valor ou usar o valor real.

trustStorePassword
Especifica uma senha de cadeia para o armazenamento de confiança. É possível codificar este valor ou usar o valor real.

clientAuthentication
Se a propriedade for configurada para true, o cliente SSL deverá ser autenticado. A autenticação do cliente SSL é diferente da autenticação do certificado cliente. A autenticação do certificado cliente significa autenticar um cliente para um registro de usuário de acordo com a cadeia de certificados. Essa propriedade garante que o servidor se conecte ao cliente correto.

Configuração SecureTokenManager

A configuração SecureTokenManager é usada para proteger a cadeia secreta para autenticações mútuas do servidor e para proteger o token de conexão única."Segurança de Grade" na página 356

secureTokenManagerType
Especifica o tipo de configuração SecureTokenManager. É possível usar uma das seguintes configurações:

- `nenhum`: Indica que nenhum gerenciador de token seguro é usado.
- `padrão`: Indica que o gerenciador de token que é fornecido com o produto WebSphere eXtreme Scale é usado. É necessário fornecer uma configuração de armazenamento de chaves SecureToken.
- `custom`: Indica que você tem seu próprio gerenciador de token que você especificou com a classe de implementação do SecureTokenManager.

customTokenManagerClass
Especifica o nome da sua classe de implementação SecureTokenManager, se você tiver especificado o valor de propriedade SecureTokenManagerType como `customizado`. A classe de implementação deve ter um construtor padrão a ser instanciado.

customSecureTokenManagerProps
Especifica as propriedades de classe de implementação SecureTokenManager customizada. Essa propriedade é usada apenas se o valor secureTokenManagerType for `customizado`. O valor é configurado para o objeto SecureTokenManager com o método setProperties(String).

Configuração de armazenamento de chaves de token seguro

secureTokenKeyStore
Especifica o nome do caminho de arquivo para o armazenamento de chaves que armazena o par de chaves público-privado e a chave secreta.

secureTokenKeyStoreType
Especifica o tipo de armazenamento de chaves, por exemplo, JCKES. É possível configurar esse valor baseado no provedor Java Secure Socket Extension (JSSE) usado. No entanto, esse armazenamento de chaves deve suportar chaves secretas.

secureTokenKeyPairAlias
Especifica o alias do par de chaves público-privado que é usado para assinatura e verificação.

secureTokenKeyPairPassword
Especifica a senha para proteger o alias de par de chaves que é usado para assinatura e verificação.

secureTokenSecretKeyAlias
Especifica o alias de chave secreta que é usado para codificação.

secureTokenSecretKeyPassword
Especifica a senha para proteger a chave secreta.

secureTokenCipherAlgorithm
Especifica o algoritmo que é usado para fornecer uma codificação. É possível configurar esse valor baseado no provedor Java Secure Socket Extension (JSSE) usado.

secureTokenSignAlgorithm
Especifica o algoritmo que é usado para assinar o objeto. É possível configurar esse valor com base no provedor JSSE utilizado.

Cadeia de autenticação

authenticationSecret
Especifica a cadeia secreta para fazer o pedido ao servidor. Quando um servidor for inicializado, ele deverá apresentar essa cadeia ao servidor principal ou ao servidor de catálogos. Se a cadeia secreta corresponder com o que está no servidor principal, esse servidor poderá fazer a união.

Configurando Portas

O WebSphere eXtreme Scale é um cache distribuído que requer portas de abertura para se comunicar com o Object Request Broker (ORB) e a fila do Protocolo de Controle de Transmissões (TCP) entre Java™ Virtual Machines e outras máquinas.

Planejamento para Portas de Rede

O WebSphere eXtreme Scale é um cache distribuído que necessita de portas de abertura para se comunicar com o Object Request Broker (ORB) e a pilha do Protocolo de Controle de Transmissões (TCP) entre Java Virtual Machines e outras máquinas. Você deve planejar e controlar suas portas, principalmente em um ambiente de firewall, como quando você está utilizando um serviço de catálogo e contêineres em várias portas.

Domínio do Serviço de Catálogo

Um domínio do serviço de catálogo requer as seguintes portas para ser definido:

peerPort
Especifica a porta para o gerenciador de alta disponibilidade (HA) se comunicar entre servidores de catálogo de peer sobre uma pilha TCP.

clientPort
Especifica a porta para servidores de catálogo acessarem dados do serviço de catálogo.

JMXServicePort
Especifica qual porta o serviço Java Management Extensions (JMX) deve usar.

listenerPort
Define a porta listener ORB para contêineres e clientes se comunicarem com o serviço de catálogo por meio de ORB.

A forma como você define essas portas depende de se você está usando o modo independente ou se está iniciando os servidores de catálogo do eXtreme Scale em um ambiente do WebSphere Application Server:

- **Para o modo independente:**

Use o comando startOgServer para especificar as portas anteriormente listadas com a opção no modo independente, conforme mostrado no seguinte exemplo:

```
–catalogServiceEndPoints cs1:host1:clientPort:peerPort, cs2:host2:clientPort:peerPort,
cs3:host3:clientPort:peerPort -listenerPort <orbPort> -JMXServicePort <jmxPort>
```

Consulte “Iniciando o Serviço de Catálogo em um Ambiente Independente” na página 329 para obter mais informações sobre como iniciar o serviço de catálogo no modo independente.

- **Para um ambiente do WebSphere Application Server:**

  É possível definir um domínio do serviço de catálogo no console administrativo. Consulte “Criando Domínios do Serviço de Catálogo no WebSphere Application Server” na página 344 para obter mais informações.

Servidores de Contêineres

Os servidores de contêineres do WebSphere eXtreme Scale também precisam de várias portas para operar. Por padrão, o servidor de contêineres do eXtreme Scale gera sus porta do gerenciador HA e porta listener ORB automaticamente com portas dinâmicas. No ambiente de firewall, como é vantajoso planejar e controlar portas, as opções são fornecidas para iniciar os servidores de contêineres do eXtreme Scale com porta HAManager e porta listener ORB especificadas com uma opção no comando startOgServer, como mostra o exemplo a seguir:

```
-HaManagerPort <peerPort>
-listenerPort <orbPort>
```

O planejamento adequado do controle da porta é um benefício, mas existe uma dificuldade inerente no planejamento e gerenciamento dessas portas quando centenas de Java Virtual Machines são iniciadas em uma máquina. Qualquer conflito de porta causará uma falha da inicialização do servidor.

Quando a segurança está ativada, uma porta Secure Socket Layer (SSL) é uma inclusão necessária às portas listadas anteriormente. Usar `Dcom.ibm.CSI.SSLPort=<sslPort>` como um argumento **-jvmArgs** configura a porta SSL para `<sslPort>`. Leia mais sobre configurações de segurança para eXtreme Scale para ajudá-lo com o planejamento de portas.

Configurando Portas no Modo Independente

Um Java Virtual Machine que hospeda uma instância de serviço de catálogo requer quatro portas. Duas portas são usadas para uso interno, a terceira porta é usada para clientes e shards de contêiner para se comunicar com o Internet Inter-ORB Protocol (IIOP) e a quarta porta é usada para comunicação com oJava Management Extensions (JMX).

Sobre Esta Tarefa

Terminais do Serviço de Catálogo do Java Virtual Machine (JVM)

O eXtreme Scale usa o IIOP principalmente para se comunicar com o Java Virtual Machines. O serviço de catálogo do Java Virtual Machines é o único Java Virtual Machines que requer configuração explícita das portas para os serviços IIOP e das portas de serviços de grupo. As portas internas são especificadas usando a opção de linha de comando **-catalogServiceEndPoints**:

`-catalogServiceEndPoints <server:host:port:port,server:host:port:port>`

Com a opção de linha de comandos **-catalogServiceEndPoints**, é possível configurar duas portas por servidor. As portas do IIOP são configuradas usando as seguintes opções de linha de comandos:

```
-listenerHost <nome_do_host>
-listenerPort <porta>
```

Quando cada JVM de serviço de catálogo for iniciado, especifique o conjunto completo dos terminais de serviço de catálogo junto com uma porta listener única para essa JVM.

Pontos de extremidade JVM do contêiner

O contêiner do Java Virtual Machines usa duas portas. Uma porta destina-se para uso interno e a outra porta para comunicação IIOP. Normalmente, o contêiner do Java Virtual Machines localiza automaticamente portas não utilizadas e, em seguida, são configuradas para usar duas portas criadas dinamicamente. Esse processo automático minimiza a configuração. Porém, se firewalls estiverem sendo usados ou se você desejar configurar explicitamente as portas, poderá usar uma opção de linha de comandos para especificar a porta do Oject Request Broker (ORB) a ser usada:

```
-listenerHost <nome_do_host>
-listenerPort <porta>
```

Com essas opções de linha de comando, é possível especificar o nome do host (importante para conexão correta com a placa de rede) e a porta a ser especificada para essa JVM. É possível especificar a porta interna com o argumento da linha de comandos **-haManagerPort**. Porém, para uma configuração mais simples, é possível deixar que o tempo de execução escolha as portas.

Procedimento

1. Inicie o primeiro servidor de catálogos no hostA. A seguir há um exemplo do comando:

   ```
   ./startOgServer.sh cs1 -listenerHost hostA -listenerPort
   2809 -catalogServiceEndPoints cs1:hostA:6601:6611,cs2:hostB:6601:6611
   ```

2. Inicie o segundo servidor de catálogos no hostB. A seguir há um exemplo do comando:

   ```
   ./startOgServer.sh cs2 -listenerHost hostB -listenerPort 2809
   -catalogServiceEndPoints cs1:hostA:6601:6611,cs2:hostB:6601:6611
   ```

   Os clientes precisam apenas saber os terminais do listener do serviço de catálogo. Os clientes recuperam os terminais para o contêiner Java Virtual Machines, que são os Java Virtual Machines que mantêm os dados, automaticamente a partir do serviço de catálogo. Para se conectar com o serviço de catálogo no exemplo anterior, o cliente deve transmitir a seguinte lista de pares `host:port` para a API de conexão:

   ```
   hostA:2809,hostB:2809
   ```

   Para que uma JVM de contêiner use o serviço de catálogo de exemplo, use o seguinte comando:

   ```
   ./startOgServer.sh c0 -catalogServiceEndPoints hostA:2809,hostB:2809
   ```

Configurando Portas em um Ambiente do WebSphere Application Server

O serviço de catálogo executa uma única instância dentro no gerenciador de implementação, por padrão, e usa a porta de autoinicialização do Internet Inter-ORB Protocol (IIOP) para o gerenciador de implementação do Java Virtual Machine.

Sobre Esta Tarefa

Os aplicativos da Web ou os aplicativos Enterprise JavaBeans™ (EJB) em uma célula podem se conectar às grades que estão dentro da mesma célula usando a chamada de conexão `null,null` em vez de especificar as portas de autoinicialização de serviço de catálogo.

Se o domínio do serviço de catálogo no WebSphere Application Server for hospedado pelo gerenciador de implementação, os clientes fora da célula (incluindo clientes do Java Platform, Standard Edition) deverão se conectar ao serviço de catálogo usando o nome do host do gerenciador de implementação e a porta de autoinicialização IIOP. Quando o serviço de catálogo for executado nas células do WebSphere Application Server enquanto os clientes forem executados fora das células, consulte as páginas de configuração de domínio do eXtreme Scale no console administrativo do WebSphere Application Server para obter as informações necessárias para apontar um cliente para o serviço de catálogo. Se os clientes estiverem nas células do WebSphere Application Server, será possível recuperar as portas diretamente a partir da interface do CatalogServerProperties.

7.1+ Embora você ainda possa usar a propriedade `catalog.services.cluster` para localizar portas de conexão do cliente, a técnica foi reprovada. Se você usar esta técnica, mas não localizar a entrada `catalog.services.cluster`, use a porta IIOP no gerenciador de implementação para a conexão do cliente.

O eXtreme Scale reutiliza as portas Distribution and Consistency Services (DCS) do gerenciador de alta disponibilidade para associação ao grupo. As portas do Java Management Extensions (JMX) também são reutilizadas.

Configurando Object Request Brokers

Use o arquivo orb.properties para transmitir as propriedades que são usadas pelo Object Request Broker (ORB) para modificar o comportamento de transporte da grade. O WebSphere® eXtreme Scale usa o Object Request Broker (ORB) para ativar a comunicação entre processos. Nenhuma ação é necessária para usar o Object Request Broker (ORB) fornecido pelo WebSphere eXtreme Scale ou WebSphere Application Server para seus servidores WebSphere eXtreme Scale. Esta seção esboça algumas considerações de ajuste e outras tarefas de configuração relacionadas ao ORB que é possível desejar ou necessitar executar.

Arquivo de Propriedades ORB

O arquivo `orb.properties` é usado para transmitir as propriedades que são usadas pelo Object Request Broker (ORB) para modificar o comportamento de transporte da grade.

Local

O arquivo `orb.properties` está no diretório `java/jre/lib`. Ao modificar o arquivo em um diretório WebSphere Application Server `java/jre/lib`, os servidores de aplicativos que são configurados nessa instalação também usam as configurações do arquivo.

Configurações de Linha de Base

As seguintes configurações são uma linha de base ideal mas não necessariamente as melhores configurações para cada ambiente. É necessário entender as configurações para ajudar a tomar uma boa decisão sobre quais valores são apropriados no seu ambiente.

```
com.ibm.CORBA.RequestTimeout=30
com.ibm.CORBA.ConnectTimeout=10
com.ibm.CORBA.FragmentTimeout=30
com.ibm.CORBA.ThreadPool.MinimumSize=256
com.ibm.CORBA.ThreadPool.MaximumSize=256
com.ibm.CORBA.ThreadPool.IsGrowable=false
com.ibm.CORBA.ConnectionMultiplicity=1
com.ibm.CORBA.MinOpenConnections=1024
com.ibm.CORBA.MaxOpenConnections=1024
com.ibm.CORBA.ServerSocketQueueDepth=1024
com.ibm.CORBA.FragmentSize=0
com.ibm.CORBA.iiop.NoLocalCopies=true
com.ibm.CORBA.NoLocalInterceptors=true
```

Descrições de Propriedades

Configurações de Tempo Limite

As seguintes configurações referem-se à quantia de tempo em que o ORB aguardará antes que as operações sobre pedidos sejam canceladas.

Tempo Limite do Pedido

Nome da Propriedade: com.ibm.CORBA.RequestTimeout

Valor: Valor de número inteiro para número de segundos.

Descrição: Indica quantos segundos um pedido deve aguardar por uma resposta antes de ser cancelado. Essa propriedade influencia a quantia de tempo em que um cliente precisa para efetuar failover se ocorrer uma falha de interrupção de rede. Se você configurar essa propriedade muito baixo, os pedidos poderão atingir o tempo limite involuntariamente. Considere cuidadosamente o valor dessa propriedade para evitar tempos limites involuntários.

Tempo Limite de Conexão

Nome da Propriedade: com.ibm.CORBA.ConnectTimeout

Valor: Valor de número inteiro para número de segundos.

Descrição: Indica quantos segundos uma tentativa de conexão de soquete deve aguardar antes de ser cancelada. Essa propriedade, como o tempo limite de pedido, pode influenciar no tempo em que um cliente precisa para efetuar failover se ocorrer uma falha de interrupção de rede. Em geral, configure essa propriedade para um valor menor do que o valor do tempo limite de pedido porque a quantia de tempo para estabelecer uma conexão deve ser relativamente constante.

Tempo Limite de Fragmento

Nome da Propriedade: com.ibm.CORBA.FragmentTimeout

Valor: Valor de número inteiro para número de segundos.

Descrição: Indica quantos segundos um pedido de fragmento deve aguardar antes de ser cancelado. Essa propriedade é semelhante à propriedade de tempo limite de pedido.

Configurações do Conjunto de Encadeamentos

Essas propriedades restringem o tamanho do conjunto de encadeamentos a um número específico de encadeamentos. Os encadeamentos são usados pelo ORB para distribuir os pedidos do servidor depois de serem recebidos no soquete. Configurar esses valores da propriedade muito baixo resulta em um aumento de profundidade da fila de soquete e possivelmente a ocorrência de tempos limites.

Multiplicidade de Conexão

Nome da Propriedade: com.ibm.CORBA.ConnectionMultiplicity

Valor: Valor de número inteiro para número de conexões entre o cliente e o servidor. O valor padrão é 1. Configurar um valor maior define uma multiplicidade entre várias conexões.**Descrição:** Permite que o ORB use várias conexões em qualquer servidor. Na teoria, configurar esse valor deve promover paralelismo sobre as conexões. Na prática, o desempenho não é beneficiado pela configuração da multiplicidade de conexão. Não configure esse parâmetro.

Conexões Abertas

Nomes da Propriedade: com.ibm.CORBA.MinOpenConnections, com.ibm.CORBA.MaxOpenConnections

Valor: Valor de número inteiro para número de conexões.**Descrição:** Especifica um número mínimo e máximo de conexões abertas. O ORB mantém um cache de conexões que foram estabelecidas com os clientes. Essas conexões são limpas quando o valor de com.ibm.CORBA.MaxOpenConnections for transmitido. A limpeza das conexões pode prejudicar o comportamento na grade.

É Aumentável

Nome da Propriedade: com.ibm.CORBA.ThreadPool.IsGrowable

Valor: Booleano; configurado para `true` ou `false`.**Descrição:** Se ativado, permite que o conjunto de encadeamentos que o ORB utiliza para pedidos recebidos aumente além do que o conjunto suporta. Se o tamanho do conjunto for excedido, novos encadeamentos serão criados para manipular o pedido, mas os encadeamentos não serão agrupados.

Profundidade da Fila de Soquete do Servidor

Nome da Propriedade: com.ibm.CORBA.ServerSocketQueueDepth

Valor: Valor de número inteiro para número de conexões.**Descrição:** Especifica o comprimento da fila para conexões recebidas dos clientes. As filas ORB recebem conexões dos clientes. Se a fila estiver cheia, as conexões serão recusadas. Recusar conexões pode prejudicar o comportamento na grade.

Tamanho do Fragmento

Nome da Propriedade: com.ibm.CORBA.FragmentSize

Valor: Um número inteiro que especifica o número de bytes. O padrão é 1024.**Descrição:** Especifica o tamanho máximo do pacote que o ORB usa ao enviar um pedido. Se um pedido for maior que o limite de tamanho de fragmento, esse pedido será dividido em fragmentos de pedido e enviados, cada um, separadamente e remontados no servidor. Os pedidos de fragmentação são úteis em redes não-confiáveis onde os pacotes podem precisar ser reenviados. Porém, se a rede for confiável, dividir os pedidos em fragmentos pode causar sobrecarga.

Nenhuma Cópia Local

Nome da Propriedade: com.ibm.CORBA.iiop.NoLocalCopies

Valor: Booleano; configurado para `true` ou `false`. **Descrição:** Especifica se o ORB é transmitido por referência. O ORB usa a chamada transmitir por valor por padrão. A chamada transmitir por valor causa um custo de serialização extra desnecessário no caminho quando uma interface for chamada localmente. Ao configurar esse valor para true, o ORB usa um método transmitir por referência que é mais eficiente do que a chamada transmitir por valor.

Nenhum Interceptor Local

Nome da Propriedade: com.ibm.CORBA.NoLocalInterceptors

Valor: Booleano; configurado para `true` ou `false`. **Descrição:** Especifica se o ORB chama os interceptores de pedido mesmo ao fazer pedidos locais (intraprocesso). Os interceptores que o WebSphere eXtreme Scale usa são para segurança e os identificadores de rota não serão necessários se o pedido for tratado dentro do processo. Os interceptores que estão entre os processos são necessários apenas para as operações de chamada de procedimento remoto (RPC). Ao configurar 'Nenhum interceptor local', é possível evitar a sobrecarga extra ao usar os interceptores locais.

Atenção: Se a segurança do WebSphere eXtreme Scale estiver ativada, configure o valor da propriedade com.ibm.CORBA.NoLocalInterceptors para `false`. A infraestrutura de segurança usa interceptores para autenticação.

Quando quiser aplicar segurança de transporte entre os clientes e servidores do ObjectGrid, é necessário incluir mais propriedades no arquivo `orb.properties`. Para obter mais informações sobre essas propriedades, consulte a seção sobre o arquivo `orb.properties` para suporte de segurança de transporte no “Transport Layer Security e Secure Sockets Layer” na página 363.

Propriedades do ORB e Configurações do Descritor de Arquivo

As considerações de ajuste incluem propriedades do Object Request Broker (ORB) e configurações do descritor de arquivo.

Propriedades do ORB

O ORB é usado pelo WebSphere eXtreme Scale para se comunicar por meio de uma pilha TCP. O arquivo `orb.properties` necessário está no diretório `java/jre/lib`. Para um carga pesada de objetos grandes, ative a fragmentação de ORB especificando as seguintes configurações:

```
com.ibm.CORBA.FragmentSize=<tamanho correto>
```

Evite o crescimento de ThreadPool especificando as seguintes configurações:
`com.ibm.CORBA.ThreadPool.IsGrowable=false`

Configure os tempos limite para evitar excesso de encadeamentos em uma situação anormal especificando as seguintes configurações:

- com.ibm.CORBA.RequestTimeout
- com.ibm.CORBA.ConnectTimeout
- com.ibm.CORBA.FragmentTimeout

Descritor de Arquivo

Sistemas UNIX® e Linux possuem um limite para a quantidade de arquivos abertos permitidos por processo. O sistema operacional especifica a quantidade de arquivos abertos permitidos. Se este valor for configurado com um valor muito baixo, um erro de alocação de memória ocorrerá no AIX, e muitos arquivos abertos serão registrados no log.

Na janela de terminal do sistema UNIX, configure este valor para um valor mais alto que o valor do sistema padrão. Para máquinas SMP maiores com clones, configure para ilimitado.

Para configurações AIX, defina este valor como `-1` (ilimitado) com o comando `ulimit -n -1`.

Para configurações Solaris, defina este valor como `16384` com o comando `ulimit -n 16384`.

Para exibir o valor atual, use o comando `ulimit –a`.

Ativando a NIO ou ChannelFramework no ORB

O WebSphere eXtreme Scale fornece uma propriedade de servidor para configurar o TransportMode para ChannelFramework para ativar a non-blocking I/O (NIO) no ORB.

Antes de Iniciar

Localize o arquivo de propriedades do servidor existente ou cria um arquivo de propriedades do servidor. É possível ativar ChannelFramework no serviço de catálogo e nos servidores de contêiner. Para obter mais detalhes, consulte o Arquivo de propriedades do servidor.

Sobre Esta Tarefa

Atualmente, é possível executar com NIO ou ChannelFramework em cenários independentes do WebSphere eXtreme Scale. Para executar com a Non-blocking I/O (NIO) no ORB IBM, você deve configurar o TransportMode do ORB IBM para ChannelFramework. Por padrão, o ORB IBM é executado no modo Conectável. O WebSphere eXtreme Scale fornece uma propriedade de servidor para configurar o TransportMode para ChannelFramework.

Importante:

O WebSphere Application Server suporta o modo Conectável apenas em releases atuais. Quando o WebSphere eXtreme Scale é executado integrado com o WebSphere Application Server, ele deve seguir o modo Conectável. Como o

WebSphere eXtreme Scale também usa o SSL/Transport Security do WebSphere Application Server, ele também suporta atualmente apenas o modo Conectável.

Procedimento

1. Inclua a propriedade enableChannelFramework=true no seu arquivo de propriedades do servidor.
2. Certifique-se de que o arquivo de propriedades do servido não contradiga o arquivo de propriedades do ORB.

   Se o arquivo de propriedades do servidor ativar o ChannelFramework TransportMode, mas o TransportMode for configurado como Conectável no arquivo orb.properties, o servidor não substituirá a configuração de orb.properties. Você verá uma mensagem de aviso no log de que há duas configurações. Para permitir que a propriedade enableChannelFramework=true tenha efeito, ajuste as propriedades que indicam que TransportMode está configurado como Conectável: altere com.ibm.CORBA.TransportMode=Pluggable para ChannelFramework ou remova a propriedade.
3. Forneça o arquivo de propriedades do servidor atualizado no serviço de catálogo ou na inicialização do servidor de contêiner. Para obter mais detalhes sobre como usar os arquivos de propriedades do servidor para iniciar um servidor, consulte Arquivo de propriedades do servidor.

Resultados

Quando um serviço de catálogo ou servidor de contêiner usar o channelFramework TransportMode, ele imprimirá a seguinte mensagem para o log.

```
CWOBJ0052I: A propriedade IBM ORB TransportMode foi configurada para ChannelFramework
```

Se você vir a seguinte mensagem no log, examine suas propriedades do ORB, conforme descrito anteriormente.

```
CWOBJ0055W: A propriedade IBM ORB TransportMode foi configurada para ChannelFramework
no arquivo de propriedades do servidor, mas o arquivo orb.properties existente já
tinha um TransportMode configurado. O TransportMode não será substituído.
```

Observe que quando você ativa ChannelFramework, o valor máximo para ServerSocketQueueDepth é 512. Se a configuração orb.properties ServerSocketQueueDepth for maior que 512, o servidor automaticamente configurará orb.properties ServerSocketQueueDepth para 512 e o notificará imprimindo uma mensagem informativa para o log. Nenhuma ação é necessária.

```
CWOBJ0053I: A propriedade IBM ORB ServerSocketQueueDepth foi configurada para
512 para ser executada
corretamente com o ChannelFramework TransportMode.
```

Utilizando o Object Request Broker com Processos do WebSphere eXtreme Scale Independentes

É possível usar o WebSphere eXtreme Scale com aplicativos que usam o Object Request Broker (ORB) diretamente em ambientes que não contêm o WebSphere Application Server ou o WebSphere Application Server Network Deployment.

Antes de Iniciar

Se você usar o ORB dentro do mesmo processo do eXtreme Scale ao executar aplicativos ou outros componentes e estruturas que não estejam incluídos com o eXtreme Scale, poderá ser necessário concluir tarefas adicionais para garantir que o

eXtreme Scale seja executado corretamente no seu ambiente.

Sobre Esta Tarefa

Inclua a propriedade **ObjectGridInitializer** no arquivo `orb.properties` para inicializar o uso do ORB em seu ambiente. Use o ORB para ativar a comunicação entre os processos do eXtreme Scale e outros processos que estão em seu ambiente. O arquivo `orb.properties` está no diretório `java/jre/lib`. Consulte “Arquivo de Propriedades ORB” na página 192 para obter as descrições das propriedades e configurações.

Procedimento

Digite a seguinte linha e salve as alterações:

```
org.omg.PortableInterceptor.ORBInitializerClass.com.ibm.ws.objectgrid.corba.ObjectGridInitializer
```

Resultados

O eXtreme Scale inicializa corretamente o ORB e coexiste com outros aplicativos para os quais o ORB está ativado.

Para usar uma versão customizada do ORB com o eXtreme Scale, consulte “Configurando um Object Request Broker Customizado”.

Configurando um Object Request Broker Customizado

O WebSphere eXtreme Scale usa o Object Request Broker (ORB) para ativar a comunicação entre processos. Nenhuma ação é necessária para usar o Object Request Broker (ORB) fornecido pelo WebSphere eXtreme Scale ou WebSphere Application Server para seus servidores WebSphere eXtreme Scale. Um esforço pequeno é necessário para usar os mesmos ORBs para seus clientes do WebSphere eXtreme Scale. Se for necessário usar um ORB "customizado", o ORB fornecido com o IBM SDK será uma boa opção, embora seja necessário executar alguma configuração, conforme descrito aqui. Os ORBs de outros fornecedores podem ser usados, também com configuração.

Antes de Iniciar

Decida se você usará o ORB fornecido com o WebSphere eXtreme Scale ou o WebSphere Application Server, o ORB fornecido com o IBM SDK ou um ORB de terceiro.

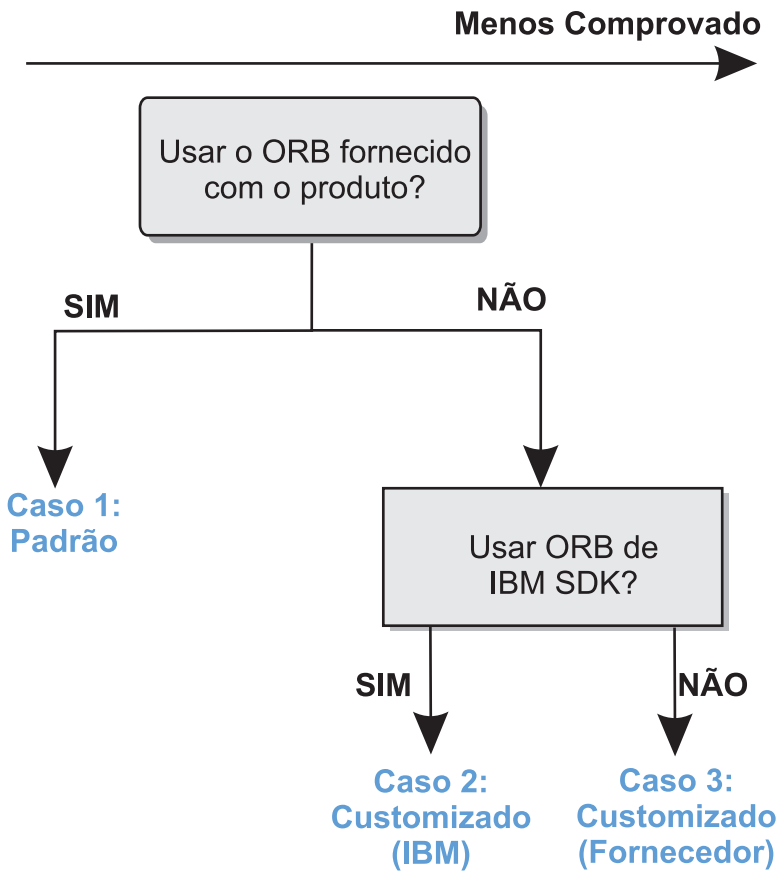


Figura 10. Escolhendo um ORB

É possível tomar decisões separadas para os processos do servidor do WebSphere eXtreme Scale e para os processos do cliente do WebSphere eXtreme Scale. Apesar de o eXtreme Scale suportar Developer Kits da maioria dos fornecedores, recomenda-se que você use o ORB que é fornecido com o eXtreme Scale para seus processos do servidor e do cliente. O eXtreme Scale não suporta o ORB que é fornecido com o Sun Microsystems Java Development Kit (JDK).

Sobre Esta Tarefa

Familiarize-se com a configuração que é necessária para usar o ORB de sua escolha.

Caso 1: ORB Padrão

- Para seus processos do servidor do WebSphere eXtreme Scale, nenhuma configuração é necessária para usar o ORB fornecido com o WebSphere eXtreme Scale ou o WebSphere Application Server.
- Para seus processos do cliente do WebSphere eXtreme Scale, uma configuração mínima de caminho de classe é necessária para usar o ORB fornecido com o WebSphere eXtreme Scale ou o WebSphere Application Server.

Caso 2: ORB Customizado (IBM)

Para configurar seus processos do cliente do WebSphere eXtreme Scale para usar o ORB fornecido com o IBM SDK, consulte as instruções posteriormente neste tópico. É possível usar o ORB IBM se estiver usando o IBM SDK ou outro kit de desenvolvimento.

Usar o IBM SDK Versão 5 (ou posterior) requer um menor esforço de configuração que o IBM SDK Versão 1.4.2.

Caso 3: ORB Customizado (Fornecedor de terceiro)
Usar um ORB de terceiro para seus processos do cliente do WebSphere eXtreme Scale é a última opção testada. Quaisquer problemas que você tiver ao usar os ORBs a partir de fornecedores de software independentes deverão ser reproduzíveis com o IBM ORB e compatíveis com o JRE antes de entrar em contato com o suporte.

O ORB fornecido com o Sun Microsystems Java Development Kit (JDK) não é suportado.

Procedimento

- Configure seus processos do cliente para usarem um dos ORBs padrão (**Caso 1)**.

  `-jvmArgs –Djava.endorsed.dirs=`*`default_ORB_directory`*

- Configure os processos do cliente ou do servidor para usarem o IBM SDK, Versão 5 **(Caso 2)**.
  1. Copie os arquivos JAR do ORB para um diretório vazio, a partir de agora referido como *custom_ORB_directory*.
     - `ibmorb.jar`
     - `ibmorbapi.jar`

     Dica: Se estiver usando um ORB customizado de um fornecedor de terceiro (**Caso 3**), esses arquivos JAR adicionais poderão ser necessários:
     - `ibmext.jar`
     - `ibmcfw.jar`, se estiver usando o ORB NIO
  2. Especifique o *custom_ORB_directory* como um diretório endossado nos scripts que iniciam o comando Java.

     Dica: Se seus comandos Java já fizerem referência a um diretório endossado, outra opção será colocar o *custom_ORB_directory* no diretório endossado existente – então, você não precisará atualizar os scripts. Se você decidir atualizar os scripts de qualquer forma, certifique-se de pré-anexar o *custom_ORB_directory* ao seu argumento `–Djava.endorsed.dirs=` existente, em vez de substituir completamente o argumento existente.
     - Atualize scripts para um ambiente independente do eXtreme Scale.

       Edite o caminho para a variável *OBJECTGRID_ENDORSED_DIRS* no arquivo `setupCmdLine.bat|sh` para fazer referência a *custom_ORB_directory*. Salve as alterações.
     - Atualize scripts quando o eXtreme Scale estiver integrado em um ambiente do WebSphere Application Server.

       Inclua a seguinte propriedade de sistema e os parâmetros no script `startOgServer`:

       `-jvmArgs –Djava.endorsed.dirs=`*`custom_ORB_directory`*
     - Atualize os scripts customizados que você usa para iniciar um processo do aplicativo cliente ou um processo do servidor.

       `-Djava.endorsed.dirs=`*`custom_ORB_directory`*

- Configure processos do cliente ou do servidor para usarem o IBM SDK, Versão 1.4.2 **(Caso 2)**. Se o seu ambiente contiver um SDK Versão 1.4.2, integre o ORB IBM no SDK especificado.
  1. Faça download e extraia o ORB de um IBM SDK, Versão 1.4.2.

Se nenhum IBM SDK estiver disponível para sua plataforma, faça download e extraia o IBM Developer Kit para Linux, Java Technology Edition. Consulte IBM Developer Kits.

2. Copie os arquivos JAR do ORB para o SDK de destino. Copie os arquivos `java/jre/lib/ibmorb.jar` e `java/jre/lib/ibmorbapi.jar` para o diretório `java/jre/lib/ext` no SDK de destino.
3. Atualize as propriedades do ORB. Crie ou edite o arquivo `orb.properties`, que está no diretório `java/jre/lib` do SDK. Inclua as seguintes propriedades ou verifique se elas existem no arquivo:

   ```
   org.omg.CORBA.ORBClass=com.ibm.CORBA.iiop.ORB
   org.omg.CORBA.ORBSingletonClass=com.ibm.rmi.corba.ORBSingleton
   ```

   Para obter descrições das propriedades e das configurações, consulte "Arquivo de Propriedades ORB" na página 192.
4. Certifique-se de que o analisador XML esteja disponível.
   - Faça download de Xerces2 Java 2.9 do The Apache Xerces Project - Downloads.
   - Localize os arquivos `xercesImpl.jar` e `xml-apis.jar`.
   - Copie os arquivos para o diretório `lib/ext`.

Configurando Clientes

É possível configurar o WebSphere® eXtreme Scale para ser executado em um ambiente independente ou pode configurar o eXtreme Scale para ser executado em um ambiente com o WebSphere Application Server ou o WebSphere Application Server Network Deployment. Para que uma implementação do eXtreme Scale selecione alterações de configuração no lado da grade do servidor, será necessário reiniciar os processos para que essas alterações tomem efeito em vez de serem aplicadas dinamicamente. Porém, no lado do cliente, embora não seja possível alterar as definições de configuração para uma instância de cliente existente, será possível criar um novo cliente com as configurações necessárias ao usar um arquivo XML ou fazer isso programaticamente. Ao criar um cliente, será possível substituir as configurações padrão fornecidas com a configuração do servidor atual.

É possível configurar um cliente do eXtreme Scale das seguintes maneiras, sendo que cada uma delas pode ser realizada com um arquivo XML de substituição do cliente ou programaticamente:

- Configuração XML
- Configuração Programática
- Configuração da Estrutura Spring
- Desativando o Cache Local

É possível substituir os seguintes plug-ins em um cliente:

- **Plug-ins do ObjectGrid**
  - Plug-in TransactionCallback
  - Plug-in ObjectGridEventListener
- **Plug-ins do BackingMap**
  - Plug-in Evictor
  - Plug-in MapEventListener
  - Atributo numberOfBuckets
  - Atributo ttlEvictorType

– Atributo timeToLive

Arquivo de Propriedades do Cliente

É possível criar um arquivo de propriedades com base nos requisitos para os processos do cliente do eXtreme Scale.

Amostra do Arquivo de Propriedades do Cliente

É possível usar o arquivo `sampleClient.properties` que esteja no diretório *extremescale_root*`\properties` para criar seu arquivo de propriedades.

Especificando um Arquivo de Propriedades do Cliente

É possível especificar o arquivo de propriedades do cliente de uma das seguintes formas. Especificar uma configuração ao usar um dos itens recentes na lista substitui a configuração anterior. Por exemplo, se você especificar um valor de propriedade de sistema para o arquivo de propriedades do cliente, as propriedades nesse arquivo substituirão os valores no arquivo `objectGridClient.properties` que estiver no caminho de classe.

1. Como um arquivo nomeado corretamente em qualquer lugar no caminho de classe. Colocar esse arquivo no diretório atual do sistema não é suportado:
   `objectGridClient.properties`
2. Como uma propriedade do sistema em uma configuração do WebSphere Application Server ou independente. Este valor pode especificar um arquivo no diretório atual do sistema, mas não um arquivo no caminho de classe:
   `-Dobjectgrid.client.props=`*`file_name`*
3. Como uma substituição programática usando o método ClientClusterContext.getClientProperties. Os dados no objeto são preenchidos com os dados a partir dos arquivos de propriedades. Não é possível configurar as propriedades de segurança com esse método.

Propriedades do Cliente

7.1+ listenerHost
Especifica o nome do host para o qual o Object Request Broker (ORB) deve conectar-se.

Se sua configuração envolver várias placas de rede, configure o host e a porta do listener para que o Object Request Broker na JVM saiba o endereço IP ao qual se ligar. Para o cliente, use o arquivo de propriedades do cliente. A negligência ao especificar qual endereço IP usar produz sintomas como tempos de conexão esgotados, falhas de API incomuns e clientes que parecem interrompidos.

7.1+ listenerPort
Especifica o número da porta para a qual o Object Request Broker (ORB) deve conectar-se.

preferLocalProcess
Especifica se o processo local é preferido para roteamento. Quando configurado para true, os pedidos são roteados para os shards que são colocados no mesmo processo do cliente quando apropriado.

Padrão: `true`

preferLocalHost
Especifica se o host local é preferido para roteamento. Quando configurado para true, os pedidos são roteados para os shards que são colocados no mesmo host do cliente quando apropriado.

Padrão: `true`

preferZones
Especifica uma lista de zonas de roteamento preferidas. Cada zona especificada é separada por vírgula no formato:
`preferZones=ZoneA,ZoneB,ZoneC`

Padrão: Nenhum valor

requestRetryTimeout
Especifica quanto tempo um pedido será tentado novamente (em milissegundos). Utilize um dos seguintes valores válidos:

- O valor `0` indica que o pedido deve falhar rapidamente e ignorar a lógica de nova tentativa interna.
- O valor `-1` indica que o tempo limite da nova tentativa do pedido não foi configurado, significando que a duração do pedido é controlada pelo tempo limite da transação. (Default)
- O valor `0` indica o valor do tempo limite de entrada do pedido em milissegundos. As exceções que não puderem ocorrer mesmo se for tentado novamente como uma exceção DuplicateException serão retornadas imediatamente. O tempo limite da transação ainda é usado como o tempo máximo de espera.

Propriedades de Segurança do Cliente

Propriedades gerais de segurança

securityEnabled
Ativa a segurança do cliente do WebSphere eXtreme Scale. Essa configuração de segurança ativada deve corresponder com a configuração securityEnabled no arquivo de propriedades do servidor WebSphere eXtreme Scale. Se as configurações não corresponderem, uma exceção ocorrerá.

Padrão: `false`

Propriedades de Configuração de Autenticação de Credencial

credentialAuthentication
Especifica o suporte de autenticação da credencial do cliente. Utilize um dos seguintes valores válidos:

- `Nunca`: O cliente não suporta a autenticação de credencial.
- `Suportado`: O cliente suporta a autenticação de credencial se o servidor também suportar a autenticação de credencial. (Default)
- `Necessário`: O cliente requer autenticação de credencial.

authenticationRetryCount
Especifica o número de vezes em que a autenticação é tentada novamente se a credencial expirar. Se o valor for configurado para 0, as tentativas de autenticação não serão feitas novamente.

Padrão: `3`

credentialGeneratorClass
Especifica o nome da classe que implementa a interface

com.ibm.websphere.objectgrid.security.plugins.CredentialGenerator. Essa classe é usada para obter credenciais para os clientes.

Padrão: Nenhum valor

credentialGeneratorProps
Especifica as propriedades para a classe de implementação CredentialGenerator. As propriedades são configuradas para o objeto com o método setProperties(String). O valor credentialGeneratorprops é usado apenas se o valor da propriedade credentialGeneratorClass não for nulo.

Propriedades da configuração de segurança da camada de transporte

transportType
Especifica o tipo de transporte do cliente. Os valores possíveis são:

- `TCP/IP`: Indica que o cliente suporta apenas conexões TCP/IP.
- `SSL-Suportado`: Indica que o cliente suporta ambas as conexões TCP/IP e Secure Sockets Layer (SSL). (Default)
- `SSL-Necessário`: Indica que o cliente requer as conexões SSL.

Propriedades de Configuração SSL

alias Especifica o nome do alias no armazenamento de chaves. Esta propriedade será utilizada se o armazenamento de chaves tiver vários certificados de pares de chaves e você desejar selecionar um dos certificados.

Padrão: nenhum valor

contextProvider
Especifica o nome do provedor de contexto para o serviço de confiança. Se você indicar um valor inválido, ocorrerá uma exceção de segurança que indica que o tipo de provedor de contexto está incorreto.

Valores válidos: `IBMJSSE2`, `IBMJSSE`, `IBMJSSEFIPS`, etc.

protocol
Indica o tipo de protocolo de segurança a ser usado para o cliente. Configure esse valor de protocolo baseado no provedor Java Secure Socket Extension (JSSE) que será usado. Se você indicar um valor inválido, ocorrerá uma exceção de segurança que indica que o valor de protocolo está incorreto.

Valores válidos: `SSL`, `SSLv2`, `SSLv3`, `TLS`, `TLSv1`, etc.

keyStoreType
Indica o tipo de armazenamento de chaves. Se você indicar um valor inválido, ocorrerá uma exceção de segurança no tempo de execução.

Valores válidos: `JKS`, `JCEK`, `PKCS12`, etc.

trustStoreType
Indica o tipo de armazenamento de confiança. Se você indicar um valor inválido, ocorrerá uma exceção de segurança no tempo de execução.

Valores válidos: `JKS`, `JCEK`, `PKCS12`, etc.

keyStore
Especifica um caminho completo para o arquivo de armazenamento de chaves.

Exemplo:

```
etc/test/security/client.private
```

trustStore
Especifica um caminho completo para o arquivo de armazenamento de confiança.

Exemplo:

```
etc/test/security/server.public
```

keyStorePassword
Especifica a senha da cadeia para o armazenamento de chaves. É possível codificar este valor ou usar o valor real.

trustStorePassword
Especifica uma senha de cadeia para o armazenamento de confiança. É possível codificar este valor ou usar o valor real.

Configurando Clientes com o WebSphere eXtreme Scale

É possível configurar um cliente eXtreme Scale com base em seus requisitos, como a necessidade de substituir configurações.

Configure o Cliente com XML

Um arquivo XML ObjectGrid pode ser usado para alterar configurações no lado do cliente. Para alterar as configurações em um cliente eXtreme Scale, você deve criar um arquivo XML ObjectGrid semelhante na estrutura ao arquivo que foi usado para o servidor eXtreme Scale.

Suponha que o seguinte arquivo XML tenha sido emparelhado com um arquivo XML de política de implementação e que esses arquivos tenham sido usados para iniciar um servidor eXtreme Scale.

companyGridServerSide.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
    xmlns="http://ibm.com/ws/objectgrid/config">

    <objectGrids>
        <objectGrid name="CompanyGrid">
            <bean id="TransactionCallback"
                className="com.company.MyTxCallback" />
            <bean id="ObjectGridEventListener"
                className="com.company.MyOgEventListener" />
            <backingMap name="Customer"
                pluginCollectionRef="customerPlugins"/>
            <backingMap name="Item" />
            <backingMap name="OrderLine" numberOfBuckets="1049"
                timeToLive="1600" ttlEvictorType="LAST_ACCESS_TIME" />
            <backingMap name="Order" lockStrategy="PESSIMISTIC"
                pluginCollectionRef="orderPlugins" />
        </objectGrid>
    </objectGrids>

    <backingMapPluginCollections>
        <backingMapPluginCollection id="customerPlugins">
            <bean id="Evictor"
                className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor" />
            <bean id="MapEventListener"
                className="com.company.MyMapEventListener" />
        </backingMapPluginCollection>
        <backingMapPluginCollection id="orderPlugins">
            <bean id="MapIndexPlugin"
                className="com.company.MyMapIndexPlugin" />
        </backingMapPluginCollection>
    </backingMapPluginCollections>
</objectGridConfig>
```

Em um servidor eXtreme Scale, a instância do ObjectGrid denominada CompanyGrid se comporta como definido pelo arquivo companyGridServerSide.xml. Por padrão, o cliente CompanyGrid tem as mesmas configurações que a instância CompanyGrid em execução no servidor. Entretanto, algumas das configurações podem ser substituídas no cliente da seguinte forma:

1. Crie uma instância ObjectGrid específica do cliente.
2. Copie o arquivo XML ObjectGrid que foi utilizado para abrir o servidor.
3. Edite o novo arquivo para customizar para o lado do cliente.
   - Para configurar ou atualizar qualquer um dos atributos no cliente, especifique um novo valor ou altere o valor existente.
   - Para remover um plug-in do cliente, use a cadeia vazia como o valor para o atributo className.
   - Para alterar um plug-in existente, especifique um novo valor para o atributo className.
   - Também é possível incluir qualquer plug-in suportado para uma substituição de cliente: TRANSACTION_CALLBACK, OBJECTGRID_EVENT_LISTENER, EVICTOR, MAP_EVENT_LISTENER.
4. Crie um cliente com o arquivo XML de substituição de cliente recém criado.

O seguinte arquivo XML ObjectGrid pode ser usado para especificar alguns dos atributos e plug-ins no cliente CompanyGrid.

companyGridClientSide.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
    xmlns="http://ibm.com/ws/objectgrid/config">

    <objectGrids>
        <objectGrid name="CompanyGrid">
            <bean id="TransactionCallback"
                className="com.company.MyClientTxCallback" />
            <bean id="ObjectGridEventListener" className="" />
            <backingMap name="Customer" numberOfBuckets="1429"
                pluginCollectionRef="customerPlugins"/>
            <backingMap name="Item" />
            <backingMap name="OrderLine" numberOfBuckets="701"
                timeToLive="800" ttlEvictorType="LAST_ACCESS_TIME" />
            <backingMap name="Order" lockStrategy="PESSIMISTIC"
                pluginCollectionRef="orderPlugins" />
        </objectGrid>
    </objectGrids>

    <backingMapPluginCollections>
        <backingMapPluginCollection id="customerPlugins">
            <bean id="Evictor"
                className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor" />
            <bean id="MapEventListener" className="" />
        </backingMapPluginCollection>
        <backingMapPluginCollection id="orderPlugins">
            <bean id="MapIndexPlugin"
                className="com.company.MyMapIndexPlugin" />
        </backingMapPluginCollection>
    </backingMapPluginCollections>
</objectGridConfig>
```

- TransactionCallback no cliente é com.company.MyClientTxCallback em vez da configuração do lado do servidor de com.company.MyTxCallback.
- O cliente não tem um plug-in ObjectGridEventListener porque o valor className é a cadeia vazia.
- O cliente configura numberOfBuckets como 1429 para Customer backingMap, retém seu plug-in Evictor e remove o plug-in MapEventListener.
- Os atributos numberOfBuckets e timeToLive de OrderLine backingMap mudaram

- Embora um atributo lockStrategy diferente seja especificado, não há nenhum efeito, pois o atributo lockStrategy não é suportado para uma substituição de cliente.

Para criar o cliente CompanyGrid usando o arquivo companyGridClientSide.xml, passe o arquivo XML ObjectGrid como uma URL para um dos métodos de conexão no ObjectGridManager.

Criando o cliente para XML

```
ObjectGridManager ogManager =
 ObjectGridManagerFactory.ObjectGridManager();
ClientClusterContext clientClusterContext = ogManager.connect
("MyServer1.company.com:2809", null, new URL(
                "file:xml/companyGridClientSide.xml"));
```

Configurar o Cliente Programaticamente

Também é possível substituir as configurações do ObjectGrid do lado do cliente programaticamente. Crie um objeto ObjectGridConfiguration que seja semelhante em estrutura à instância ObjectGrid do lado do servidor. O código a seguir cria uma instância ObjectGrid do lado do cliente funcionalmente equivalente à substituição do cliente na seção anterior que utiliza um arquivo XML.

Substituição do lado do cliente programaticamente

```
ObjectGridConfiguration companyGridConfig = ObjectGridConfigFactory
    .createObjectGridConfiguration("CompanyGrid");
Plugin txCallbackPlugin = ObjectGridConfigFactory.createPlugin(
    PluginType.TRANSACTION_CALLBACK, "com.company.MyClientTxCallback");
companyGridConfig.addPlugin(txCallbackPlugin);

Plugin ogEventListenerPlugin = ObjectGridConfigFactory.createPlugin(
    PluginType.OBJECTGRID_EVENT_LISTENER, "");
companyGridConfig.addPlugin(ogEventListenerPlugin);

BackingMapConfiguration customerMapConfig = ObjectGridConfigFactory
    .createBackingMapConfiguration("Customer");
customerMapConfig.setNumberOfBuckets(1429);
Plugin evictorPlugin = ObjectGridConfigFactory.createPlugin(PluginType.EVICTOR,
    "com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor");
customerMapConfig.addPlugin(evictorPlugin);

companyGridConfig.addBackingMapConfiguration(customerMapConfig);

BackingMapConfiguration orderLineMapConfig = ObjectGridConfigFactory
    .createBackingMapConfiguration("OrderLine");
orderLineMapConfig.setNumberOfBuckets(701);
orderLineMapConfig.setTimeToLive(800);
orderLineMapConfig.setTtlEvictorType(TTLType.LAST_ACCESS_TIME);

companyGridConfig.addBackingMapConfiguration(orderLineMapConfig);

List ogConfigs = new ArrayList();
ogConfigs.add(companyGridConfig);

Map overrideMap = new HashMap();
overrideMap.put(CatalogServerProperties.DEFAULT_DOMAIN, ogConfigs);

ogManager.setOverrideObjectGridConfigurations(overrideMap);
ClientClusterContext client = ogManager.connect(catalogServerAddresses, null, null);
ObjectGrid companyGrid = ogManager.getObjectGrid(client, objectGridName);
```

A instância ogManager da interface do ObjectGridManager verifica se há substituições apenas nos objetos ObjectGridConfiguration e BackingMapConfiguration que você inclui no Mapa overrideMap. Por exemplo, o código anterior substitui o número de depósitos no mapa OrderLine. Entretanto, o mapa Order permanece inalterado no lado do cliente porque nenhuma configuração para esse mapa é incluída.

Configurar o Cliente na Estrutura Spring

As configurações de ObjectGrid do lado do cliente também podem ser substituídas utilizando a Estrutura Spring. O arquivo XML de exemplo a seguir mostra como construir um elemento ObjectGridConfiguration e utilizá-lo para substituir algumas configurações do lado do cliente. Esse exemplo chama as mesmas APIs que são demonstradas na configuração programática. O exemplo também é uma funcionalidade equivalente ao exemplo na configuração XML ObjectGrid.

configuração do cliente com Spring

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
  "http://www.springframework.org/dtd/spring-beans.dtd">
<beans>
  <bean id="companyGrid" factory-bean="manager" factory-method="getObjectGrid"
    singleton="true">
    <constructor-arg type="com.ibm.websphere.objectgrid.ClientClusterContext">
      <ref bean="client" />
    </constructor-arg>
    <constructor-arg type="java.lang.String" value="CompanyGrid" />
</bean>

  <bean id="manager" class="com.ibm.websphere.objectgrid.ObjectGridManagerFactory"
    factory-method="getObjectGridManager" singleton="true">
    <property name="overrideObjectGridConfigurations">
      <map>
        <entry key="DefaultDomain">
          <list>
            <ref bean="ogConfig" />
          </list>
        </entry>
      </map>
    </property>
  </bean>

  <bean id="ogConfig"
    class="com.ibm.websphere.objectgrid.config.ObjectGridConfigFactory"
    factory-method="createObjectGridConfiguration">
    <constructor-arg type="java.lang.String">
      <value>CompanyGrid</value>
    </constructor-arg>
    <property name="plugins">
      <list>
   <bean class="com.ibm.websphere.objectgrid.config.ObjectGridConfigFactory"
         factory-method="createPlugin">
         <constructor-arg type="com.ibm.websphere.objectgrid.config.PluginType"
           value="TRANSACTION_CALLBACK" />
         <constructor-arg type="java.lang.String"
           value="com.company.MyClientTxCallback" />
       </bean>
       <bean class="com.ibm.websphere.objectgrid.config.ObjectGridConfigFactory"
         factory-method="createPlugin">
         <constructor-arg type="com.ibm.websphere.objectgrid.config.PluginType"
           value="OBJECTGRID_EVENT_LISTENER" />
         <constructor-arg type="java.lang.String" value="" />
       </bean>
      </list>
 </property>
      <property name="backingMapConfigurations">
        <list>
     <bean class="com.ibm.websphere.objectgrid.config.ObjectGridConfigFactory"
           factory-method="createBackingMapConfiguration">
           <constructor-arg type="java.lang.String" value="Customer" />
             <property name="plugins">
      <bean class="com.ibm.websphere.objectgrid.config.ObjectGridConfigFactory"
                 factory-method="createPlugin">
                 <constructor-arg type="com.ibm.websphere.objectgrid.config.PluginType"
     value="EVICTOR" />
   <constructor-arg type="java.lang.String"
                   value="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor" />
              </bean>
            </property>
            <property name="numberOfBuckets" value="1429" />
          </bean>
          <bean class="com.ibm.websphere.objectgrid.config.ObjectGridConfigFactory"
            factory-method="createBackingMapConfiguration">
```

```
              <constructor-arg type="java.lang.String" value="OrderLine" />
                <property name="numberOfBuckets" value="701" />
   <property name="timeToLive" value="800" />
   <property name="ttlEvictorType">
                  <value type="com.ibm.websphere.objectgrid.
          TTLType">LAST_ACCESS_TIME</value>
   </property>
            </bean>
          </list>
        </property>
      </bean>

      <bean id="client" factory-bean="manager" factory-method="connect"
        singleton="true">
        <constructor-arg type="java.lang.String">
     <value>localhost:2809</value>
        </constructor-arg>
   <constructor-arg
          type="com.ibm.websphere.objectgrid.security.
      config.ClientSecurityConfiguration">
          <null />
        </constructor-arg>
   <constructor-arg type="java.net.URL">
     <null />
        </constructor-arg>
      </bean>
</beans>
```

Depois de criar o arquivo XML, carregue o arquivo e crie o ObjectGrid com o seguinte fragmento de código.

```
BeanFactory beanFactory = new XmlBeanFactory(new

UrlResource("file:test/companyGridSpring.xml"));

ObjectGrid companyGrid = (ObjectGrid) beanFactory.getBean("companyGrid");
```

Leia sobre a Visão geral de integração de estrutura spring para obter mais informações sobre como criar um arquivo descritor XML.

Desativar o Cache Local do Cliente

O cache local é ativado por padrão quando um bloqueio é configurado como otimista ou nenhum. Os clientes não mantêm um cache local quando a configuração do bloqueio é configurada como pessimista. Para desativar o cache local, configure o atributo numberOfBuckets para 0 no arquivo descritor do ObjectGrid de substituição do cliente.

Ativando o Mecanismo de Invalidação do Cliente

Em um ambiente do WebSphere eXtreme Scale distribuído, o lado do cliente possui um cache local por padrão ao utilizar a estratégia de bloqueio otimista ou quando o bloqueio está desativado. O cache perto tem seus próprios dados locais armazenados em cache. Se um eXtreme Scale cliente confirmar uma atualização, a atualização chegará próxima do cache e do servidor. Entretanto, outros eXtreme Scale clientes não recebem as informações de atualização e poderão ter dados desatualizados.

Cache Local

Os aplicativos devem ficar cientes desse problema de dados obsoletos no cliente do eXtreme Scale. É possível usar a classe ObjectGridEventListener baseada em Java Message Service (JMS) integrado, com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener, para

ativar o mecanismo de invalidação do cliente dentro de um ambiente do eXtreme Scale distribuído que é conhecido como uma grade eXtreme Scale.

O mecanismo de invalidação do cliente é a solução para o problema de dados antigos no cache local do cliente no ambiente do eXtreme Scale distribuído. Esse mecanismo garante que o cache local ao cliente esteja em sincronia com os servidores ou outros clientes. Contudo, mesmo com esse mecanismo de invalidação do cliente baseado em JMS, o cache local do cliente não é atualizado imediatamente. Um atraso ocorre quando o tempo de execução do eXtreme Scale publica atualizações.

Dois modelos estão disponíveis para o mecanismo de invalidação do cliente em um ambiente do eXtreme Scale distribuído:

- Modelo cliente-servidor: Neste modelo, todos os processos do servidor assumem a função de publicador e publicam todas as alterações na transação para o destino do JMS designado. Todos os processos do cliente estão nas funções de receptor e recebem todas as alterações na transação do destino do JMS designado.
- Modelo cliente como dupla função: Neste modelo, os processos do servidor não têm nada a fazer com o destino do JMS. Todos os processos do cliente assumem ambas as funções do JMS, publicador e receptor. As alterações na transação que ocorrem no cliente são publicadas no destino do JMS e todos os clientes recebem tais alterações.

Para obter mais informações, leia sobre "Listener de Eventos da JMS" na página 138.

Modelo de Cliente-servidor

Em um modelo cliente/servidor, os servidores exercem a função de publicador JMS e o cliente a função de receptor JMS.

Exemplo de XML de modelo cliente/servidor

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
  xmlns="http://ibm.com/ws/objectgrid/config">
  <objectGrids>
    <objectGrid name="AgentObjectGrid">
      <bean id="ObjectGridEventListener"
        className="com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener">
        <property name="invalidationModel" type="java.lang.String"
 value="CLIENT_SERVER_MODEL" description="" />
        <property name="invalidationStrategy" type="java.lang.String" value="PUSH" description="" />
        <property name="mapsToPublish" type="java.lang.String" value="agent;profile;pessimisticMap" description="" />
        <property name="jms_topicConnectionFactoryJndiName" type="java.lang.String" value="defaultTCF" description="" />
        <property name="jms_topicJndiName" type="java.lang.String" value="defaultTopic" description="" />
        <property name="jms_topicName" type="java.lang.String" value="defaultTopic" description="" />
        <property name="jms_userid" type="java.lang.String" value="" description="" />
        <property name="jms_password" type="java.lang.String" value="" description="" />
        <property name="jndi_properties" type="java.lang.String"
          value="java.naming.factory.initial=org.apache.activemq.jndi.ActiveMQInitialContextFactory;
    java.naming.provider.url=
   tcp://localhost:61616;connectionFactoryNames=defaultTCF;topic.defaultTopic=defaultTopic"
          description="jndi properties" />
      </bean>

      <backingMap name="agent" readOnly="false" pluginCollectionRef="agent" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="28800" />
      <backingMap name="profile" readOnly="false" pluginCollectionRef="profile" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
      <backingMap name="pessimisticMap" readOnly="false" pluginCollectionRef="pessimisticMap" preloadMode="false"
        lockStrategy="PESSIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
      <backingMap name="excludedMap1" readOnly="false" pluginCollectionRef="excludedMap1" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
      <backingMap name="excludedMap2" readOnly="false" pluginCollectionRef="excludedMap2" preloadMode="false"
```

```
      lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
      timeToLive="2700" />
  </objectGrid>
</objectGrids>

<backingMapPluginCollections>
  <backingMapPluginCollection id="agent">
    <bean id="ObjectTransformer" className="com.ibm.ws.objectgrid.test.scenario.AgentObjectTransformer" />
  </backingMapPluginCollection>
  <backingMapPluginCollection id="profile">
    <bean id="ObjectTransformer" className="com.ibm.ws.objectgrid.test.scenario.ProfileObjectTransformer" />
    <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      <property name="maxSize" type="int" value="2000" description="set max size for LRU evictor" />
      <property name="sleepTime" type="int" value="15" description="evictor thread sleep time" />
      <property name="numberOfLRUQueues" type="int" value="50" description="set number of LRU queues" />
    </bean>
  </backingMapPluginCollection>

  <backingMapPluginCollection id="pessimisticMap" />
  <backingMapPluginCollection id="excludedMap1" />
  <backingMapPluginCollection id="excludedMap2" />
</backingMapPluginCollections>

</objectGridConfig>
```

Cliente como Modelo de Funções Duplas

No modelo cliente como dupla função, cada cliente assume ambas as funções do JMS, publicador e receptor. O cliente publica cada mudança da transação confirmada para um destino JMS designado e recebe todas as alterações transacionais confirmadas de outros clientes. O servidor não tem nada a fazer com o JMS neste modelo.

Exemplo de XML de modelo de função dupla

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
  xmlns="http://ibm.com/ws/objectgrid/config">
  <objectGrids>
    <objectGrid name="AgentObjectGrid">
      <bean id="ObjectGridEventListener"
        className="com.ibm.websphere.objectgrid.plugins.builtins.JMSObjectGridEventListener">
        <property name="invalidationModel" type="java.lang.String" value="CLIENT_AS_DUAL_ROLES_MODEL" description="" />
        <property name="invalidationStrategy" type="java.lang.String" value="PUSH" description="" />
        <property name="mapsToPublish" type="java.lang.String" value="agent;profile;pessimisticMap" description="" />
        <property name="jms_topicConnectionFactoryJndiName" type="java.lang.String" value="defaultTCF" description="" />
        <property name="jms_topicJndiName" type="java.lang.String" value="defaultTopic" description="" />
        <property name="jms_topicName" type="java.lang.String" value="defaultTopic" description="" />
        <property name="jms_userid" type="java.lang.String" value="" description="" />
        <property name="jms_password" type="java.lang.String" value="" description="" />
        <property name="jndi_properties" type="java.lang.String"
          value="java.naming.factory.initial=org.apache.activemq.jndi.ActiveMQInitialContextFactory;java.naming.provider.url=
     tcp://localhost:61616;connectionFactoryNames=defaultTCF;topic.defaultTopic=defaultTopic"
          description="jndi properties" />
      </bean>

      <backingMap name="agent" readOnly="false" pluginCollectionRef="agent" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="28800" />
      <backingMap name="profile" readOnly="false" pluginCollectionRef="profile" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
      <backingMap name="pessimisticMap" readOnly="false" pluginCollectionRef="pessimisticMap" preloadMode="false"
        lockStrategy="PESSIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
      <backingMap name="excludedMap1" readOnly="false" pluginCollectionRef="excludedMap1" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
      <backingMap name="excludedMap2" readOnly="false" pluginCollectionRef="excludedMap2" preloadMode="false"
        lockStrategy="OPTIMISTIC" copyMode="COPY_ON_READ_AND_COMMIT" ttlEvictorType="LAST_ACCESS_TIME"
        timeToLive="2700" />
    </objectGrid>
  </objectGrids>

  <backingMapPluginCollections>
    <backingMapPluginCollection id="agent">
      <bean id="ObjectTransformer" className="com.ibm.ws.objectgrid.test.scenario.AgentObjectTransformer" />
    </backingMapPluginCollection>
    <backingMapPluginCollection id="profile">
      <bean id="ObjectTransformer" className="com.ibm.ws.objectgrid.test.scenario.ProfileObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
        <property name="maxSize" type="int" value="2000" description="set max size for LRU evictor" />
        <property name="sleepTime" type="int" value="15" description="evictor thread sleep time" />
        <property name="numberOfLRUQueues" type="int" value="50" description="set number of LRU queues" />
      </bean>
    </backingMapPluginCollection>
```

```
    <backingMapPluginCollection id="pessimisticMap" />
    <backingMapPluginCollection id="excludedMap1" />
    <backingMapPluginCollection id="excludedMap2" />
  </backingMapPluginCollections>

</objectGridConfig>
```

Configuração de Tempo Limite de Nova Tentativa de Solicitação

Com mapas confiáveis, é possível fornecer um tempo limite de nova tentativa para o WebSphere eXtreme Scale para pedidos de transação.

Há duas formas de configura mapas confiáveis. Se o valor for maior que zero, a solicitação é tentada até que a condição de tempo limite seja atendida ou uma falha permanente como uma exceção DuplicateKeyException seja encontrada. Um valor de zero indica que a configuração do modo fail-fast e o eXtreme Scale não tentam novamente.

Você fornece um valor de tempo limite em milissegundos no arquivo de propriedades do cliente ou em uma sessão. A sessão sempre substitui a definição das propriedades do cliente. Durante o tempo de execução, o tempo limite da transação é usado com o tempo limite de nova tentativa, garantindo que o tempo limite de nova tentativa não exceda o tempo limite da transação.

Devido às variações de como são concluídas as transações de auto-consolidação e sem auto-consolidação (transações que estão utilizando métodos begin e commit explícitos), as exceções válidas para a nova tentativa são diferentes.

Para transações que são chamadas dentro de uma sessão, a nova tentativa é válida para SystemExceptions do CORBA e exceções TargetNotAvailable do eXtreme Scale.

Para transações de autoconfirmação, a nova tentativa é válida para SystemExceptions do CORBA e Exceções de Disponibilidade do eXtreme Scale (ReplicationVotedToRollbackTransactionException, TargetNotAvailable, AvailabilityException e outras).

Para obter mais informações, consulte o tópico sobre o uso das sessões para acessar os dados na grade no *Guia de Programação*.

As falhas do aplicativo ou outras falhas permanentes são retornadas imediatamente, e o cliente não tenta a transação novamente. Essas falhas permanentes incluem as exceções DuplicateKeyException e KeyNotFoundException.

A configuração fail-fast retorna todas as exceções sem tentar novamente por nenhuma exceção.

A seguinte lista mostra as exceções em mais detalhes:

As exceções onde o cliente tenta novamente

- ReplicationVotedToRollbackTransactionException (somente na auto-consolidação)
- TargetNotAvailable
- org.omg.CORBA.SystemException
- AvailabilityException (somente na auto-consolidação)

- LockTimeoutException (somente na auto-consolidação)
- UnavailableServiceException (somente na auto-consolidação)

Outras exceções
- DuplicateKeyException
- KeyNotFoundException
- LoaderException
- TransactionAffinityException
- LockDeadlockException
- OptimisticCollisionException

Configuração da Propriedade requestRetryTimeout em um Arquivo de Propriedades

Para configurar o valor requestRetryTimeout em um cliente, inclua ou modifique a propriedade requestRetryTimeout no "Arquivo de Propriedades do Cliente" na página 202. As propriedades do cliente estão no arquivo `objectGridClient.properties` pelo padrão. A propriedade requestRetryTimeout é configurada em milissegundos. Configure o valor maior que zero para que o pedido seja tentado novamente em exceções para as quais uma nova tentativa está disponível. Configure o valor como `0` para falhar sem as novas tentativas em exceções. Para usar o comportamento padrão, remova a propriedade ou configure o valor como `-1`.

objectGridClient.properties

```
# eXtreme Scale client config
preferLocalProcess = false
preferLocalhost = false
requestRetryTimeout = 30000
```

O valor requestRetryTimeout é especificado em milissegundos. No exemplo anterior, se o valor for utilizado em uma instância de ObjectGrid, o valor de requestRetryTimeout será 30 segundos.

Configuração das Propriedades do Cliente pela Obtenção de uma Conexão com ObjectGrid Programaticamente

Para configurar as propriedades do cliente programaticamente, primeiro crie um arquivo de propriedades do cliente em um `<local>` apropriado para seu aplicativo. No exemplo a seguir, o arquivo de propriedades do cliente refere-se ao fragmento de `objectGridClient.properties` na seção anterior. Após estabelecer uma conexão com ObjectGridManager, configure as propriedades do cliente conforme a descrição. Em seguida, quando você tiver uma instância do ObjectGrid, ela terá as propriedades do cliente que você definiu no arquivo. Se você alterar o arquivo de propriedades do cliente, será necessário toda vez obter explicitamente uma nova instância de ObjectGrid.

```
ObjectGridManager manager= ObjectGridManagerFactory.getObjectGridManager();
String objectGridName = "testObjectGrid";
URL clientXML = null;
ClientClusterContext ccc = manager.connect("localhost:2809", null,
clientXML);
File file = new File("<location>/objectGridClient.properties");
URL url = file.toURI().toURL();
ccc.setClientProperties(objectGridName, url);
ObjectGrid objectGrid = ogManager.getObjectGrid(ccc, objectGridName);
```

Exemplo de Substituição de Sessão com Auto-consolidação

Para configurar o tempo limite de nova tentativa de solicitação em uma sessão ou para substituir a propriedade do cliente requestRetryTimeout, chame o método setRequestRetryTimeout(long) na interface Session.

```
Session sessionA = objectGrid.getSession();
sessionA.setRequestRetryTimeout(30000);
ObjectMap mapA = sessionA.getMap("payroll");
String key = "key:" + j;
mapA.insert(key, "valueA");
```

Esta sessão agora usa um valor requestRetryTimeout de 30.000 milissegundos ou de 30 segundos, independente do valor configurado no arquivo de propriedades do cliente. Para obter mais informações sobre a interface de sessão, consulte Usando sessões para acessar dados na grade.

Configurando Entidades

Um ObjectGrid pode ter inúmeros esquemas de entidade lógicos. As entidades são definidas usando classes Java™ anotadas, XML ou uma combinação de XML e classes Java. As entidades definidas são, então, registradas com um servidor eXtreme Scale e ligadas a BackingMaps, índices e outros plug-ins.

Antes de Iniciar

Um esquema de entidade é um conjunto de entidades e relacionamentos entre as entidades. Leia sobre Definindo um esquema de entidade para obter detalhes sobre a definição de esquema e a configuração da entidade.

Gerenciamento de Relacionamentos

Linguagens orientadas a objetos como Java, e relacionamentos ou associações de suporte a bancos de dados relacionais. Os relacionamentos diminuem a quantidade de armazenamento por meio do uso de referências de objetos ou chaves estrangeiras.

Ao usar relacionamentos em uma grade, os dados devem ser organizados em uma árvore limitada. Deve existir um tipo root na árvore e todos os filhos devem estar associados a somente um root. Por exemplo: Departamento pode ter muitos Funcionários e um Funcionário pode ter muitos Projetos. Porém um Projeto não pode ter muitos Funcionários pertencentes a diferentes departamentos. Depois de uma raiz ser definida, todo o acesso a este objeto raiz e seus descendentes será gerenciado por meio da raiz. O WebSphere eXtreme Scale usa o código hash da chave do objeto raiz para escolher uma partição.

Por exemplo: partition = (hashCode MOD numPartitions).

Quando todos os dados para um relacionamento estiverem ligados a um única instância do objeto, toda a árvore pode ser co-localizada em uma única partição e pode ser acessada muito eficientemente usando uma transação. Se os dados englobarem múltiplos relacionamentos, então múltiplas partições devem estar envolvidas que envolvem chamadas remotas adicionais, o que pode levar a gargalos no desempenho.

Dados de Referência

Alguns relacionamentos incluem dados de busca ou referência como: CountryName. Este é um caso especial em que os dados devem existir em cada partição. Aqui, os dados podem ser acessados por qualquer chave root e o mesmo resultado será retornado. Os dados de referência como estes devem ser usados somente em casos onde os dados forem razoavelmente estáticos sendo que a atualização deles pode ser cara, pois necessitam de atualização em cada partição. A API DataGrid é uma técnica comum para manter os dados de referência atualizados.

Custos e Benefícios de Normalização

A normalização dos dados usando os relacionamentos pode ajudar a reduzir a quantidade de memória usada pela grade pois a duplicação de dados é diminuída. Porém, em geral, quanto mais dados relacionais forem incluídos, menos eles irão expandir. Quando os dados são agrupados juntos, torna-se mais caro manter os relacionamentos e manter os tamanhos gerenciáveis. Como os dados das partições da grade baseiam-se na chave da raiz da árvore, o tamanho da árvore não é levado em consideração. Assim, se você tiver uma grande quantidade de relacionamentos para uma instância da árvore, a grade pode ficar desequilibrada, fazendo com que uma partição mantenha mais dados do que as outras.

Quando os dados são não-normalizados ou sequenciais, os dados que seriam normalmente compartilhados entre dois objetos são, em vez disso, duplicados e cada tabela pode ser independentemente particionada, oferecendo uma grade mais balanceada. Apesar disto aumentar a quantidade de memória usada, permite que o aplicativo escale pois uma única linha de dados pode ser acessada que pode ter todos os dados necessários. Isto é ideal para grades com maior quantidade de leituras pois a manutenção dos dados se torna mais cara.

Para obter informações adicionais, consulte Classificação de sistemas XTP e escalamento.

Gerenciamento de Relacionamentos Usando as APIs de Acesso a Dados

A API ObjectMap é a mais rápida, mais flexível e granular das APIs de acesso a dados, oferecendo uma abordagem transacional baseada em sessão no acesso aos dados na grade de mapas. A API ObjectMap permite aos clientes usarem operações CRUD comuns (create, read, update e delete) para gerenciar pares chave-valor de objetos na grade distribuída.

Ao usar a API ObjectMap, os relacionamentos de objetos devem ser expressos pela incorporação da chave estrangeira para todos os relacionamentos no objeto-pai.

A seguir, está um exemplo.

```
public class Department {
 Collection<String> employeeIds;
}
```

A API EntityManager simplifica o gerenciamento de relacionamentos por meio da extração de dados persistentes a partir de objetos incluindo as chaves estrangeiras. Quando o objeto é posteriormente recuperado da grade, o gráfico de relacionamentos é reconstruído, como no exemplo a seguir.

```
@Entity
public class Department {
 Collection<String> employees;
}
```

A API EntityManager é muito semelhante a outras tecnologias de persistência do objeto Java como JPA e Hibernate na qual ela sincroniza um gráfico de instâncias de objetos Java gerenciados com o armazenamento persistente. Neste caso, o armazenamento persistente é uma grade eXtreme Scale, em que cada entidade é representada como um mapa e o mapa contém os dados da entidade em vez das instâncias do objeto.

Arquivo Descritor XML de Metadados de Entidade

O arquivo descritor de metadados da entidade é um arquivo XML que é utilizado para definir um esquema de entidade para o WebSphere eXtreme Scale. Defina todos os metadados de entidade no arquivo XML ou defina os metadados de entidade como anotações no arquivo de classe Java de entidade. O uso principal é para entidades que não podem usar as anotações Java.

Utilize a configuração do XML para criar os metadados de entidade com base no arquivo XML. Quando utilizado em conjunto com a anotação, alguns dos atributos definidos na configuração do XML sobrescrevem as anotações correspondentes. Se for possível substituir um elemento, a substituição estará explicitamente nas seções a seguir. Consulte o “Arquivo `emd.xsd`” na página 227 para obter um exemplo do arquivo descritor XML de metadados da entidade.

Elemento ID

O elemento id implica que o atributo é uma chave. Pelo menos, um elemento id deve ser especificado. É possível especificar várias chaves de id para serem utilizadas como chave composta.

Atributos

name
Especifica o nome do atributo. O atributo deve existir no arquivo Java.

alias
Especifica o alias do elemento. O valor do alias é substituído se utilizado em conjunto com uma entidade anotada.

Elemento basic

O elemento basic implica que o atributo é um tipo primitivo ou wrappers para tipos primitivos:

- java.lang.String
- java.math.BigInteger
- java.math.BigDecimal
- java.util.Date
- java.util.Calendar
- java.sql.Date
- java.sql.Time
- java.sql.Timestamp
- byte[]
- Byte[]

- char[]
- Character[]
- Enum Java Platform, Standard Edition Versão 5

É necessário especificar qualquer atributo como básico. Os atributos element básicos são automaticamente configurados utilizando reflexo.

Elemento id-class

O elemento id_class especifica uma classe de chave composta, que ajuda a localizar entidades com chaves compostas.

Atributos

class-name
Especifica o nome da classe, que é um id-class, para uso com o elemento id-class.

transient

O elemento transient implica que ele é ignorado e não processado. Ele também pode ser substituído se utilizado em conjunto com entidades anotadas.

Atributos

name
Especifica o nome do atributo, que é ignorado.

versão

O elemento transient implica que ele é ignorado e não processado. Ele também pode ser substituído se utilizado em conjunto com entidades anotadas.

Atributos

name
Especifica o nome do atributo, que é ignorado.

Elemento property

Utilize o elemento property para incluir propriedades nos plug-ins. O nome da propriedade deve corresponder a um método configurado na classe referenciada pelo bean de conteúdo.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

name
Especifica o nome da propriedade. O valor atribuído a esse atributo deve corresponder a um método set na classe que é fornecida como atributo className no bean que contém o atributo. Por exemplo, se você configurar o atributo className do bean como `com.ibm.MyPlugin` e o nome da propriedade que é fornecido for `size`, a classe `com.ibm.MyPlugin` deve ter um método setSize. (Necessário)

type
Especifica o tipo da propriedade. O tipo é passado para o método configurado que é identificado pelo atributo name. Os valores válidos são os primitivos Java, os correspondentes java.lang e o java.lang.String. Os atributos name e type devem corresponder a uma assinatura de método no atributo className do bean. Por exemplo, se você configurar o nome como `size` e o tipo como `int`, um método setSize(int) deve existir na classe que é especificada como o atributo className para o bean. (Necessário)

value
Especifica o valor da propriedade. Este valor é convertido no tipo especificado pelo atributo type e, em seguida, é utilizado como um parâmetro na chamada para o método set identificado pelos atributos name e type. O valor deste atributo não é validado de nenhuma maneira. (Necessário)

description
Descreve a propriedade. (Opcional)

```
<bean
(1)   name="name"
(2)   type="java.lang.String" | "boolean" | "java.lang.Boolean" | "int" |
       "java.lang.Integer" | "double" | "java.lang.Double" | "byte" |
       "java.lang.Byte" | "short" | "java.lang.Short" | "long" |
       "java.lang.Long" | "float" | "java.lang.Float" | "char" |
       "java.lang.Character"
(3)   value="value"
(4)   description="description"
/>
```

No exemplo a seguir, o arquivo `companyGridProperty.xml` é utilizado para demonstrar como incluir um elemento property em um bean. Neste exemplo, uma propriedade com o nome maxSize e o tipo int são incluídos em um evictor. O Evictor com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor tem uma assinatura de método que corresponde ao método setMaxSize(int). Um valor de número inteiro de 499 é passado para o método setMaxSize(int) na classe com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Customer"
    pluginCollectionRef="customerPlugins"/>
  </objectGrid>
 </objectGrids>
 <backingMapPluginCollections>
  <backingMapPluginCollection id="customerPlugins">
   <bean id="Evictor"
     className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor>
     <property name="maxSize" type="int" value="449"
       description="The maximum size of the LRU Evictor"/>
   </bean>
  </backingMapPluginCollection>
 </backingMapPluginCollections>
</objectGridConfig>
```

A amostra de código a seguir demonstra a abordagem programática para obter a mesma configuração do arquivo `companyGridProperty.xml` no exemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
```

```
BackingMap customerMap = companyGrid.defineMap("Customer");

LRUEvictor lruEvictor = new com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor();
// if the XML file is used instead,
// the property that was added would cause the following call to occur
lruEvictor.setMaxSize(449);
customerMap.setEvictor(lruEvictor);
```

Elemento backingMapPluginsCollections

O elemento backingMapPluginsCollections é um contêiner para todos os elementos backingMapPluginCollection. No arquivo `companyGridProperty.xml` na seção anterior, o elemento backingMapPluginCollections contém um elemento backingMapPluginCollection com o ID `customerPlugins`.

- Número de ocorrências: Zero para uma
- Elemento-filho: Elemento backingMapPluginCollection

Elemento backingMapPluginCollection

O elemento backingMapPluginCollection define os plug-ins do BackingMap e é identificado pelo atributo **id**. Especifique o atributo pluginCollectionRef para referenciar os plug-ins. Ao configurar vários plug-ins do BackingMaps de maneira semelhante, cada BackingMap pode referenciar o mesmo elemento backingMapPluginCollection.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Elemento bean

Atributos

id Identifica a backingMapPluginCollection e é referenciado pelo atributo pluginCollectionRef do elemento backingMap. Cada ID deve ser exclusivo. Se o valor de um atributo pluginCollectionRef não corresponder ao ID de um elemento backingMapPluginCollection, a validação XML falha. Qualquer número de elementos backingMap pode fazer referência a um único elemento backingMapPluginCollection. (Necessário)

```
<backingMapPluginCollection
(1)   id="id"
/>
```

No exemplo a seguir, o arquivo `companyGridCollection.xml` é utilizado para demonstrar como utilizar o elemento backingMapPluginCollection. Neste arquivo, o BackingMap do Cliente utilizar o customerPlugins backingMapPluginCollection para configurar o BackingMap do Cliente com um LRUEvictor. O Item e OrderLine BackingMaps referenciam o collection2 backingMapPluginCollection. Cada um destes BackingMaps possuem um conjunto de LFUEvictor.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Customer"
    pluginCollectionRef="customerPlugins"/>
   <backingMap name="Item" pluginCollectionRef="collection2"/>
   <backingMap name="OrderLine"
    pluginCollectionRef="collection2"/>
   <backingMap name="Order"/>
  </objectGrid>
 </objectGrids>
 <backingMapPluginCollections>
  <backingMapPluginCollection id="customerPlugins">
   <bean id="Evictor"
     className="com.ibm.websphere.objectGrid.plugins.builtins.LRUEvictor/>
  </backingMapPluginCollection>
```

```
  <backingMapPluginCollection id="collection2">
   <bean id="Evictor"
     className="com.ibm.websphere.objectgrid.plugins.builtins.LFUEvictor/>
   <bean id="OptimisticCallback"
     className="com.ibm.websphere.samples.objectgrid.EmployeeOptimisticCallBackImpl"/>
  </backingMapPluginCollection>
  </backingMapPluginCollections>
</objectGridConfig>
```

A amostra de código a seguir demonstra a abordagem programática para obter a mesma configuração que o arquivo `companyGridCollection.xml` no exemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();

ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
BackingMap customerMap = companyGrid.defineMap("Customer");
LRUEvictor customerEvictor = new LRUEvictor();
customerMap.setEvictor(customerEvictor);

BackingMap itemMap = companyGrid.defineMap("Item");
LFUEvictor itemEvictor = new LFUEvictor();
itemMap.setEvictor(itemEvictor);

BackingMap orderLineMap = companyGrid.defineMap("OrderLine");
LFUEvictor orderLineEvictor = new LFUEvictor();
orderLineMap.setEvictor(orderLineEvictor);

BackingMap orderMap = companyGrid.defineMap("Order");
```

Elemento querySchema

O elemento querySchema define relacionamentos entre BackingMaps e identifica o tipo de objeto em cada mapa. Estas informações são utilizadas pelo ObjectQuery para converter cadeias de linguagem de consulta em chamadas de acesso de mapa. Para obter mais informações, consulte os detalhes sobre a definição de um esquema ObjectQuery no *Guia de Programação*.

- Número de ocorrências: Zero para uma
- Elemento-filho: Elemento mapSchemas, elemento relationships

Elemento mapSchemas

Cada elemento querySchema tem um elemento mapSchemas que contém um ou mais elementos mapSchema.

- Número de ocorrências: Uma
- Elemento-filho: Elemento mapSchema

Elemento mapSchema

Um elemento mapSchema define o tipo de objeto que é armazenado em um BackingMap e instruções sobre como acessar os dados.

- Número de ocorrências: Uma ou mais
- Elemento-filho: Nenhum

Atributos

mapName
Especifica o nome do BackingMap a incluir no esquema. (Necessário)

valueClass
Especifica o tipo de objeto que é armazenado na parte do valor do BackingMap. (Necessário)

primaryKeyField
Especifica o nome do atributo-chave principal no atributo valueClass. A chave primária também deve ser armazenada na parte da chave do BackingMap. (Opcional)

accessType
Identifica como o mecanismo de consulta examina e acessa os dados persistentes nas instâncias do objeto valueClass. Se você configurar o valor como FIELD, os campos de classe são examinados e incluídos no esquema. Se o valor for PROPERTY, os atributos que estão associados com os métodos get e is são utilizados. O valor padrão é PROPERTY. (Opcional)

```
<mapSchema
(1)   mapName="backingMapName"
(2)   valueClass="com.mycompany.OrderBean"
(3)   primaryKeyField="orderId"
(4)   accessType="PROPERTY" | "FIELD"
/>
```

No exemplo a seguir, o arquivo companyGridQuerySchemaAttr.xml é utilizado para demonstrar uma configuração do mapSchema de amostra.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Order"/>
   <backingMap name="Customer"/>

   <querySchema>
    <mapSchemas>
     <mapSchema mapName="Order"
      valueClass="com.mycompany.OrderBean"
      primaryKeyField="orderNumber"
      accessType="FIELD"/>
     <mapSchema mapName="Customer"
      valueClass="com.mycompany.CustomerBean"
      primaryKeyField="id"
      accessType="FIELD"/>
    </mapSchemas>
   </querySchema>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

A amostra de código a seguir demonstra a abordagem programática para obter a mesma configuração do arquivo companyGridQuerySchemaAttr.xml no exemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
companyGrid.defineMap("Order");
companyGrid.defineMap("Customer");

// Define the schema
QueryConfig queryCfg = new QueryConfig();
queryCfg.addQueryMapping(new QueryMapping(
    "Order", OrderBean.class.getName(), "orderNumber", QueryMapping.FIELD_ACCESS));
queryCfg.addQueryMapping(new QueryMapping(
    "Customer", CustomerBean.class.getName(), "id", QueryMapping.FIELD_ACCESS));
companyGrid.setQueryConfig(queryCfg);
```

Elemento relationships

Cada elemento querySchema tem zero ou um elemento relationships que contém um ou mais elementos relationship.

- Número de ocorrências: Zero ou uma
- Elemento-filho: Elemento relationship

Elemento relationship

Um elemento relationship define o relacionamento entre dois BackingMaps e os atributos no atributo valueClass que liga o relacionamento.

- Número de ocorrências: Uma ou mais
- Elemento-filho: Nenhum

Atributos

source
Especifica o nome da valueClass do lado da origem de um relacionamento. (Necessário)

target
Especifica o nome da valueClass do lado do destino de um relacionamento. (Necessário)

relationField
Especifica o nome do atributo na valueClass de origem que faz referência ao destino. (Necessário)

invRelationField
Especifica o nome do atributo na valueClass de destino que faz referência à origem. Se este atributo não for especificado, o relacionamento é unidirecional. (Opcional)

```
<mapSchema
(1)   source="com.mycompany.OrderBean"
(2)   target="com.mycompany.CustomerBean"
(3)   relationField="customer"
(4)   invRelationField="orders"
/>
```

No exemplo a seguir, o arquivo `companyGridQuerySchemaWithRelationshipAttr.xml` é utilizado para demonstrar uma configuração de mapSchema de amostra que inclui um relacionamento bidirecional.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">

 <objectGrids>
  <objectGrid name="CompanyGrid">
   <backingMap name="Order"/>
   <backingMap name="Customer"/>

   <querySchema>
    <mapSchemas>
     <mapSchema mapName="Order"
      valueClass="com.mycompany.OrderBean"
      primaryKeyField="orderNumber"
      accessType="FIELD"/>
     <mapSchema mapName="Customer"
      valueClass="com.mycompany.CustomerBean"
      primaryKeyField="id"
      accessType="FIELD"/>
    </mapSchemas>
    <relationships>
```

```
     <relationship
      source="com.mycompany.OrderBean"
      target="com.mycompany.CustomerBean"
      relationField="customer"/>
      invRelationField="orders"/>
    </relationships>
   </querySchema>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

A amostra de código a seguir demonstra a abordagem programática para obter a mesma configuração que o arquivo `companyGridQuerySchemaWithRelationshipAttr.xml` no exemplo anterior.

```
ObjectGridManager objectGridManager = ObjectGridManagerFactory.getObjectGridManager();
ObjectGrid companyGrid = objectGridManager.createObjectGrid("CompanyGrid", false);
companyGrid.defineMap("Order");
companyGrid.defineMap("Customer");

// Define the schema
QueryConfig queryCfg = new QueryConfig();
queryCfg.addQueryMapping(new QueryMapping(
    "Order", OrderBean.class.getName(), "orderNumber", QueryMapping.FIELD_ACCESS));
queryCfg.addQueryMapping(new QueryMapping(
    "Customer", CustomerBean.class.getName(), "id", QueryMapping.FIELD_ACCESS));
queryCfg.addQueryRelationship(new QueryRelationship(
     OrderBean.class.getName(), CustomerBean.class.getName(), "customer", "orders"));
companyGrid.setQueryConfig(queryCfg);
```

Elemento timeBasedDBUpdate

Um elemento timeBasedDBUpdate define uma configuração para um atualizador de banco de dados baseado em tempo. Um elemento timeBasedDBUpdate contém informações sobre com que frequência os registros inseridos e atualizados recentemente são recuperados a partir do banco de dados usando o Java Persistence API (JPA) e sobre como atualizar os dados nos mapas ObjectGrid correspondentes.

- Número de ocorrências: Zero ou uma
- Elemento-filho: Nenhum

Atributos

entityClass
Especifica o nome da classe de entidade utilizado para interagir com o provedor JPA. O nome da classe de entidade é utilizado para recuperar entidades JPA utilizando consultas de entidade. (Necessário)

persistenceUnitName
Especifica o nome da unidade de persistência JPA para criação de um factory do gerenciador de entidades JPA. O valor padrão é o nome da primeira unidade de persistência definida no arquivo `persistence.xml`. (Opcional)

mode
Especifica o modo de atualização de banco de dados baseado em tempo. Por padrão, o modo de atualização de banco de dados baseado em tempo é configurado como INVALIDATE_ONLY. Um tipo INVALIDATE_ONLY indica a invalidação das entradas no mapa do ObjectGrid se os registros correspondentes no banco de dados foram alterados. Um tipo UPDATE_ONLY indica a atualização das entradas existentes no mapa do ObjectGrid com os valores mais recentes do banco de dados. Entretanto, todos os registros recentemente inseridos no banco de dados são ignorados. Um tipo INSERT_UPDATE indica a atualização das entradas existentes no mapa do

ObjectGrid com os valores mais recentes do banco de dados. Além disso, todos os registros recentemente inseridos no banco de dados são inseridos no mapa do ObjectGrid. (Opcional)

timestampField
Especifica o nome do campo do registro de data e hora. Um valor de campo de registro de data e hora é utilizado para identificar a hora ou sequência quando um registro de backend de banco de dados foi utilizado pela última vez. (Opcional)

jpaPropertyFactory
Identifica o nome da classe de implementação do JPAPropertyFactory ou Spring bean. A interface com.ibm.websphere.objectgrid.jpa.JPAPropertyFactory é utilizada para conectar a mapa da propriedade de persistência para substituir as propriedades JPA padrão. Utilize os beans de estrutura spring se for necessário configurar atributos adicionais na instância do JPAPropertyFactory. Consulte Visão Geral da Integração do Spring Framework para obter mais informações. (Opcional)

```
<timeBasedDBUpdate
(1)   persistenceUnitName="SamplePU"
(2)   mode="INVALIDATE_ONLY" | "UPDATE_ONLY" | "INSERT_UPDATE"
(3)   timestampField="TIMESTAMP"
(4)   entityClass="entity class"
(5)   jpaPropertyFactory="JPA property factory class" | "{spring}bean name"
/>
```

Elemento streamQuerySet

O elemento streamQuerySet é o elemento de nível superior para definir um conjunto de consultas de fluxo.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Elemento stream, elemento view

Elemento stream

O elemento stream representa um fluxo para o mecanismo de consulta do fluxo. Cada atributo do elemento stream corresponde a um método da interface StreamMetadata.

- Número de ocorrências: Uma para muitas
- Elemento-filho: Elemento basic

Atributos

name
Especifica o nome do fluxo. A validação falha se este atributo não for especificado. (Necessário)

valueClass
Especifica o tipo de classe do valor que é armazenado no ObjectMap do fluxo. O tipo de classe é utilizado para converter o objeto para os eventos de fluxo e para gerar uma instrução SQL se a instrução não for fornecida. (Necessário)

sql
Especifica a instrução SQL do fluxo. Se essa propriedade não for fornecida, um SQL de fluxo será gerado, refletindo os atributos ou métodos do acessador no atributo valueClass ou utilizando os atributos tuple do metadados da entidade. (Opcional)

access
Especifica o tipo para acessar os atributos da classe de valor. Se você configurar o valor para FIELD, os atributos serão recuperados diretamente a partir dos campos usando o reflexo Java. Caso contrário, os métodos do acessador serão utilizados para ler os atributos. O valor padrão é PROPERTY. (Opcional)

```
<stream
(1)   name="streamName"
(2)   valueClass="streamMapClassType"
(3)   sql="streamSQL create stream stockQuote
       keyed by t ( transactionvolume INTEGER, price DECIMAL (9,2), issue VARCHAR(100) );"
(4)   access="PROPERTY" | "FIELD"
/>
```

Elemento view

O elemento view representa uma visualização de consulta de fluxo. Cada elemento stream corresponde a um método na interface ViewMetadata.

- Número de ocorrências: Uma para muitas
- Elemento-filho: Elemento basic, elemento id

Atributos

name
Especifica o nome da visualização. A validação falha se este atributo não for especificado. (Necessário)

sql
Especifica o SQL do fluxo, que define a transformação da visualização. A validação falha se este atributo não for especificado. (Necessário)

valueClass
Especifica o tipo de classe do valor que é armazenado nesta visualização do ObjectMap. O tipo de classe é utilizado para converter eventos de visualização no formato de tupla correto que é compatível com este tipo de classe. Se o tipo de classe não for fornecido, um formato padrão após as definições da coluna em SPTSQL (Stream Processing Technology Structured Query Language) será utilizado. Se um metadado de entidade é definido para este mapa de visualização, este atributo não deve ser utilizado. O metadado da entidade é utilizado em seu lugar. (Opcional)

access
Especifica o tipo para acessar os atributos da classe de valor. Se você configurar o tipo de acesso para FIELD, os valores de coluna serão configurados diretamente para os campos usando o reflexo Java. Caso contrário, os métodos do acessador são utilizados para configurar os atributos. O valor padrão é PROPERTY. (Opcional)

```
<view
(1)   name="viewName"
(2)   valueClass="viewMapValueClass"
(3)   sql="viewSQL CREATE VIEW last5MinuteAvgPrice AS
      SELECT issue, avg(price) as totalVolume
        FROM (SELECT * FROM stockQuote FETCH LATEST 5 MINUTES) group by issue;"/>
(4)   access="PROPERTY" | "FIELD"
/>
```

Elemento basic

O elemento basic é utilizado para definir um mapeamento a partir do nome do atributo na classe de valor ou metadados da entidade para a coluna que é definida no SPTSQL.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

```
<basic
(1)   name="attributeName"
(2)   column="columnName"
/>
```

Elemento ID

O elemento id é utilizado para um mapeamento de atributo-chave.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

```
<id
(1)   name="idName"
(2)   column="columnName"
/>
```

No exemplo a seguir, o arquivo `StreamQueryApp2.xml` é utilizado para demonstrar como configurar os atributos de um streamQuerySet. O conjunto de consultas de fluxo _stockQuoteSQS_ possui um fluxo e uma visualização. O fluxo e a visualização definem o nome, valueClass, sql e o tipo de acesso respectivamente. O fluxo também define um elemento básico, que especifica que o atributo volume na classe StockQuote é mapeado para o transactionvolume da coluna SQL que é definida na instrução SQL.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
 <objectGrids>
  <objectGrid name="og1">
   <backingMap name="stockQuote" readOnly="false" copyKey="true" streamRef="stockQuote"/>
   <backingMap name="last5MinuteAvgPrice" readOnly="false" copyKey="false" viewRef="last5MinuteAvgPrice"/>

   <streamQuerySet name="stockQuoteSQS">
    <stream
     name="stockQuote"
     valueClass="com.ibm.ws.objectgrid.streamquery.sample.guide.StockQuote"
     sql="create stream stockQuote
         keyed by t ( transactionvolume INTEGER, price DECIMAL (9,2), issue VARCHAR(100) );"
     access="FIELD">
     <basic name="volume" column="transactionvolume"/>
    </stream>

    <view
     name="last5MinuteAvgPrice"
     valueClass="com.ibm.ws.objectgrid.streamquery.sample.guide.AveragePrice"
     sql="CREATE VIEW last5MinuteAvgPrice AS SELECT issue, avg(price) as avgPrice
       FROM (SELECT * FROM stockQuote FETCH LATEST 5 MINUTES) group by issue;"
     access="FIELD"
    </view>
   </streamQuerySet>
  </objectGrid>
 </objectGrids>
</objectGridConfig>
```

Arquivo emd.xsd

Use a definição de esquema XML de metadados de entidade para criar um arquivo descritor XML e definir um esquema de entidade para o WebSphere eXtreme Scale.

Consulte “Arquivo Descritor XML de Metadados de Entidade” na página 216 para obter descrições de cada elemento e atributo do arquivo emd.xsd.

Arquivo emd.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:emd="http://ibm.com/ws/projector/config/emd"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://ibm.com/ws/projector/config/emd"
    elementFormDefault="qualified" attributeFormDefault="unqualified"
    version="1.0">

    <!-- ************************************************** -->
    <xsd:element name="entity-mappings">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="description" type="xsd:string" minOccurs="0"/>
                <xsd:element name="entity" type="emd:entity" minOccurs="1" maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:unique name="uniqueEntityClassName">
            <xsd:selector xpath="emd:entity" />
            <xsd:field xpath="@class-name"/>
        </xsd:unique>
    </xsd:element>

    <!-- ************************************************** -->
    <xsd:complexType name="entity">
        <xsd:sequence>
            <xsd:element name="description" type="xsd:string" minOccurs="0"/>
            <xsd:element name="id-class" type="emd:id-class" minOccurs="0"/>
            <xsd:element name="attributes" type="emd:attributes" minOccurs="0"/>
            <xsd:element name="entity-listeners" type="emd:entity-listeners" minOccurs="0"/>
            <xsd:element name="pre-persist" type="emd:pre-persist" minOccurs="0"/>
            <xsd:element name="post-persist" type="emd:post-persist" minOccurs="0"/>
            <xsd:element name="pre-remove" type="emd:pre-remove" minOccurs="0"/>
            <xsd:element name="post-remove" type="emd:post-remove" minOccurs="0"/>
            <xsd:element name="pre-invalidate" type="emd:pre-invalidate" minOccurs="0"/>
            <xsd:element name="post-invalidate" type="emd:post-invalidate" minOccurs="0"/>
            <xsd:element name="pre-update" type="emd:pre-update" minOccurs="0"/>
            <xsd:element name="post-update" type="emd:post-update" minOccurs="0"/>
            <xsd:element name="post-load" type="emd:post-load" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="name" type="xsd:string" use="required" />
        <xsd:attribute name="class-name" type="xsd:string" use="required"/>
        <xsd:attribute name="access" type="emd:access-type"/>
        <xsd:attribute name="schemaRoot" type="xsd:boolean"/>
    </xsd:complexType>

    <!-- ************************************************** -->
    <xsd:complexType name="attributes">
        <xsd:sequence>
            <xsd:choice>
                <xsd:element name="id" type="emd:id" minOccurs="0" maxOccurs="unbounded"/>
            </xsd:choice>
            <xsd:element name="basic" type="emd:basic" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="version" type="emd:version" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="many-to-one" type="emd:many-to-one" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="one-to-many" type="emd:one-to-many" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="one-to-one" type="emd:one-to-one" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="many-to-many" type="emd:many-to-many" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="transient" type="emd:transient" minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexType>

    <!-- ************************************************** -->
    <xsd:simpleType name="access-type">
        <xsd:restriction base="xsd:token">
            <xsd:enumeration value="PROPERTY"/>
            <xsd:enumeration value="FIELD"/>
        </xsd:restriction>
    </xsd:simpleType>

    <!-- ************************************************** -->
    <xsd:complexType name="id-class">
        <xsd:attribute name="class-name" type="xsd:string" use="required"/>
    </xsd:complexType>

    <!-- ************************************************** -->
    <xsd:complexType name="id">
        <xsd:attribute name="name" type="xsd:string" use="required" />
        <xsd:attribute name="type" type="xsd:string" />
```

```
    <xsd:attribute name="alias" type="xsd:string" use="optional"/>
</xsd:complexType>

<!-- ************************************************** -->
<xsd:complexType name="transient">
    <xsd:attribute name="name" type="xsd:string" use="required" />
</xsd:complexType>

<!-- ************************************************** -->
<xsd:complexType name="basic">
    <xsd:attribute name="name" type="xsd:string" use="required" />
    <xsd:attribute name="alias" type="xsd:string"/>
    <xsd:attribute name="type" type="xsd:string" />
    <xsd:attribute name="fetch" type="emd:fetch-type"/>
</xsd:complexType>

<!-- ************************************************** -->
<xsd:simpleType name="fetch-type">
    <xsd:restriction base="xsd:token">
        <xsd:enumeration value="LAZY"/>
        <xsd:enumeration value="EAGER"/>
    </xsd:restriction>
</xsd:simpleType>

<!-- ************************************************** -->
<xsd:complexType name="many-to-one">
    <xsd:sequence>
        <xsd:element name="cascade" type="emd:cascade-type" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="required" />
    <xsd:attribute name="alias" type="xsd:string"/>
    <xsd:attribute name="target-entity" type="xsd:string"/>
    <xsd:attribute name="fetch" type="emd:fetch-type"/>
    <xsd:attribute name="id" type="xsd:boolean"/>
</xsd:complexType>
<!-- ************************************************** -->
<xsd:complexType name="one-to-one">
    <xsd:sequence>
        <xsd:element name="cascade" type="emd:cascade-type" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="required" />
    <xsd:attribute name="alias" type="xsd:string"/>
    <xsd:attribute name="target-entity" type="xsd:string"/>
    <xsd:attribute name="fetch" type="emd:fetch-type"/>
    <xsd:attribute name="mapped-by" type="xsd:string"/>
    <xsd:attribute name="id" type="xsd:boolean"/>
</xsd:complexType>
<!-- ************************************************** -->
<xsd:complexType name="one-to-many">
    <xsd:sequence>
        <xsd:element name="order-by" type="emd:order-by" minOccurs="0"/>
        <xsd:element name="cascade" type="emd:cascade-type" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="required" />
    <xsd:attribute name="alias" type="xsd:string"/>
    <xsd:attribute name="target-entity" type="xsd:string"/>
    <xsd:attribute name="fetch" type="emd:fetch-type"/>
    <xsd:attribute name="mapped-by" type="xsd:string"/>
</xsd:complexType>

<!-- ************************************************** -->
<xsd:complexType name="many-to-many">
    <xsd:sequence>
        <xsd:element name="order-by" type="emd:order-by" minOccurs="0"/>
        <xsd:element name="cascade" type="emd:cascade-type" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="xsd:string" use="required" />
    <xsd:attribute name="alias" type="xsd:string"/>
    <xsd:attribute name="target-entity" type="xsd:string"/>
    <xsd:attribute name="fetch" type="emd:fetch-type"/>
    <xsd:attribute name="mapped-by" type="xsd:string"/>
</xsd:complexType>

<!-- ************************************************** -->
<xsd:simpleType name="order-by">
    <xsd:restriction base="xsd:string"/>
</xsd:simpleType>

<!-- ************************************************** -->
<xsd:complexType name="cascade-type">
    <xsd:sequence>
        <xsd:element name="cascade-all" type="emd:emptyType" minOccurs="0"/>
        <xsd:element name="cascade-persist" type="emd:emptyType" minOccurs="0"/>
        <xsd:element name="cascade-remove" type="emd:emptyType" minOccurs="0"/>
        <xsd:element name="cascade-invalidate" type="emd:emptyType" minOccurs="0"/>
        <xsd:element name="cascade-merge" type="emd:emptyType" minOccurs="0"/>
        <xsd:element name="cascade-refresh" type="emd:emptyType" minOccurs="0"/>
    </xsd:sequence>
</xsd:complexType>

<!-- ************************************************** -->
```

```
    <xsd:complexType name="emptyType" />

    <!-- ************************************************** -->
    <xsd:complexType name="version">
          <xsd:attribute name="name" type="xsd:string" use="required" />
          <xsd:attribute name="alias" type="xsd:string"/>
          <xsd:attribute name="type" type="xsd:string" />
    </xsd:complexType>

    <!-- ************************************************** -->

    <xsd:complexType name="entity-listeners">
        <xsd:sequence>
            <xsd:element name="entity-listener" type="emd:entity-listener" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>

    <!-- ************************************************** -->
    <xsd:complexType name="entity-listener">
        <xsd:sequence>
            <xsd:element name="pre-persist" type="emd:pre-persist" minOccurs="0"/>
            <xsd:element name="post-persist" type="emd:post-persist" minOccurs="0"/>
            <xsd:element name="pre-remove" type="emd:pre-remove" minOccurs="0"/>
            <xsd:element name="post-remove" type="emd:post-remove" minOccurs="0"/>
            <xsd:element name="pre-invalidate" type="emd:pre-invalidate" minOccurs="0"/>
            <xsd:element name="post-invalidate" type="emd:post-invalidate" minOccurs="0"/>
            <xsd:element name="pre-update" type="emd:pre-update" minOccurs="0"/>
            <xsd:element name="post-update" type="emd:post-update" minOccurs="0"/>
            <xsd:element name="post-load" type="emd:post-load" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="class-name" type="xsd:string" use="required"/>
    </xsd:complexType>

    <!-- ************************************************** -->
    <xsd:complexType name="pre-persist">
        <xsd:attribute name="method-name" type="xsd:string" use="required"/>
    </xsd:complexType>

    <!-- ************************************************** -->
    <xsd:complexType name="post-persist">
        <xsd:attribute name="method-name" type="xsd:string" use="required"/>
    </xsd:complexType>

    <!-- ************************************************** -->
    <xsd:complexType name="pre-remove">
        <xsd:attribute name="method-name" type="xsd:string" use="required"/>
    </xsd:complexType>

    <!-- ************************************************** -->
    <xsd:complexType name="post-remove">
        <xsd:attribute name="method-name" type="xsd:string" use="required"/>
    </xsd:complexType>

    <!-- ************************************************** -->
    <xsd:complexType name="pre-invalidate">
        <xsd:attribute name="method-name" type="xsd:string" use="required"/>
    </xsd:complexType>

    <!-- ************************************************** -->
    <xsd:complexType name="post-invalidate">
        <xsd:attribute name="method-name" type="xsd:string" use="required"/>
    </xsd:complexType>

    <!-- ************************************************** -->
    1 <xsd:complexType name="pre-update">
        <xsd:attribute name="method-name" type="xsd:string" use="required"/>
    </xsd:complexType>

    <!-- ************************************************** -->
    <xsd:complexType name="post-update">
        <xsd:attribute name="method-name" type="xsd:string" use="required"/>
    </xsd:complexType>

    <!-- ************************************************** -->
    <xsd:complexType name="post-load">
        <xsd:attribute name="method-name" type="xsd:string" use="required"/>
    </xsd:complexType>

</xsd:schema>
```

Configurando a Integração de Cache

O WebSphere eXtreme Scale pode integrar-se com outros produtos relacionados ao armazenamento em cache. O JPA pode ser usado entre o WebSphere eXtreme Scale e o banco de dados para integrar as alterações como um utilitário de carga. Também é possível usar o provedor de cache dinâmico do WebSphere eXtreme

Scale para plugar o WebSphere eXtreme Scale no componente de cache dinâmico no WebSphere Application Server. Outra extensão para o WebSphere Application Server é o gerenciador de sessões HTTP do WebSphere eXtreme Scale, que pode ajudar a armazenar em cache as sessões HTTP.

Visão Geral da Integração de Cache: JPA, Sessões e Armazenamento em Cache Dinâmico

O elemento crucial que dá ao WebSphere eXtreme Scale a capacidade para executar com tal versatilidade e confiabilidade é seu aplicativo de conceitos de armazenamento em cache para otimizar a persistência e a nova coleta de dados em virtualmente qualquer ambiente de implementação.

Configurando Utilitários de Carga do JPA

Um Utilitário de Carga do Java Persistence API (JPA) é uma implementação de plug-in que utiliza JPA para interagir com o banco de dados.

Antes de Iniciar

- É necessário ter uma implementação do JPA, como Hibernate ou OpenJPA.
- O banco de dados pode ser qualquer back end que seja suportado pelo provedor JPA escolhido.
- É possível usar o plug-in do JPALoader ao armazenar dados usando a API ObjectMap. Use o plug-in do JPAEntityLoader ao armazenar dados usando a API EntityManager.

Sobre Esta Tarefa

Para obter mais informações sobre como o Java Persistence API (JPA) Loader funciona, consulte as informações no *Visão Geral do Produto*.

Procedimento

1. Configure os parâmetros necessários que o JPA requer para interagir com o banco de dados.

   Os seguintes parâmetros são necessários. Esses parâmetros são configurados no bean JPALoader ou JPAEntityLoader e no bean JPATxCallback.

   - **persistenceUnitName**: Especifica o nome da unidade de persistência. Esse parâmetro é necessário para duas finalidades: criar um EntityManagerFactory do JPA e localizar os metadados da entidade JPA no arquivo `persistence.xml`. Este atributo é configurado no bean JPATxCallback.
   - **JPAPropertyFactory**: Especifica o factory para criar um mapa de propriedade de persistência para substituir as propriedades de persistência padrão. Este atributo é configurado no bean JPATxCallback. Para configurar esse atributo, a configuração de estilo Spring é necessária.
   - **entityClassName**: Especifica o nome da classe de entidade que é necessário para usar os métodos JPA, como EntityManager.persist, EntityManager.find e assim por diante. O JPALoader requer este parâmetro, mas o parâmetro é opcional para **JPAEntityLoader**. No caso do JPAEntityLoader, se um parâmetro **entityClassName** não estiver configurado, a classe de entidade configurada no mapa de entidade ObjectGrid será usada. A mesma classe deve ser usada para o eXtreme Scale EntityManager e para o provedor JPA. Este atributo é configurado no bean JPALoader ou JPAEntityLoader.
   - **preloadPartition**: Especifica a partição na qual o pré-carregamento do mapa será iniciado. Se a partição do pré-carregamento for menor que zero ou

maior que o número total de partições menos 1, o pré-carregamento do mapa não será iniciado. O valor padrão é -1, significando que o pré-carregamento não é iniciado por padrão. Este atributo é configurado no bean JPALoader ou JPAEntityLoader.

Com exceção dos quatro parâmetros JPA a serem configurados no eXtreme Scale, os metadados JPA são utilizados para recuperar a chave das entidades JPA. Os metadados JPA podem ser configurados como anotação ou como um arquivo `orm.xml` especificado no arquivo `persistence.xml`. Eles não fazem parte da configuração do eXtreme Scale.

2. Configure os arquivos XML para a configuração do JPA.

   Para configurar o JPALoader ou o JPAEntityLoader, consulte as informações sobre os plug-ins do utilitário de carga no *Guia de Programação*.

   Configure um retorno de chamada de transação JPATxCallback junto com a configuração do utilitário de carga. O seguinte exemplo mostra um arquivo descritor XML ObjectGrid (`objectgrid.xml`), que tem o JPAEntityLoader e o JPATxCallback configurados:

Configurando um utilitário de carga incluindo um retorno de chamada - XML de exemplo

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
    <objectGrids>
      <objectGrid name="JPAEM" entityMetadataXMLFile="jpaEMD.xml">
        <bean id="TransactionCallback"
           className="com.ibm.websphere.objectgrid.jpa.JPATxCallback">
           <property
              name="persistenceUnitName"
              type="java.lang.String"
              value="employeeEMPU" />
       </bean>
       <backingMap name="Employee" pluginCollectionRef="Employee" />
    </objectGrid>
  </objectGrids>

  <backingMapPluginCollections>
    <backingMapPluginCollection id="Employee">
        <bean id="Loader"
          className="com.ibm.websphere.objectgrid.jpa.JPAEntityLoader">
       <property
               name="entityClassName"
               type="java.lang.String"
               value="com.ibm.ws.objectgrid.jpa.test.entity.Employee"/>
        </bean>
    </backingMapPluginCollection>
  </backingMapPluginCollections>
</objectGridConfig>
```

   Se desejar configurar um JPAPropertyFactory, será necessário usar uma configuração de estilo Spring. A seguir há uma amostra de arquivo de configuração XML, `JPAEM_spring.xml`, que configura um bean Spring a ser usado para as configurações do eXtreme Scale.

Configurando um utilitário de carga incluindo um factory de propriedade JPA - XML de exemplo

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:aop="http://www.springframework.org/schema/aop"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xmlns:objectgrid="http://www.ibm.com/schema/objectgrid"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans-2.0.xsd">

  <objectgrid:JPAEntityLoader id="jpaLoader" entityClassName="com.ibm.ws.objectgrid.jpa.test.entity.Employee"/>
  <objectgrid:JPATxCallback id="jpaTxCallback" persistenceUnitName="employeeEMPU" />
</beans>
```

O arquivo XML de configuração `Objectgrid.xml` vem a seguir. Observe que o nome do ObjectGrid é JPAEM, que corresponde ao nome do ObjectGrid no arquivo de configuração Spring `JPAEM_spring.xml`.

Configuração de um utilitário de carga JPAEM - XML de exemplo

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
  <objectGrids>
    <objectGrid name="JPAEM" entityMetadataXMLFile="jpaEMD.xml">
      <bean id="TransactionCallback"
            className="{spring}jpaTxCallback"/>
        <backingMap name="Employee" pluginCollectionRef="Employee"
                    writeBehind="T4"/>
      </objectGrid>
  </objectGrids>

  <backingMapPluginCollections>
    <backingMapPluginCollection id="Employee">
       <bean id="Loader" className="{spring}jpaLoader" />
    </backingMapPluginCollection>
  </backingMapPluginCollections>
</objectGridConfig>
```

Uma entidade pode ser anotada com as anotações do JPA e com as anotações do gerenciador de entidade do eXtreme Scale. Cada anotação possui um XML equivalente que pode ser utilizado. Assim, o eXtreme Scale incluiu o espaço de nomes Spring. Também é possível configurá-los utilizando o suporte a espaço de nomes Spring.

Configurando um Atualizador de Dados Baseado em Tempo do JPA

É possível configurar uma atualização de banco de dados baseado em tempo usando o XML para uma configuração do eXtreme Scale local ou distribuída. Também é possível definir uma configuração local programaticamente.

Sobre Esta Tarefa

Para obter mais informações sobre como o atualizador de dados baseado em tempo do Java Persistence API (JPA) trabalha, consulte as informações no *Guia de Programação*.

Procedimento

Criar uma configuração timeBasedDBUpdate.

- **Com o arquivo XML:**

  O seguinte exemplo mostra um arquivo `objectgrid.xml` que contém uma configuração timeBasedDBUpdate:

  Atualizador baseado em tempo do JPA - exemplo de XML

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
    <objectGrids>
        <objectGrid name="changeOG"
            entityMetadataXMLFile="userEMD.xml">
            <backingMap name="user" >
                <timeBasedDBUpdate timestampField="rowChgTs"
                      persistenceUnitName="userderby"
                      entityClass="com.test.UserClass"
                      mode="INVALIDATE_ONLY"
                />
            </backingMap>
        </objectGrid>
    </objectGrids>
    <backingMapPluginCollections>
</objectGridConfig>
```

Neste exemplo, o mapa "user" é configurado com a atualização do banco de dados baseada em tempo. O modo de atualização de banco de dados é INVALIDATE_ONLY e o campo do registro de data e hora é rowChgTs.

Quando um ObjectGrid "changeOG" distribuído é iniciado no servidor de contêiner, um encadeamento de atualização de banco de dados baseado em tempo é iniciado automaticamente na partição 0.

- **Programaticamente:**

  Se você criar um ObjectGrid local, também é possível criar um objeto TimeBasedDBUpdateConfig e configurá-lo na instância BackingMap:

  ```
  public void setTimeBasedDBUpdateConfig(TimeBasedDBUpdateConfig dbUpdateConfig);
  ```

  Para obter mais informações sobre como configurar um objeto na instância BackingMap, consulte as informações sobre a interface BackingMap na Documentação de API.

  Por outro lado, não é possível anotar o campo do registro de data e hora na classe de entidade usando a anotação com.ibm.websphere.objectgrid.jpa.dbupdate.annotation.Timestamp. Ao configurar o valor na classe, não é necessário configurar o timestampField na configuração XML.

O que Fazer Depois

Inicie um atualizador de dados baseado em tempo do JPA. Consulte as informações sobre a iniciação de um atualizador de dados baseado em tempo do JPA no *Guia de Programação* para obter mais informações.

Configurando Plug-ins do Cache JPA

O WebSphere eXtreme Scale inclui plug-ins de cache de nível 2 para ambos os provedores, OpenJPA e Hibernate Java Persistence API (JPA).

Propriedades de Configuração do Cache JPA do ObjectGrid

O plug-in do cache JPA do ObjectGrid pode ser configurado com as seguintes propriedades, sendo que todas são opcionais.

ObjectGridName
: Especifica o nome exclusivo do ObjectGrid. O valor padrão é o nome da unidade de persistência definido. Se o nome da unidade de persistência não estiver disponível a partir do provedor JPA, um nome gerado será usado.

ObjectGridType
: Especifica o tipo de ObjectGrid.

  Valores válidos:

  - **EMBEDDED**: O tipo de configuração padrão e recomendado. As configurações padrão incluem: `NumberOfPartitions=1`, `ReplicaMode=SYNC`, `ReplicaReadEnabled=true` e `MaxNumberOfReplicas=47`. Use o parâmetro **ReplicaMode** para configurar o modo de replicação e o parâmetro **MaxNumberOfReplicas** para configurar o número máximo de réplicas. Se um sistema tiver mais de 47 Java Virtual Machines, configure o valor **MaxNumberOfReplicas** para que seja igual ao número de Java Virtual Machines.
  - **EMBEDDED_PARTITION**: O tipo a ser usado quando o sistema precisar armazenar em cache uma grande quantidade de dados em um

sistema distribuído. O número padrão de partições é 47 com um modo de réplica de NONE. Em um sistema pequeno com apenas alguns Java Virtual Machines, configure o valor **NumberOfPartitions** para que seja igual ou menor que o número de Java Virtual Machines. É possível especificar os valores **ReplicaMode**, **NumberOfPartitions** e **ReplicaReadEnabled** para ajustar o sistema.

- **REMOTE**: O cache tenta se conectar com um ObjectGrid remoto e distribuído a partir do serviço de catálogo.

NumberOfPartitions

Valores válidos: maior ou igual a 1Especifica o número de partições a serem usadas para o cache. Essa propriedade é aplicada quando o valor ObjectGridType for configurado para EMBEDDED_PARTITION. O valor padrão é 47. Para o tipo EMBEDDED, o valor **NumberOfPartitions** é sempre 1.

ReplicaMode

Valores válidos: SYNC/ASYNC/NONEEspecifica o método que é usado para copiar o cache para as réplicas. Essa propriedade é aplicada ao configurar o valor ObjectGridType para EMBEDDED ou EMBEDDED_PARTITION. O valor padrão é NONE para o tipo EMBEDDED_PARTITION e SYNC para o tipo EMBEDDED. Se o valor **ReplicaMode** for configurado para NONE para o ObjectGridType EMBEDDED, o tipo EMBEDDED ainda usará um **ReplicaMode** de SYNC.

ReplicaReadEnabled

Valores válidos: TRUE ou FALSEQuando ativado, os clientes leem a partir das réplicas. Essa propriedade é aplicada para o tipo EMBEDDED_PARTITION. O valor padrão é FALSE para o tipo EMBEDDED_PARTITION. O tipo EMBEDDED sempre é configurado para o valor **ReplicaReadEnabled** para TRUE.

MaxUsedMemory

Valores válidos: TRUE ou FALSEAtiva a liberação das entradas de cache quando a memória se tornar limitada. O valor padrão é TRUE e despeja os dados quando o limite de uso do heap da JVM exceder 70%. É possível modificar a porcentagem do limite de uso do heap JVM padrão ao configurar a propriedade memoryThresholdPercentage no arquivo objectGridServer.properties e colocar esse arquivo no caminho de classe. Para obter mais informações sobre os evictors, consulte as informações sobre os evictors no *Visão Geral do Produto*. Para obter mais informações sobre o arquivo de propriedades do servidor, consulte o *Guia de Administração*.

MaxNumberOfReplicas

Valores válidos: maior ou igual a 1Especifica o número máximo de réplicas a ser usado para o cache. Esse valor á aplicado apenas para o tipo EMBEDDED. Esse número deve ser igual ou maior que o número de Java Virtual Machines em um sistema. O valor padrão é 47.

As propriedades NumberOfPartitions, ReplicaMode, ReplicaReadEnabled e MaxNumberOfReplicas são fatores de implementação do ObjectGrid. As propriedades NumberOfPartitions, ReplicaMode e ReplicaReadEnabled aplicam-se ao tipo EMBEDDED_PARTITION. ReplicaMode e MaxNumberOfReplicas aplicam-se ao tipo EMBEDDED.

Considerações sobre EMBEDDED e EMBEDDED_PARTITION

Os tipos de ObjectGrid integrados utilizam as propriedades de configuração anteriormente descritas para configurar e implementar um conjunto de servidores de contêineres do ObjectGrid e um serviço de catálogo, quando necessário. O ciclo de vida dos contêineres é limitado ao aplicativo JPA e é colocado no caminho de classe do aplicativo. Quando um aplicativo for iniciado, o plug-in detecta ou inicia automaticamente um serviço de catálogo, inicia um contêiner ou se conecta ao serviço de catálogo. Em seguida, o plug-in se comunica com o contêiner ObjectGrid e seus equivalentes que estão em execução em outros processos do servidor de aplicativos que usam a conexão do cliente.

Cada entidade JPA possui um mapa de apoio independente designado para usar o nome da classe da entidade. Cada BackingMap possui os seguintes atributos.

- `readOnly="false"`
- `copyKey="false"`
- `lockStrategy="NONE"`
- `copyMode="NO_COPY"`

Nota: Quando estiver usando o valor de ObjectGridType `EMBEDDED` ou `EMBEDDED_PARTITION` em um ambiente do Java SE, use o método System.exit(0) no final do programa para parar o servidor eXtreme Scale integrado. Caso contrário, o programa poderá parecer não estar respondendo.

Padrões do Valor ObjectGridType

O valor ObjectGridType especifica a topologia na qual o cache ObjectGrid é implementado. O tipo padrão e com melhor desempenho é EMBEDDED. As seguintes seções descrevem as propriedades padrão para cada um dos valores ObjectGridType.

Padrões da topologia do cache JPA do ObjectGrid EMBEDDED

Quando você estiver usando o tipo ObjectGrid `EMBEDDED`, os seguintes valores de propriedade padrão serão usados se você não especificar nenhum valor na configuração:

- **ObjectGridName**: nome da unidade de persistência
- **ObjectGridType**: `EMBEDDED`
- **NumberOfPartitions**: `1` (não é possível alterar quando o tipo de ObjectGrid for `EMBEDDED`)
- **ReplicaMode**: `SYNC`
- **ReplicaReadEnabled**: `TRUE` (não é possível alterar quando o tipo de ObjectGrid for `EMBEDDED`)
- **MaxUsedMemory**: `TRUE`
- **MaxNumberOfReplicas**: `47` (deve ser menor ou igual ao número de Java Virtual Machines em um sistema distribuído)

E necessário especificar um único valor de ObjectGridName para evitar conflitos de nomenclatura. O valor de MaxNumberOfReplicas deve ser igual ou maior que o número total de Java Virtual Machines no sistema.

Topologia de Cache do ObjectGrid REMOTE

O tipo de ObjectGrid REMOTE não requer nenhuma configuração de propriedade porque a política de ObjectGrid e de implementação é definida separadamente a partir do aplicativo JPA. O plug-in do cache JPA conecta-se remotamente a um ObjectGrid remoto existente.

Como toda a interação com o ObjectGrid é remota, essa topologia possui o menor desempenho entre todos os tipos de ObjectGrid.

Considerações e Configuração do Serviço de Catálogo

Ao executar uma topologia `EMBEDDED` ou `EMBEDDED_PARTITION`, o plug-in de cache JPA inicia automaticamente um único serviço de catálogo dentro dos processos de aplicativos se necessário. Em um ambiente de produção, você deve criar um domínio do serviço de catálogo. Para obter mais informações sobre a definição de um serviço de catálogo, consulte .as informações sobre o serviço de catálogo de alta disponibilidade no *Visão Geral do Produto*

Se você estiver executando dentro de um processo do WebSphere Application Server, o plug-in do cache JPA conectará automaticamente ao serviço de catálogo ou ao domínio do serviço de catálogo definido para a célula do WebSphere Application Server. Para obter mais informações sobre como definir um domínio do serviço de catálogo, consulte as informações sobre como iniciar o serviço de catálogo no *Guia de Administração*.

Se você não estiver executando seus servidores dentro de um processo do WebSphere Application Server, os hosts e as portas do domínio do serviço de catálogo serão especificados usando o arquivo de propriedades denominado `objectGridServer.properties`. Este arquivo deve ser armazenado no caminho de classe do aplicativo e ter a propriedade **catalogServiceEndPoints** definida. A grade do serviço de catálogo é iniciada independentemente dos processos do aplicativo e deve ser iniciada antes que os processos do aplicativo sejam iniciados.

O formato do arquivo `objectGridServer.properties` é:

```
catalogServiceEndPoints=<hostname1>:<port1>,<hostname2>:<port2>
```

Plug-in do Cache JPA

O WebSphere eXtreme Scale inclui plug-ins de cache de nível 2 (L2) para ambos os provedores OpenJPA e Hibernate Java Persistence API (JPA).

Usar o eXtreme Scale como um provedor de cache L2 aumenta o desempenho na leitura e consulta de dados e reduz a carga para o banco de dados. O WebSphere eXtreme Scale tem vantagens sobre as implementações de cache integradas porque o cache é automaticamente replicado entre todos os processos. Quando um cliente armazena em cache um valor, todos os outros clientes conseguem usar o valor armazenado em cache que está localmente na memória.

Com os plug-ins de cache do ObjectGrid OpenJPA e Hibernate, é possível criar três tipos de topologia: integrada, integrada-particionada e remota.

Topologia Integrada

Uma topologia integrada cria um servidor eXtreme Scale dentro do espaço do processo de cada aplicativo. O OpenJPA e o Hibernate lêem a cópia na memória do cache diretamente e gravam para todas as outras cópias. É possível melhorar o desempenho de gravação usando replicação assíncrona. Esta topologia padrão executa melhor quando a quantidade de dados armazenados em cache é pequena

o suficiente para ajustar-se a um único processo.

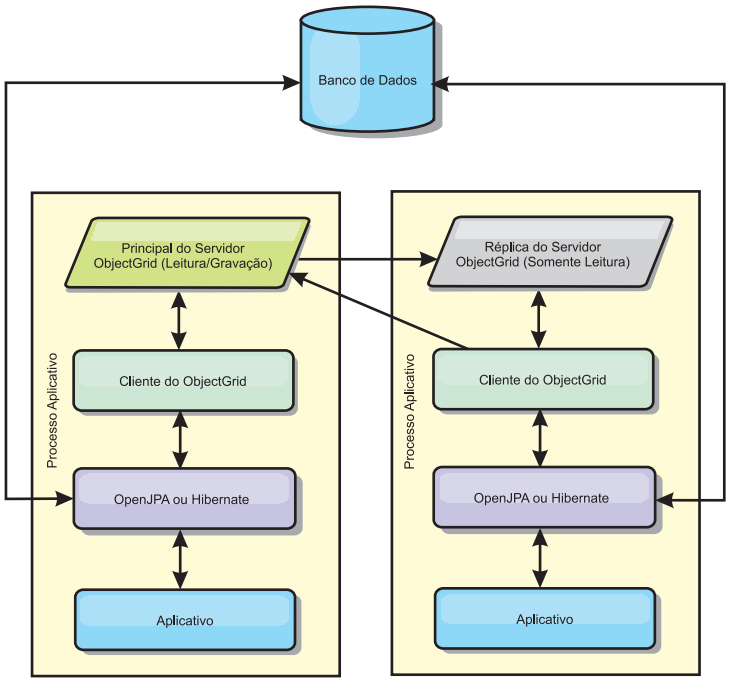


Figura 11. Topologia Integrado do JPA

Vantagens:

- Todas as leituras de cache são muito rápidas, com acessos locais.
- Simples para configurar.

Limitações:

- A quantidade de dados é limitada ao tamanho do processo.
- Todas as atualizações de cache são enviadas para um processo.

Topologia Integrada e Particionada

Quando os dados armazenados em cache são muito grandes para ajustar-se em um único processo, a topologia integrada e particionada utiliza partições do ObjectGrid para dividir os dados sobre vários processos. O desempenho não é tão alto quanto o da topologia integrada porque a maioria dos caches é remota. Porém, ainda é possível usar esta opção quando a latência do banco de dados é alta.

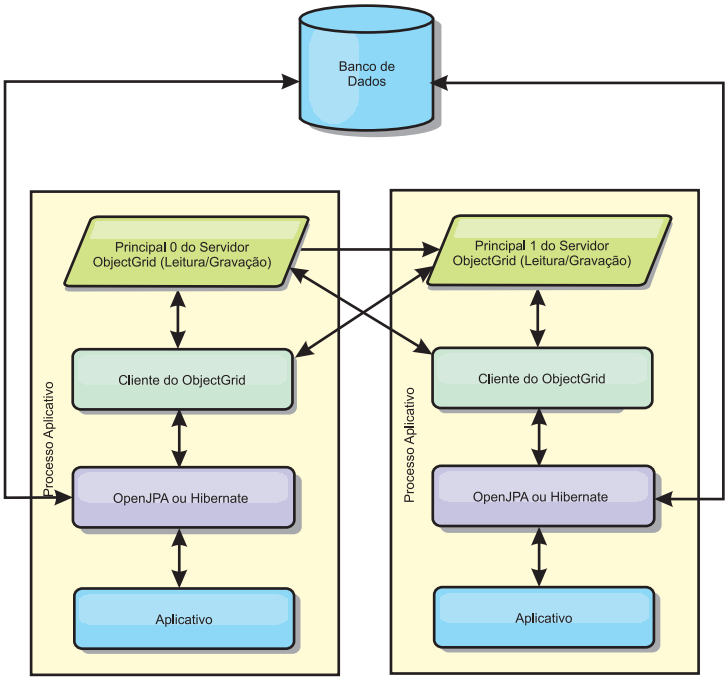


Figura 12. Topologia Particionada Integrada do JPA

Vantagens:

- Armazena grandes quantidades de dados.
- Simples para configurar
- As atualizações de cache são propagadas para vários processos.

Limitação:

- A maioria das leituras e atualizações de cache é remota.

Por exemplo, para armazenar em cache 10 GB de dados com no máximo 1 GB por JVM, dez Java Virtual Machines serão necessários. Portanto, o número de partições deve ser configurado para 10 ou mais. De forma ideal, o número de partições deve ser definido para um número primo no qual cada shard armazena uma quantidade razoável de memória. Geralmente, a configuração de numberOfPartitions é igual ao número de Java Virtual Machines. Com esta configuração, cada JVM armazena uma partição. Se você ativar replicação, será necessário aumentar o número de Java Virtual Machines no sistema. Caso contrário, cada JVM também armazenará uma partição de réplica, que consome tanta memória quanto uma partição primária.

Leia sobre o dimensionamento de memória e o cálculo da contagem de partições no *Guia de Administração* para maximizar o desempenho da sua configuração escolhida.

Por exemplo, em um sistema com 4 Java Virtual Machines, e o valor da configuração de numberOfPartitions de 4, cada JVM hospeda uma partição primária. Uma operação de leitura tem 25% mais chance de buscar dados de uma partição disponível localmente, o que é muito mais rápido comparado ao obter dados de uma JVM remota. Se uma operação de leitura, como a execução de uma consulta, precisar buscar uma coleta de dados que envolva 4 partições igualmente, 75% das chamadas são remotas e 25% das chamadas são locais. Se a configuração de ReplicaMode for definida para `SYNC` ou `ASYNC` e a configuração de

ReplicaReadEnabled for definida para `true`, as quatro partições de réplica são criadas e espalhadas pelos quatro Java Virtual Machines. Cada JVM hospeda uma partição primária e uma partição de réplica. A chance de que a operação de leitura execute localmente aumenta em 50%. A operação de leitura que busca uma coleta de dados que envolve quatro partições igualmente tem somente 50% de chamadas remotas e 50% de chamadas locais. Chamadas locais são muito mais rápidas do que chamadas remotas. Sempre que ocorrem chamada remotas, o desempenho cai.

Topologia Remota

Uma topologia remota armazena todos os dados armazenados em cache em um ou mais processos separados, reduzindo o uso da memória dos processos do aplicativo. É possível aproveitar as vantagens da distribuição de dados em processos separados implementando uma grade do eXtreme Scale replicada particionada. Ao contrário das configurações integradas e particionadas integradas descritas nas seções anteriores, se quiser gerenciar a grade remota, você deverá fazer isso independentemente do aplicativo e provedor JPA. Leia sobre o monitoramento do seu ambiente de implementação para obter mais informações sobre o gerenciamento de uma implementação em grade do eXtreme Scale.

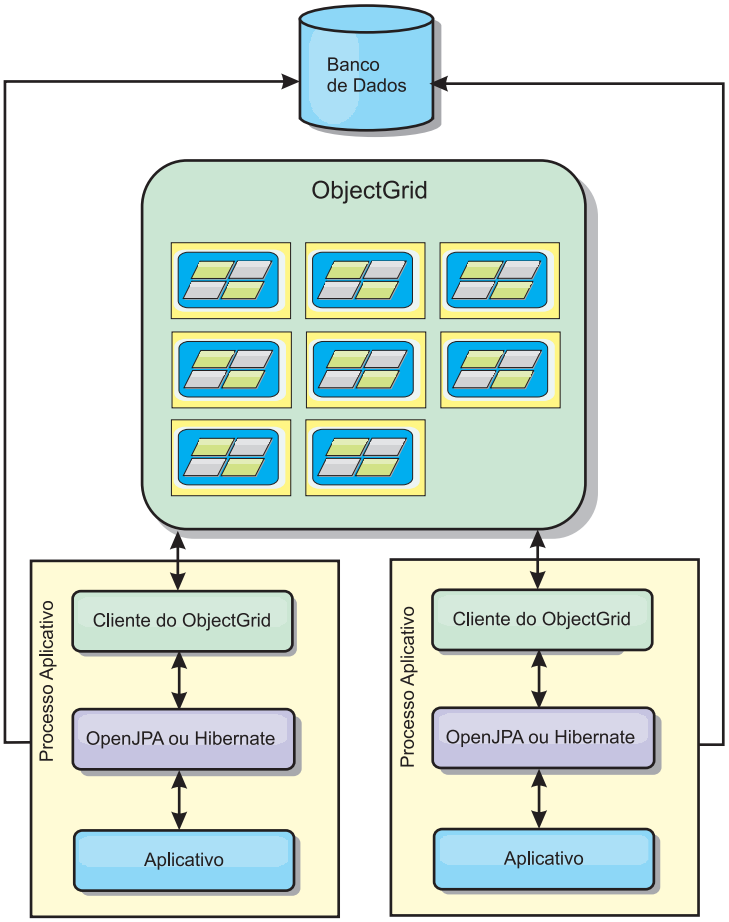


Figura 13. Topologia Remota do JPA

Vantagens:

- Armazena grandes quantidades de dados.
- O processo aplicativo é livre de dados em cache.

- As atualizações de cache são propagadas para vários processos.
- Opções de configuração muito flexíveis.

Limitação:

- Todas as leituras e atualizações de cache são remotas.

Configuração de Plug-in do Cache de Hibernação

Um cache do eXtreme Scale pode ser ativado para Hibernate por meio da definição das propriedades no arquivo de configuração.

7.1+ Para integração com o WebSphere Application Server, o plug-in de cache de hibernação é compactado no `oghibernate-cache.jar` e instalado em `WAS_HOME/optionalLibraries/ObjectGrid`. Para usar o plug-in de cache de hibernação, é necessário incluir `oghibernate-cache.jar` na biblioteca de hibernação. Por exemplo, se você incluir a biblioteca de hibernação em seu aplicativo, será necessário incluir o arquivo `oghibernate-cache.jar` também. Se você definir uma biblioteca compartilhada para incluir a biblioteca de hibernação, será necessário colocar `oghibernate-cache.jar` no diretório da biblioteca compartilhada.

7.1+ O eXtreme Scale, Versão 7.1, não instala cglib.jar no ambiente do WebSphere Application Server. Se você tiver aplicativos existentes ou bibliotecas compartilhadas, como hibernação, que depende de `cglib.jar`, localize `cglib.jar` e inclua-o no caminho de classe. Por exemplo, se o aplicativo incluir todos os arquivos JAR da biblioteca de hibernação, mas excluir `cglib.jar` disponível com a hibernação, será necessário incluir o `cglib.jar` que vem da hibernação em seu aplicativo.

Configurações

A sintaxe para configurar a propriedade no arquivo `persistence.xml` é a seguinte:

persistence.xml

```
<property name="hibernate.cache.provider_class"
         value="com.ibm.websphere.objectgrid.hibernate.cache.ObjectGridHibernateCacheProvider" />
<property name="hibernate.cache.use_query_cache" value="true"/>
<property name="objectgrid.configuration" value="<property>=<value>,..." />
<property name="objectgrid.hibernate.regionNames" value="<regionName>,.." />
```

A sintaxe para configurar a propriedade no arquivo `hibernate.cfg.xml` é a seguinte:

hibernate.cfg.xml

```
<property name="cache.provider_class">com.ibm.websphere.objectgrid.
          hibernate.cache.ObjectGridHibernateCacheProvider</property>
<property name="cache.use_query_cache">true</property>
<property name="objectgrid.configuration"><property>=<value>,...</property>
<property name="objectgrid.hibernate.regionNames"><regionName>,...</property>
```

A propriedade provider_class é com.ibm.websphere.objectgrid.hibernate.cache.ObjectGridHibernateCacheProvider. Para ativar o cache de consulta, defina o valor para `true` na propriedade use_query_cache. Use a propriedade objectgrid.configuration para especificar as propriedades de configuração do cache do eXtreme Scale.

E necessário especificar um único valor da propriedade ObjectGridName para evitar potenciais conflitos de nomenclatura. As outras propriedades de configuração de cache do eXtreme Scale são opcionais.

A propriedade objectgrid.hibernate.regionNames é opcional e deve ser especificada quando os valores regionNames forem definidos após o cache do eXtreme Scale ser inicializado. Considere o exemplo de uma classe de entidade mapeada para uma regionName com a classe de entidade não-especificada no arquivo `persistence.xml` ou não incluída no arquivo de mapeamento Hibernate. Além disso, suponha que ele possua a anotação Entity. Então, o regionName para esta classe de entidade será resolvido no momento do carregamento da classe quando o cache do eXtreme Scale for inicializado. Outro exemplo é a execução do método Query.setCacheRegion(String regionName) que é executado após a inicialização do cache do eXtreme Scale. Nas situações acima, inclua todos os regionNames possíveis determinados como dinâmico na propriedade objectgrid.hibernate.regionNames para que o cache do eXtreme Scale possa preparar o BackingMaps para todos os regionNames.

A seguir estão exemplos dos arquivos `persistence.xml` e `hibernate.cfg.xml`:

persistence.xml

```
<persistence-unit name="testPU2">
    <provider>org.hibernate.ejb.HibernatePersistence</provider>
    <class>com.ibm.websphere.objectgrid.jpa.test.Department</class>
    <properties>
      <property name="hibernate.show_sql" value="true" />
      <property name="hibernate.connection.url" value="jdbc:derby:DB_testPU2;create=true" />
      <property name="hibernate.connection.driver_class" value="org.apache.derby.jdbc.EmbeddedDriver" />

      <property name="hibernate.cache.provider_class" value="com.ibm.websphere.objectgrid.
               hibernate.cache.ObjectGridHibernateCacheProvider" />
      <property name="hibernate.cache.use_query_cache" value="true"/>
      <property name="objectgrid.configuration" value="ObjectGridName=myOGName,ObjectGridType=
               EMBEDDED,MaxNumberOfReplicas=4" writeBehind=true, writeBehindInterval=5000,
               writeBehindPoolSize=10, writeBehindMaxBatchSize=1000" />
      <property name="objectgrid.hibernate.regionNames" value="queryRegion1, queryRegion2" />
    </properties>
</persistence-unit>
```

7.1+ A Versão 7.1 introduz opções adicionais de configuração específicas da função write behind para o plug-in do cache de Hibernação, além das opções padrão de configuração do plug-in do cache JPA.

writeBehind

Valores válidos: TRUE ou FALSE

Valor padrão: FALSE

Quando writeBehind é ativada, as atualizações são temporariamente armazenadas em um armazenamento de dados de escopo JVM até a condição writeBehindInterval ou writeBehindMaxBatchSize ser atendida.

Atenção: A menos que writeBehind seja ativada, as outras definições de configuração write behind são desconsideradas.

writeBehindInterval

Valores válidos: maior ou igual a 1

Valor padrão: 5000 (5 segundos)

Especifica o intervalo de tempo em milissegundos para nivelar as atualizações para o cache.

writeBehindPoolSize

Valores válidos: maior ou igual a 1

Valor padrão: 5

Especifica o tamanho máximo do conjunto de encadeamentos usado no nivelamento de atualizações para o cache.

writeBehindMaxBatchSize

Valores válidos: maior ou igual a 1

Valor padrão: 1000

Especifica o tamanho máximo do lote por cache de região no nivelamento de atualizações para o cache.

O exemplo de código precedente exibe a seguinte configuração da função write behind:

```
writeBehind=true, writeBehindInterval=5000, writeBehindPoolSize=10, writeBehindMaxBatchSize=1000
```

em que

- writeBehind=TRUE ativa a função write behind
- writeBehindInterval=5000 significa que as atualizações serão niveladas para o cache aproximadamente a cada 5 segundos
- writeBehindPoolSize=10 indica que o número máximo de encadeamentos usado para executar o trabalho é de 10 encadeamentos, ao nivelar as atualizações para o cache
- writeBehindMaxBatchSize=1000 significa que, se as atualizações armazenadas no armazenamento write behind de um cache de região exceder 1000 entradas, as atualizações serão niveladas para o cache, mesmo que a condição writeBehindInterval especificada não seja atendida. Em outras palavras, as atualizações serão niveladas para o cache aproximadamente a cada 5 segundos ou sempre que o tamanho do armazenamento write behind do cache de cada região exceder 1000 entradas. Observe, no caso da condição writeBehindMaxBatchSize atendida; apenas o cache da região que atende esta condição nivelará suas atualizações no armazenamento write behind para o cache. Um cache de região geralmente é correspondente a uma entidade ou a uma consulta.

Importante:

Tome cuidado ao usar a configuração da função write behind. Ela introduz uma latência mais longa de sincronização de dados por meio de todas as JVMs e uma chance mais alta de atualizações perdidas. Em um sistema usando a configuração write behind para quatro ou mais JVMs, a atualização executada em uma JVM terá um atraso de aproximadamente 15 segundos antes da atualização ficar disponível para outras JVMs. Se quaisquer duas JVMs atualizarem a mesma entrada, aquela que nivelar a atualização primeiro perderá sua atualização.

hibernate.cfg.xml

```
<hibernate-configuration>
    <session-factory>
        <!-- Database connection settings -->
        <property name="connection.driver_class">org.apache.derby.jdbc.EmbeddedDriver</property>
        <property name="connection.url">jdbc:derby:DB_testPU2;create=true</property>
        <!-- ObjectGrid cache setting-->
        <property name="cache.provider_class">com.ibm.websphere.objectgrid.hibernate.cache.
                 ObjectGridHibernateCacheProvider</property>
        <property name="cache.use_query_cache">true</property>
        <property name="objectgrid.configuration">ObjectGridName=myOGName,
                 ObjectGridType=EMBEDDED,MaxNumberOfReplicas=4 </property>
        <property name="objectgrid.hibernate.regionNames">queryRegion1, queryRegion2</property>

        <mapping resource="com/ibm/websphere/objectgrid/jpa/test/Employee.hbm.xml"/>

    </session-factory>
</hibernate-configuration>
```

Pré-carregando Dados no Cache do ObjectGrid

É possível usar o método preload da classe ObjectGridHibernateCacheProvider para pré-carregar dados no cache do ObjectGrid para uma classe de entidade.

Exemplo 1

Usando EntityManagerFactory

```
EntityManagerFactory emf = Persistence.createEntityManagerFactory("testPU");
ObjectGridHibernateCacheProvider.preload("objectGridName", emf, TargetEntity.class, 100, 100);
```

Exemplo 2

Using SessionFactory

```
org.hibernate.cfg.Configuration cfg = new Configuration();
// use addResource, addClass, and setProperty method of Configuration to prepare
// configuration required to create SessionFactor
SessionFactory sessionFactory= cfg.buildSessionFactory();
ObjectGridHibernateCacheProvider.preload("objectGridName", sessionFactory, TargetEntity.class, 100, 100);
```

Nota:

1. Em um sistema distribuído, este mecanismo de pré-carregamento somente pode ser chamado de uma Java virtual machine. O mecanismo de pré-carregamento não pode executar simultaneamente a partir de várias Java Virtual Machines.
2. Antes de executar o pré-carregamento, você deve inicializar o cache doeXtreme Scale por meio da criação do EntityManager usando EntityManagerFactory para ter todos os BackingMaps correspondentes criados; caso contrário, o pré-carregamento força o cache a ser inicializado com somente um BackingMap padrão para suportar todas as entidades. Isto significa que um único BackingMap é compartilhado por todas as entidades.

Customizando a Configuração do Cache Hibernate com XML

Para a maioria dos cenários, a configuração das propriedades de cache devem ser suficientes. Para customizar ainda mais o ObjectGrid usado pelo cache, é possível fornecer os arquivos XML de configuração Hibernate ObjectGrid no diretório `META-INF` da mesma forma que o arquivo `persistence.xml`. Durante a inicialização, o cache tentará localizar esses arquivos XML e processá-los, caso sejam localizados.

Há três tipos de arquivos XML de configuração Hibernate ObjectGrid: `hibernate-objectGrid.xml` (configuração ObjectGrid), `hibernate-objectGridDeployment.xml` (política de implementação) e `hibernate-objectGrid-client-override.xml` (configuração de substituição do ObjectGrid cliente). Dependendo da topologia do eXtreme Scale configurada, é possível fornecer qualquer um destes três arquivos XML para customizar essa topologia.

Para os tipos EMBEDDED e EMBEDDED_PARTITION, é possível fornecer qualquer um dos três arquivos XML para customizar o ObjectGrid, a política de implementação e a configuração de substituição do ObjectGrid cliente.

Para um REMOTE ObjectGrid, o cache não cria um ObjectGrid dinâmico. O cache obtém apenas um ObjectGrid do lado do cliente a partir do serviço do catálogo. É possível fornecer apenas um arquivo `hibernate-objectGrid-client-override.xml` para customizar a configuração de substituição do ObjectGrid do cliente.

1. **Configuração do ObjectGrid**: Use o arquivo `META-INF/hibernate-objectGrid.xml`. Este arquivo é utilizado para customizar a configuração do ObjectGrid para os tipos EMBEDDED e EMBEDDED_PARTITION. Com o tipo

REMOTE, este arquivo é ignorado. Por padrão, cada classe de entidade possui uma regionName associada (padrão para o nome da classe de entidade) que é mapeada para uma configuração de BackingMap denominada como regionName na configuração do ObjectGrid. Por exemplo, a classe de entidade com.mycompany.Employee possui uma regionName associada definida por padrão como com.mycompany.Employee BackingMap. A configuração padrão do BackingMap é readOnly="false", copyKey="false", lockStrategy="NONE" e `copyMode="NO_COPY"`. É possível customizar alguns BackingMaps com uma configuração escolhida. A palavra-chave reservada "ALL_ENTITY_MAPS" pode ser utilizada para representar todos os mapas, exceto outros mapas customizados listados no arquivo `hibernate-objectGrid.xml`. Os BackingMaps que não estiverem listados neste arquivo `hibernate-objectGrid.xml` utilizarão a configuração padrão.

2. **Configuração do ObjectGridDeployment**: Use o arquivo `META-INF/hibernate-objectGridDeployment.xml`. Este arquivo é utilizado para customizar a política de implementação. Ao customizar a política de implementação, se o arquivo `hibernate-objectGridDeployment.xml` for fornecido, a política de implementação padrão será descartada. Todos os valores de atributo da política de implementação serão fornecidos pelo arquivo `hibernate-objectGridDeployment.xml`.
3. **Configuração de substituição do ObjectGrid do cliente**: Use o arquivo `META-INF/hibernate-objectGrid-client-override.xml`. Este arquivo é utilizado para customizar um ObjectGrid do lado do cliente. Por padrão, o cache do ObjectGrid aplica uma configuração de substituição do cliente padrão que desativa o cache local. Se o aplicativo requerer um cache perto, ele pode fornecer este arquivo e especificar `numberOfBuckets="xxx"`. A substituição do cliente padrão desativa o cache próximo ao configurar `numberOfBuckets="0"`. O cache perto pode estar ativo ao reconfigurar numberOfBuckets com um valor superior a 0 por meio de `hibernate-objectGrid-client-override.xml`. A maneira como o arquivo `hibernate-objectGrid-client-override.xml` funciona é similar ao `hibernate-objectGrid.xml`: ele substitui ou estende a configuração de substituição do ObjectGrid do cliente padrão.

Exemplos de Arquivo XML Hibernate ObjectGrid

Os arquivos XML Hibernate ObjectGrid devem ser criados com base na configuração de uma unidade de persistência.

Um arquivo `persistence.xml` de exemplos que representa a configuração de uma unidade de persistência está a seguir:

persistence.xml

```
<persistence xmlns="http://java.sun.com/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  version="1.0">
  <persistence-unit name="AnnuityGrid">
    <provider>org.hibernate.ejb.HibernatePersistence</provider>

    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityPersistebleObject</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.FixedAnnuity</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.EquityAnnuity</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Person</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityHolder</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Address</class>

    <exclude-unlisted-classes>true</exclude-unlisted-classes>

    <properties>
      <property name="hibernate.show_sql" value="false" />
      <property name="hibernate.connection.url" value="jdbc:db2:Annuity" />
      <property name="hibernate.connection.driver_class" value="com.ibm.db2.jcc.DB2Driver" />
```

```
      <property name="hibernate.default_schema" value="EJB30" />

      <!-- Cache -->
      <property name="hibernate.cache.provider_class"
        value="com.ibm.websphere.objectgrid.hibernate.cache.ObjectGridHibernateCacheProvider" />
      <property name="hibernate.cache.use_query_cache" value="true"/>
      <property name="objectgrid.configuration" value="ObjectGridType=EMBEDDED,
              ObjectGridName=Annuity, MaxNumberOfReplicas=4" />
    </properties>
  </persistence-unit>

</persistence>
```

A seguir está o arquivo`hibernate-objectGrid.xml` que corresponde ao arquivo `persistence.xml`:

hibernate-objectGrid.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
  xmlns="http://ibm.com/ws/objectgrid/config">
  <objectGrids>
    <objectGrid name="Annuity">
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity" />
      <backingMap name="defaultCacheMap" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="defaultCacheMap" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout" />
      <backingMap name="org.hibernate.cache.UpdateTimestampsCache" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="org.hibernate.cache.UpdateTimestampsCache" />
      <backingMap name="org.hibernate.cache.StandardQueryCache" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="org.hibernate.cache.StandardQueryCache" />
    </objectGrid>
  </objectGrids>
  <backingMapPluginCollections>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="defaultCacheMap">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="org.hibernate.cache.UpdateTimestampsCache">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>


    </backingMapPluginCollection>
    <backingMapPluginCollection id="org.hibernate.cache.StandardQueryCache">
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
```

```
      </bean>
    </backingMapPluginCollection>
  </backingMapPluginCollections>
</objectGridConfig>
```

Nota: Os mapas org.hibernate.cache.UpdateTimestampsCache, org.hibernate.cache.StandardQueryCache e defaultCacheMap são necessários.

A seguir está o arquivo `hibernate-objectGridDeployment.xml` que corresponde ao `persistence.xml`:

hibernate-objectGridDeployment.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
  xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">
  <objectgridDeployment objectgridName="Annuity">
    <mapSet name="MAPSET_Annuity" numberOfPartitions="1" numInitialContainers="1" minSyncReplicas="0"
          maxSyncReplicas="4" maxAsyncReplicas="0" replicaReadEnabled="true">
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity" />
      <map ref="defaultCacheMap" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout" />
      <map ref="org.hibernate.cache.UpdateTimestampsCache" />
      <map ref="org.hibernate.cache.StandardQueryCache" />
    </mapSet>
  </objectgridDeployment>
</deploymentPolicy>
```

Nota: Os mapas org.hibernate.cache.UpdateTimestampsCache, org.hibernate.cache.StandardQueryCache e defaultCacheMap são necessários.

Sistema Externo para um Cache com o Tipo REMOTE do ObjectGrid

É necessário configurar um sistema eXtreme Scale externo se desejar configurar um cache com um tipo de ObjectGrid REMOTE. É necessário ambos os arquivos XML de configuração ObjectGrid e ObjectGridDeployment, que se baseiam no arquivo `persistence.xml`, para configurar um sistema externo. Os arquivos XML de configuração Hibernate de ObjectGrid e ObjectGridDeployment descritos na seção de exemplo de arquivo XML Hibernate do ObjectGrid também podem ser usados para configurar um sistema eXtreme Scale externo.

Um sistema externo do ObjectGrid possui processos de serviço do catálogo e do servidor ObjectGrid. É necessário iniciar um serviço de catálogos antes de iniciar servidores de contêiner. Consulte os detalhes sobre como iniciar servidores eXtreme Scale independentes e processos de contêiner no *Guia de Administração* para obter mais informações.

Resolução de Problemas

1. **`CacheException: Falha ao obter o servidor ObjectGrid`**

   Com um ObjectGridType EMBEDDED ou EMBEDDED_PARTITION, o cache tentará obter uma instância do servidor do tempo de execução do eXtreme Scale. Em um ambiente Java Platform, Standard Edition, um servidor eXtreme Scale com serviço de catálogo integrado será iniciado. O serviço de catálogo integrado tentará atender na porta 2809. Se esta porta estiver sendo utilizada por outro processo, ocorrerá um erro. Se terminais de serviço de catálogo externo forem especificados, por exemplo, com o arquivo `objectGridServer.properties`, este erro ocorrerá se o nome do host ou a porta for especificado incorretamente.

2. **CacheException: Falha ao obter o REMOTE ObjectGrid para o REMOTE ObjectGrid configurado. objectGridName = [ObjectGridName], Nome da PU = [persistenceUnitName]**

   Este erro ocorre quando o cache falha ao obter um ObjectGrid dos terminais de serviço do catálogo fornecido. Normalmente, o erro é causado por um nome do host ou porta incorreto.

3. **CacheException: Não é possível ter duas PUs [persistenceUnitName_1, persistenceUnitName_2] configuradas com o mesmo ObjectGridName [ObjectGridName] do EMBEDDED ObjectGridType**

   Essa exceção ocorrerá se você tiver uma configuração de muitas unidades de persistência e os caches dessas unidades estiverem forem configurados com o mesmo nome do ObjectGrid e ObjectGridType EMBEDDED. Estas configurações de unidade de persistência podem estar no mesmo arquivo `persistence.xml` ou diferente. É necessário verificar se o nome do ObjectGrid é exclusivo para cada unidade de persistência quando o ObjectGridType for EMBEDDED.

4. **CacheException: REMOTE ObjectGrid [ObjectGridName] não inclui os BackingMaps necessários [mapName_1, mapName_2,...]**

   Com um tipo de ObjectGrid REMOTE, se o ObjectGrid do lado do cliente obtido não tiver BackingMaps de entidade completos para suportar o cache da unidade de persistência, ocorrerá esta exceção. Por exemplo, cinco classes de entidade são listadas na configuração da unidade de persistência, mas o ObjectGrid obtido possui apenas dois BackingMaps. Embora o ObjectGrid obtido possa ter dez BackingMaps, se algum destes cinco BackingMaps de entidade necessários não for localizado nos dez BackingMaps, esta exceção ainda ocorrerá.

Configuração de Plug-in do Cache OpenJPA

O WebSphere eXtreme Scale fornece tanto as implementações DataCache quanto QueryCache para OpenJPA. Em resumo, o cache ObjectGrid do OpenJPA ou o cache do ObjectGrid é um termo comum para as implementações DataCache e QueryCache.

Configurações

O cache do eXtreme Scale é ativado ou desativado para OpenJPA por meio da configurações das propriedades de configuração openjpa.DataCache e openjpa.QueryCache no arquivo `persistence.xml`. A sintaxe para configuração da propriedade é a seguinte:

```
<property name="openjpa.DataCache"
          value="<object_grid_datacache_class(<property>=<value>,...)"/>
<property name="openjpa.QueryCache"
          value="<object_grid_querycache_class(<property>=<value>,...)"/>
```

Tanto DataCache quanto QueryCache pode pegar as propriedades do cache do eXtreme Scale para configurar o cache usado pela unidade de persistência.

Além da configuração do cache do eXtreme Scale, a propriedade openjpa.RemoteCommitProvider deve ser configurada como `sjvm`:

```
<property name="openjpa.RemoteCommitProvider" value="sjvm"/>
```

O valor de tempo limite especificado com a anotação @DataCache para cada classe de entidade é transportado para o BackingMap ao qual cada entidade é armazenada em cache. Porém, o valor de nome especificado com a anotação

@DataCache é ignorado pelo cache do eXtreme Scale. O nome da classe de entidade completamente qualificado é o nome do mapa do cache.

Os métodos pin e unpin de OpenJPA StoreCache e QueryCache não são suportados e não desempenham uma função.

É possível executar a propriedade ObjectGridName, a propriedade ObjectGridType e outras propriedades relacionadas à política de implementação simples na lista de propriedades da classe do cache ObjectGrid para customizar a configuração do cache. Este é um exemplo:

```
<property name="openjpa.DataCache"
          value="com.ibm.websphere.objectgrid.openjpa.ObjectGridDataCache(
          ObjectGridName=BasicTestObjectGrid,ObjectGridType=EMBEDDED,
          maxNumberOfReplicas=4)"/>
<property name="openjpa.QueryCache"
          value="com.ibm.websphere.objectgrid.openjpa.ObjectGridQueryCache()" />
<property name="openjpa.RemoteCommitProvider" value="sjvm"/>
```

As configurações DataCache e QueryCache são independentes uma da outra. É possível ativar qualquer uma das configurações. Entretanto, se as duas estiverem ativadas, a configuração do QueryCache utilizará a mesma configuração que a do DataCache e suas propriedades de configuração serão descartadas.

Customizando a Configuração do Cache OpenJPA com XML

Para a maioria dos cenários, a configuração das propriedades do cache do eXtreme Scale deve ser suficiente. Para customizar ainda mais o ObjectGrid usado pelo cache, é possível fornecer os arquivos XML de configuração OpenJPA ObjectGrid no diretório META-INF da mesma forma que o arquivo `persistence.xml`. Durante a inicialização do cache, o cache do ObjectGrid tentará localizar estes arquivos XML e processá-los, se forem localizados.

Há três tipos de arquivos XML de configuração OpenJPA ObjectGrid: `openjpa-objectGrid.xml` (configuração ObjectGrid), `openjpa-objectGridDeployment.xml` (política de implementação) e `openjpa-objectGrid-client-override.xml` (configuração de substituição do ObjectGrid cliente). Dependendo do tipo de ObjectGrid configurado, é possível fornecer algum destes três arquivos XML para customizar o ObjectGrid.

Para ambos os tipos EMBEDDED e EMBEDDED_PARTITION, é possível fornecer qualquer um dos três arquivos XML para customizar o ObjectGrid, a política de implementação e a configuração de substituição do ObjectGrid cliente.

Para um REMOTE ObjectGrid, o cache ObjectGrid não cria um ObjectGrid dinâmico. Em vez disso, ele obtém um ObjectGrid do lado do cliente a partir do serviço do catálogo. É possível fornecer apenas o arquivo `openjpa-objectGrid-client-override.xml` para customizar a configuração de substituição do ObjectGrid cliente.

1. **Configuração do ObjectGrid**: Use o arquivo `META-INF/openjpa-objectGrid.xml`. Este arquivo é utilizado para customizar a configuração do ObjectGrid para os tipos EMBEDDED e EMBEDDED_PARTITION. Com o tipo REMOTE, este arquivo é ignorado. Por padrão, cada classe de entidade é mapeada para sua própria configuração do BackingMap denominada como um nome de classe de entidade na configuração do ObjectGrid. Por exemplo, a classe de entidade com.mycompany.Employee é mapeada para o BackingMap com.mycompany.Employee. A configuração padrão do BackingMap é `readOnly="false"`, `copyKey="false"`, `lockStrategy="NONE"` e

copyMode="NO_COPY". É possível customizar alguns BackingMaps com uma configuração escolhida. A palavra-chave reservada ALL_ENTITY_MAPS pode ser utilizada para representar todos os mapas, exceto outros mapas customizados listados no arquivo openjpa-objectGrid.xml. Os BackingMaps que não estiverem listados neste arquivo openjpa-objectGrid.xml utilizarão a configuração padrão. Se os BackingMaps customizados não especificarem o atributo ou as propriedades de BackingMaps e estes atributos forem especificados na configuração padrão, os valores de atributo da configuração padrão serão aplicados. Por exemplo, se uma classe de entidade for anotada com timeToLive=30, a configuração padrão do BackingMap para essa entidade terá timeToLive=30. Se o arquivo openjpa-objectGrid.xml customizado também incluir esse BackingMap mas não especificar o valor timeToLive, o BackingMap customizado terá timeToLive=30 por padrão. O arquivo openjpa-objectGrid.xml pretende substituir ou estender a configuração padrão.

2. **Configuração do ObjectGridDeployment**: Use o arquivo META-INF/openjpa-objectGridDeployment.xml. Este arquivo é utilizado para customizar a política de implementação. Ao customizar a política de implementação, se o arquivo openjpa-objectGridDeployment.xml for fornecido, a política de implementação padrão será descartada. Todos os valores de atributo da política de implementação são fornecidos pelo arquivo openjpa-objectGridDeployment.xml.
3. **Configuração de substituição do ObjectGrid do cliente**: Use o arquivo META-INF/openjpa-objectGrid-client-override.xml. Este arquivo é utilizado para customizar um ObjectGrid do lado do cliente. Por padrão, o cache do ObjectGrid aplica uma configuração do ObjectGrid de substituição do cliente que desativa um cache local. Se um aplicativo requerer um cache perto, ele pode fornecer este arquivo e especificar numberOfBuckets="xxx". A substituição do cliente padrão desativa o cache perto ao configurar numberOfBuckets="0". O cache perto pode estar ativo ao reconfigurar numberOfBuckets com um valor maior que 0 com o arquivo openjpa-objectGrid-client-override.xml. O arquivo openjpa-objectGrid-client-override.xml funciona da mesma forma que o arquivo openjpa-objectGrid.xml: Ele substitui ou estende a configuração padrão de substituição do ObjectGrid cliente.

Exemplos do Arquivo XML OpenJPA do ObjectGrid

Os arquivos XML OpenJPA do ObjectGrid devem ser criados com base na configuração da unidade de persistência.

A seguir há um arquivo persistence.xml de exemplo que representa a configuração de uma unidade de persistência:

persistence.xml

```
<persistence xmlns="http://java.sun.com/xml/ns/persistence"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  version="1.0">
  <persistence-unit name="AnnuityGrid">
    <provider>org.apache.openjpa.persistence.PersistenceProviderImpl</provider>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityPersistebleObject</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.FixedAnnuity</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.EquityAnnuity</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Person</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityHolder</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact</class>
    <class>com.ibm.wssvt.acme.annuity.common.bean.jpa.Address</class>
    <exclude-unlisted-classes>true</exclude-unlisted-classes>

    <properties>
    <!-- Database setting -->


    <!--  enable cache -->
```

```
      <property name="openjpa.DataCache"
        value="com.ibm.websphere.objectgrid.openjpa.ObjectGridDataCache(objectGridName=Annuity,
              objectGridType=EMBEDDED, maxNumberOfReplicas=4)" />
      <property name="openjpa.RemoteCommitProvider" value="sjvm" />
      <property name="openjpa.QueryCache"
                value="com.ibm.websphere.objectgrid.openjpa.ObjectGridQueryCache()" />
    </properties>
  </persistence-unit>
</persistence>
```

A seguir há o arquivo `openjpa-objectGrid.xml` que corresponde ao arquivo `persistence.xml`:

openjpa-objectGrid.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
  xmlns="http://ibm.com/ws/objectgrid/config">
  <objectGrids>
    <objectGrid name="Annuity">
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Address" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Address" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityPersistebleObject"
                  readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityPersistebleObject" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider" />
      <backingMap name="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" evictionTriggers="MEMORY_USAGE_THRESHOLD"
                  pluginCollectionRef="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout" />
      <backingMap name="ObjectGridQueryCache" readOnly="false" copyKey="false"
                  lockStrategy="NONE" copyMode="NO_COPY" pluginCollectionRef="ObjectGridQueryCache"
                  evictionTriggers="MEMORY_USAGE_THRESHOLD"  />
    </objectGrid>
  </objectGrids>
  <backingMapPluginCollections>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Address">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection
            id="com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityPersistebleObject">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
```

```
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout">
      <bean id="ObjectTransformer"
            className="com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer" />
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
    <backingMapPluginCollection id="ObjectGridQueryCache">
      <bean id="MapIndexPlugin" className="com.ibm.websphere.objectgrid.plugins.index.HashIndex" >
          <property name="Name" type="java.lang.String"
                    value="QueryCacheKeyIndex" description="name of index"/>
          <property name="POJOKeyIndex" type="boolean" value="true" description="POJO Key Index "/>
      </bean>
      <bean id="Evictor" className="com.ibm.websphere.objectgrid.plugins.builtins.LRUEvictor">
      </bean>
    </backingMapPluginCollection>
  </backingMapPluginCollections>
</objectGridConfig>
```

Nota:

1. Cada entidade é mapeada para um BackingMap denominado como o nome completo da classe de entidade.
2. Se as classes de entidade estão em uma hierarquia de herança, as classes-filha serão mapeadas para o BackingMap pai. A hierarquia de herança compartilha um único BackingMap.
3. O mapa ObjectGridQueryCache é necessário para suportar QueryCache.
4. O backingMapPluginCollection para cada mapa de entidade deve ter o ObjectTransformer utilizando a classe com.ibm.ws.objectgrid.openjpa.ObjectGridPCDataObjectTransformer.
5. O backingMapPluginCollection para o mapa ObjectGridQueryCache deve ter o índice de chaves denominado como QueryCacheKeyIndex, conforme mostrado no modelo.
6. O evictor é opcional para cada mapa.

A seguir há o arquivo `openjpa-objectGridDeployment.xml` que corresponde ao arquivo `persistence.xml`:

openjpa-objectGridDeployment.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
  xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">
  <objectgridDeployment objectgridName="Annuity">
    <mapSet name="MAPSET_Annuity" numberOfPartitions="1" numInitialContainers="1"
            minSyncReplicas="0" maxSyncReplicas="4" maxAsyncReplicas="0"
            replicaReadEnabled="true">
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Annuity" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Address" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payor" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Person" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Contact" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.AnnuityPersistebleObject" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Rider" />
      <map ref="com.ibm.wssvt.acme.annuity.common.bean.jpa.Payout" />
      <map ref="ObjectGridQueryCache" />
    </mapSet>
  </objectgridDeployment>
</deploymentPolicy>
```

Nota: O mapa ObjectGridQueryCache é necessário para suportar QueryCache.

Sistema Externo para um Cache com o Tipo REMOTE do ObjectGrid

É necessário configurar um sistema externo se desejar configurar um cache com um tipo de ObjectGrid REMOTE. Você precisa de ambos os arquivos XML de configuração ObjectGrid e ObjectGridDeployment, que se baseiam no arquivo

persistence.xml, para configurar um sistema externo. Os arquivos XML de configuração OpenJPA ObjectGrid e ObjectGridDeployment, descritos na seção de exemplos de arquivo XML OpenJPA do ObjectGrid, também podem ser usados para configurar um sistema eXtreme Scale externo.

Um sistema eXtreme Scale externo possui ambos os processos de serviço do catálogo e do servidor de contêineres. É necessário iniciar o servidor de catálogos antes de iniciar servidores de contêineres.

Resolução de Problemas

1. **CacheException: Falha ao obter o servidor ObjectGrid**

   Com um ObjectGridType EMBEDDED ou EMBEDDED_PARTITION, o cache do eXtreme Scale tenta obter uma instância do servidor do tempo de execução. Em um ambiente Java Platform, Standard Edition, um servidor eXtreme Scale com serviço de catálogo integrado é iniciado. O serviço de catálogo integrado tentará atender na porta 2809. Se esta porta estiver sendo utilizada por outro processo, ocorrerá um erro. Se terminais de serviço de catálogo externo forem especificados, por exemplo, com o arquivo objectGridServer.properties, este erro ocorrerá se o nome do host ou a porta for especificado incorretamente.

2. **CacheException: Falha ao obter o REMOTE ObjectGrid para o REMOTE ObjectGrid configurado. objectGridName = [ObjectGridName], Nome da PU = [persistenceUnitName]**

   Este erro ocorre quando o cache não consegue obter um ObjectGrid dos terminais de serviço de catálogo fornecidos. Normalmente, o erro é causado por um nome do host ou porta incorreto.

3. **CacheException: Não é possível ter duas PUs [persistenceUnitName_1, persistenceUnitName_2] configuradas com o mesmo ObjectGridName [ObjectGridName] do EMBEDDED ObjectGridType**

   Essa exceção ocorrerá se você tiver uma configuração de muitas unidades de persistência e os caches do eXtreme Scale destas unidades forem configurados com o mesmo nome de ObjectGrid e ObjectGridType EMBEDDED. Estas configurações de unidade de persistência podem estar no mesmo arquivo persistence.xml ou diferente. É necessário verificar se o nome do ObjectGrid é exclusivo para cada unidade de persistência quando o ObjectGridType for EMBEDDED.

4. **CacheException: REMOTE ObjectGrid [ObjectGridName] não inclui os BackingMaps necessários [mapName_1, mapName_2,...]**

   Com um tipo de ObjectGrid REMOTE, se o ObjectGrid do lado do cliente obtido não tiver BackingMaps de entidade completos para suportar o cache da unidade de persistência, ocorrerá esta exceção. Por exemplo, cinco classes de entidade são listadas na configuração da unidade de persistência, mas o ObjectGrid obtido possui apenas dois BackingMaps. Embora o ObjectGrid obtido possa ter dez BackingMaps, se algum destes cinco BackingMaps de entidade necessários não for localizado nos dez BackingMaps, esta exceção ainda ocorrerá.

Nota: O cache OpenJPA do eXtreme Scale alterou o formato de dados para aumentar o desempenho. Todos os sistemas que estão hospedando aplicativos OpenJPA que são configurados com o eXtreme Scale como um cache L2 devem ser interrompidos antes da migração para o WebSphere eXtreme Scale Versão 7.0.

Configurando os Gerenciadores de Sessão HTTP

O gerenciador de sessões HTTP fornece recursos de replicação de sessão para um aplicativo associado. O gerenciador de replicação de sessão trabalha com o contêiner de Web para o gerenciador de sessões para criar sessões HTTP e gerenciar os ciclos de vida de sessões HTTP que estão associadas ao aplicativo.

Arquivos XML para Configuração do Gerenciador de Sessões HTTP

Quando é iniciado um servidor de contêiner que armazena dados da sessão HTTP, é possível usar os arquivos XML padrão ou pode especificar arquivos XML customizados que criam nomes específicos de ObjectGrid, número de réplicas e assim por diante.

Local dos Arquivos de Amostra

Esses arquivos XML são compactados em *<extremescale>*/ObjectGrid/session/samples para uma instalação independente ou *<WAS_install_root>*/optionalLibraries/ObjectGrid/session/samples para o WebSphere eXtreme Scale instalado em uma célula do WebSphere Application Server.

Pacote XML Integrado

Se você estiver configurando um cenário integrado, que significa que o servidor de contêiner é iniciado na camada do contêiner de Web, os arquivos objectGrid.xml e objectGridDeployment.xml serão fornecidos por padrão. É possível atualizar esses arquivos para customizar o comportamento do gerenciador de sessões HTTP.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd" xmlns="http://ibm.com/ws/objectgrid/config">
 <objectGrids>
  <objectGrid name="session">
   <bean id="ObjectGridEventListener" className="com.ibm.ws.xs.sessionmanager.SessionHandleManager"/>
   <backingMap name="objectgridSessionMetadata" pluginCollectionRef="objectgridSessionMetadata" readOnly="false"
     lockStrategy="PESSIMISTIC" ttlEvictorType="LAST_ACCESS_TIME" timeToLive="3600" copyMode="NO_COPY"/>
   <backingMap name="objectgridSessionAttribute.*" template="true" readOnly="false" lockStrategy="PESSIMISTIC"
     ttlEvictorType="NONE" copyMode="NO_COPY"/>
       <backingMap name="objectgridSessionTTL.*" template="true" readOnly="false" lockStrategy="PESSIMISTIC"
     ttlEvictorType="LAST_ACCESS_TIME" timeToLive="3600" copyMode="NO_COPY"/>
  </objectGrid>
 </objectGrids>
    <backingMapPluginCollections>
        <backingMapPluginCollection id="objectgridSessionMetadata">
            <bean id="MapEventListener" className="com.ibm.ws.xs.sessionmanager.MetadataMapListener"/>
        </backingMapPluginCollection>
    </backingMapPluginCollections>
</objectGridConfig>
```

Valores que é possível alterar:

- **Atributo de nome de ObjectGrid:** Deve ser igual à propridade de objectGridName no arquivo `splicer.properties` que é usado para conectar o aplicativo da Web e o atributo objectgridName no arquivo `objectGridDeployment.xml`. Se você tiver vários aplicativos e desejar que os dados da sessão sejam armazenados em grades diferentes, esses aplicativos deverão ter valores diferentes de atributo de nome de ObjectGrid. O nome do ObjectGrid é a única coisa que pode ser alterada neste arquivo.

Valores que não é possível alterar:

- **ObjectGridEventListener:** A linha ObjectGridEventListener não pode ser alterada e é usada internamente.
- **objectgridSessionMetadata**: A linha objectgridSessionMetadata refere-se ao mapa no qual os metadados da sessão HTTP são armazenados. Há uma entrada para cada sessão HTTP armazenada na grade neste mapa.
- **objectgridSessionAttribute.*** A linha objectgridSessionAttribute.* define um mapa dinâmico que é usado para criar o mapa no qual os atributos de sessão HTTP são armazenados quando o parâmetro **fragmentedSession** é configurado para `true` no arquivo `splicer.properties` usado para conectar o aplicativo da Web. Este mapa dinâmico é chamado objectgridSessionAttribute. Outro mapa é criado com base neste modelo chamado objectgridSessionAttributeEvicted, que armazena sessões que atingiram o tempo limite, mas o contêiner de Web não foi invalidado.
- A linha **MetadataMapListener** é interna e não pode ser modificada

Figura 14. objectGrid.xml file

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

 <objectgridDeployment objectgridName="session">
  <mapSet name="sessionMapSet" numberOfPartitions="5" minSyncReplicas="0" maxSyncReplicas="0"
    maxAsyncReplicas="1" developmentMode="false" placementStrategy="PER_CONTAINER">
             <map ref="objectgridSessionMetadata"/>
             <map ref="objectgridSessionAttribute.*"/>
             <map ref="objectgridSessionTTL.*"/>
    </mapSet>
   </objectgridDeployment>
</deploymentPolicy>
```

Valores que é possível alterar:

- **Atributo objectgridName**: Deve ter o mesmo nome que a propriedade objectGridName no arquivo `splicer.properties` que é usado para conectar o aplicativo da Web e o atributo de nome de ObjectGrid no arquivo `objectGrid.xml`.
- **Atributos de elemento mapSet**: É possível alterar todo o mapSet. As propriedades podem ser alteradas, exceto para o atributo placementStrategy.

  Name Pode ser atualizado para qualquer valor.

  numberOfPartitions
  Especifica o número de partições primárias que são iniciadas em cada servidor que está hospedando o aplicativo da Web. Conforme você inclui partições, os dados se tornam mais espalhados no caso de um failover. O valor padrão é de 5 partições e é bom para a maioria dos aplicativos.

  minSyncReplicas, maxSyncReplicas e maxAsyncReplicas
  Especifica o número e o tipo de réplicas que armazenam os dados da sessão HTTP. O padrão é 1 réplica assíncrona, que é bom para a maioria dos aplicativos. A replicação síncrona ocorre durante o caminho do pedido, que pode aumentar os tempos de resposta para seu aplicativo da Web.

  developmentMode
  Informa o serviço de posicionamento do eXtreme Scale se os shards de réplica para uma partição podem ser posicionados no mesmo nó que seu shard primário. É possível configurar o valor para true em um ambiente de desenvolvimento, mas desative esta função em um ambiente de produção, pois uma falha do nó pode causar a perda de dados da sessão.

  placementStrategy
  Não altere o valor deste atributo.
- O restante do arquivo faz referência aos mesmos nomes do mapa que no objectGrid.xml. Esses nomes não podem ser alterados.

Valores que não podem ser alterados:

- O atributo placementStrategy no elemento mapSet.

Figura 15. objectGridDeployment.xml file

Pacote XML Remoto

Quando estiver usando o modo remoto, no qual os contêineres são executados como processos independentes, você deve usar os arquivos `objectGridStandAlone.xml` e `objectGridDeploymentStandAlone.xml` para iniciar os processos. É possível atualizar esses arquivos para modificar a configuração.

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
 <objectGrids>
  <objectGrid name="session">
   <bean id="ObjectGridEventListener" className="com.ibm.ws.xs.sessionmanager.SessionHandleManager"/>
          <backingMap name="objectgridSessionMetadata" pluginCollectionRef="objectgridSessionMetadata"
       readOnly="false" lockStrategy="PESSIMISTIC" ttlEvictorType="LAST_ACCESS_TIME" timeToLive="3600"
       copyMode="COPY_TO_BYTES"/>
          <backingMap name="objectgridSessionAttribute.*" template="true" readOnly="false" lockStrategy="PESSIMISTIC"
       ttlEvictorType="NONE" copyMode="COPY_TO_BYTES"/>
          <backingMap name="objectgridSessionTTL.*" template="true" readOnly="false" lockStrategy="PESSIMISTIC"
       ttlEvictorType="LAST_ACCESS_TIME" timeToLive="3600" copyMode="COPY_TO_BYTES"/>
  </objectGrid>
 </objectGrids>
    <backingMapPluginCollections>
        <backingMapPluginCollection id="objectgridSessionMetadata">
            <bean id="MapEventListener" className="com.ibm.ws.xs.sessionmanager.MetadataMapListener"/>
        </backingMapPluginCollection>
    </backingMapPluginCollections>
</objectGridConfig>
```

Valores que é possível alterar:

- **Atributo objectgridName**: Pode ser alterado, mas deve ser o mesmo que a propriedade objectGridName no arquivo `splicer.properties` que é usado para conectar o aplicativo da Web e o atributo objectgridName no arquivo `objectGridDeploymentStandAlone.xml`. Se você tiver vários aplicativos e desejar que os dados da sessão sejam armazenados em grades diferentes, esses aplicativos deverão ter nomes de grade diferentes. O nome da grade é a única coisa que pode ser alterada neste arquivo.

Valores que não podem ser alterados:

- **ObjectGridEventListener:** A linha ObjectGridEventListener não pode ser alterada e é usada internamente.
- **objectgridSessionMetadata**: A linha objectgridSessionMetadata refere-se ao mapa no qual os metadados da sessão HTTP são armazenados. Há uma entrada para cada sessão HTTP armazenada na grade neste mapa.
- **objectgridSessionAttribute.*** A linha objectgridSessionAttribute.* define um mapa dinâmico que é usado para criar o mapa no qual os atributos de sessão HTTP são armazenados quando o parâmetro **fragmentedSession** é configurado para `true` no arquivo `splicer.properties` usado para conectar o aplicativo da Web. Este mapa dinâmico é chamado objectgridSessionAttribute. Outro mapa é criado com base neste modelo chamado objectgridSessionAttributeEvicted, que armazena sessões que atingiram o tempo limite, mas o contêiner de Web não foi invalidado.
- A linha **MetadataMapListener** é interna e não pode ser modificada

Figura 16. objectGridStandAlone.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

 <objectgridDeployment objectgridName="session">
  <mapSet name="sessionMapSet" numberOfPartitions="5" minSyncReplicas="0" maxSyncReplicas="0"
   maxAsyncReplicas="1" developmentMode="false" placementStrategy="PER_CONTAINER">
            <map ref="objectgridSessionMetadata"/>
            <map ref="objectgridSessionAttribute.*"/>
            <map ref="objectgridSessionTTL.*"/>
     </mapSet>
    </objectgridDeployment>
</deploymentPolicy>
```

Valores que é possível alterar:

- **Atributo objectgridName**: Deve ter o mesmo nome que a propriedade objectGridName no arquivo `splicer.properties` que é usado para conectar o aplicativo da Web e o atributo de nome de ObjectGrid no arquivo `objectGrid.xml`.
- **Atributos de elemento mapSet**: É possível alterar todo o mapSet. As propriedades podem ser alteradas, exceto para o atributo placementStrategy.

  Name Pode ser atualizado para qualquer valor.

  numberOfPartitions
  Especifica o número de partições primárias que são iniciadas em cada servidor que está hospedando o aplicativo da Web. Conforme você inclui partições, os dados se tornam mais espalhados no caso de um failover. O valor padrão é de 5 partições e é bom para a maioria dos aplicativos.

  minSyncReplicas, maxSyncReplicas e maxAsyncReplicas
  Especifica o número e o tipo de réplicas que armazenam os dados da sessão HTTP. O padrão é 1 réplica assíncrona, que é bom para a maioria dos aplicativos. A replicação síncrona ocorre durante o caminho do pedido, que pode aumentar os tempos de resposta para seu aplicativo da Web.

  developmentMode
  Informa o serviço de posicionamento do eXtreme Scale se os shards de réplica para uma partição podem ser posicionados no mesmo nó que seu shard primário. É possível configurar o valor para true em um ambiente de desenvolvimento, mas desative esta função em um ambiente de produção, pois uma falha do nó pode causar a perda de dados da sessão.

  placementStrategy
  Não altere o valor deste atributo.
- O restante do arquivo faz referência aos mesmos nomes do mapa que no arquivo `objectGrid.xml`. Esses nomes não podem ser alterados.

Valores que não podem ser alterados:

- O atributo placementStrategy no elemento mapSet.

Figura 17. objectGridDeploymentStandAlone.xml:

Configurando o Gerenciador de Sessões HTTP com WebSphere Application Server

Embora o WebSphere Application Server forneça a função de gerenciamento de sessões, este suporte não escala sob carregamentos de pedido extremos. O WebSphere eXtreme Scale vem com uma implementação de gerenciamento de sessões que fornece opções de replicação de sessão, alta disponibilidade, melhor escalabilidade e configuração mais firme.

Antes de Iniciar

- O WebSphere eXtreme Scale deve ser instalado na célula do seu WebSphere Application Server ou WebSphere Application Server Network Deployment para usar o gerenciador de sessões do eXtreme Scale. Consulte “Integrando o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client com o WebSphere Application Server” na página 22 para obter mais informações.

Sobre Esta Tarefa

O gerenciador de sessão HTTP WebSphere eXtreme Scale suporta servidores integrados e remotos para armazenamento em cache.

Cenário Integrado

No cenário integrado, os servidores WebSphere eXtreme Scale são colocados no mesmo processo onde os servlets são executados. O gerenciador de sessões pode ser comunicar diretamente com a instância do ObjectGrid local, evitando atrasos de rede caros.

Se estiver usando o WebSphere Application Server, coloque os arquivos `objectgridRoot/session/samples/objectGrid.xml` e `objectgridRoot/session/samples/objectGridDeployment.xml` fornecidos nos diretórios `META-INF` de seus arquivos WAR (Web Archive). O eXtreme Scale detecta automaticamente esses arquivos quando o aplicativo é iniciado e inicia automaticamente os contêineres do eXtreme Scale no mesmo processo do gerenciador de sessões.

É possível modificar o arquivo `objectGridDeployment.xml` dependendo se você deseja usar a replicação síncrona ou assíncrona e de quantas réplicas você deseja configurar.

Cenário dos Servidores Remotos

No cenário de servidores remotos, os servidores de contêiner são executados em processos diferentes dos servlets. O gerenciador de sessões se comunica com um servidor de contêiner remoto. Para usar um servidor de contêiner remoto, conectado à rede, o gerenciador de sessões deve ser configurado com os nomes de host e números de porta do domínio do serviço de catálogo. O gerenciador de sessões, então, usa uma conexão do cliente do eXtreme Scale para se comunicar com o servidor de catálogos e com os servidores de contêiner.

Se os servidores de contêiner forem iniciados em processos independentes, inicie os contêineres do eXtreme Scale usando os arquivos `objectGridStandAlone.xml` e `objectGridDeploymentStandAlone.xml` que são fornecidos no diretório de amostras do gerenciador de sessões.

Procedimento

1. Una seu aplicativo para que ele possa utilizar o gerenciador de sessões. Para usar o gerenciador de sessões, é necessário incluir as declarações de filtro apropriadas nos descritores de implementação da Web para o aplicativo. Além disso, os parâmetros de configuração do gerenciador de sessões são passados no gerenciador de sessões no formato de parâmetros de inicialização de contexto do servlet nos descritores de implementação. Há várias maneiras pelas quais é possível introduzir essas informações no seu aplicativo:

   - 7.1+ **No console administrativo do WebSphere Application Server**

É possível configurar seu aplicativo para usar o gerenciador de sessões HTTP do WebSphere eXtreme Scale quando instalar seu aplicativo. Pode também editar o aplicativo ou a configuração do servidor para usar o gerenciador de sessões HTTP do WebSphere eXtreme Scale. Consulte "Criando Persistência de Sessão para um grade de dados" na página 261 para obter mais informações.

Nota: Esta opção funcionará apenas se todos os nós que estão executando o aplicativo contiverem o arquivo `splicer.properties` no mesmo caminho. Para ambientes mistos que contêm nós Windows e UNIX, esta opção não é possível, portanto, você deve conectar o aplicativo manualmente.

- **Opção de conexão automática**

  Não é necessário conectar seus aplicativos manualmente quando o aplicativo está em execução no WebSphere Application Server ou no WebSphere Application Server Network Deployment. É possível incluir uma propriedade customizada em uma célula ou em um servidor que afetará todos os aplicativos da Web nesse escopo. O nome da propriedade é `com.ibm.websphere.xs.sessionFilterProps`, e você deve configurar seu valor para o local do arquivo `splicer.properties` que seu aplicativo requer.

  A seguir está um caminho de exemplo para o local de um arquivo: `/opt/splicer.properties`.

  – **Para especificar todos os aplicativos da Web em uma célula como conectados, use o console administrativo:**

    Vá para **Administração do sistema** → **Célula** → **Propriedades customizadas** e inclua a propriedade aí.

  – **Para especificar todos os aplicativos da Web em um determinado servidor de aplicativos, use o console administrativo:**

    Vá para **Servidor de aplicativos** → **<*server_name*>** → **Administração** → **Propriedades customizadas** e inclua a propriedade aí.

  Os escopos da célula e do servidor são os únicos escopos disponíveis e são os únicos disponíveis quando estiver executando em um gerenciador de implementação. Se você precisar de um escopo diferente, será necessário fazer a junção manualmente dos seus aplicativos da Web.

  Nota: A opção de conexão automática funcionará apenas se todos os nós que estão executando o aplicativo contiverem o arquivo `splicer.properties` no mesmo caminho. Para ambientes mistos que contêm nós Windows e UNIX, esta opção não é possível, portanto, você deve conectar o aplicativo manualmente.

- **Script addObjectGridFilter**

  Utilize o script de linha de comandos fornecido junto com o eXtreme Scale para unir um aplicativo com declarações de filtro e configuração no formato de parâmetros de inicialização de contexto do servlet. Esse script, `objectgridRoot/bin/addObjectGridFilter.sh` ou `objectgridRoot/session/bin/addObjectGridFilter.bat`, utiliza dois parâmetros: o aplicativo (caminho absoluto para o arquivo corporativo) que precisa da junção, e o caminho absoluto para o arquivo de propriedades que contém várias propriedades de configuração. O formato de uso deste script é o seguinte:

  Windows

  ```
  addObjectGridFilter.bat [location of ear file] [location of properties file]
  ```

  UNIX

```
addObjectGridFilter.sh [location of ear file] [location of properties file]
```

UNIX **Exemplo utilizando o eXtreme Scale instalado no WebSphere Application Server no Unix:**

a. `cd objectgridRoot/optionalLibraries/ObjectGrid/session/bin`

b. `addObjectGridFilter.sh /tmp/mySessionTest.ear wasRoot/optionalLibraries/ObjectGrid/session/samples/splicer.properties`

UNIX **Exemplo utilizando o eXtreme Scale instalado em um diretório independente no Unix:**

a. `cd objectgridRoot/session/bin`

b. `addObjectGridFilter.sh /tmp/mySessionTest.ear objectgridRoot/session/samples/splicer.properties`

O filtro do servlet que é dividido mantém os padrões para os valores de configuração. É possível substituir estes valores-padrão com opções de configuração especificados no arquivo de propriedades no segundo argumento. Para obter uma lista dos parâmetros que podem ser utilizados, consulte "Parâmetros de inicialização do contexto do servlet" na página 267.

É possível modificar e usar o arquivo `splicer.properties` de amostra que é fornecido com a instalação do eXtreme Scale. Também é possível usar o script addObjectGridServlets, que insere o gerenciador de sessões estendendo cada servlet. No entanto, o script recomendado é o addObjectGridFilter.

- **Script de construção Ant**

  O WebSphere eXtreme Scale é fornecido com um arquivo `build.xml` que pode ser utilizado pelo Apache Ant, que está incluído na pasta `wasRoot/bin` de uma instalação do WebSphere Application Server. O `build.xml` pode ser modificado para alterar as propriedades de configuração do gerenciador de sessões. As propriedades de configuração são idênticas aos nomes de propriedades no arquivo `splicer.properties`. Depois que `build.xml` tiver sido modificado, chame o processo Ant executando:

  – UNIX `ant.sh`, `ws_ant.sh`

  – Windows `ant.bat`, `ws_ant.bat`

  (UNIX) ou (Windows).

- **Atualize o descritor da Web manualmente**

  Edite o arquivo `web.xml` que é empacotado com o aplicativo da Web para incorporar a declaração do filtro, seu mapeamento de servlet e os parâmetros de inicialização de contexto do servlet. Você não deve utilizar este método porque ele é propenso a erros.

Para obter uma lista dos parâmetros que podem ser utilizados, consulte "Parâmetros de inicialização do contexto do servlet" na página 267.

2. Implemente o aplicativo. Implemente o aplicativo com seu conjunto normal de etapas para um servidor ou cluster. Após implementar o aplicativo, é possível iniciar o aplicativo.
3. Acesse o aplicativo. É possível acessar o aplicativo, que interage com o gerenciador de sessões e o WebSphere eXtreme Scale.

O que Fazer Depois

É possível alterar a maioria dos atributos de conexão para o gerenciador de sessões quando você instrumenta seu aplicativo para utilizar o gerenciador de sessões. Esses atributos incluem: replicação síncrona ou assíncrona, comprimento do ID de sessão, tamanho da tabela de sessão na memória e assim por diante. Não

considerando os atributos que podem ser alterados no momento da instrumentação do aplicativo, os únicos outros atributos de configuração que podem ser alterados após a implementação do aplicativo são os atributos que estão relacionados à topologia em cluster do servidor WebSphere eXtreme Scale e a maneira pela qual seus clientes (gerenciadores de sessões) se conectam a eles.

Criando Persistência de Sessão para um grade de dados:

É possível configurar o aplicativo WebSphere Application Server para persistir as sessões para uma grade de dados. Esta grade de dados pode estar em um servidor de contêiner integrado executado no WebSphere Application Server ou pode estar em uma grade de dados remota.

Antes de Iniciar

Antes de alterar a configuração no WebSphere Application Server, você deve ter as seguintes informações:

- O nome da grade de dados da sessão que você deseja usar. Consulte "Configurando o Gerenciador de Sessões HTTP com WebSphere Application Server" na página 257 para obter informações sobre como criar uma grade de dados da sessão.
- Se o serviço de catálogo que você deseja usar para gerenciar suas sessões estiver fora da célula na qual está instalando o aplicativo da sessão, será necessário criar um domínio do serviço de catálogo. Consulte "Criando Domínios do Serviço de Catálogo no WebSphere Application Server" na página 344 para obter mais informações.

Procedimento

- **Para configurar o gerenciamento de sessões quando estiver instalando o aplicativo, conclua as seguintes etapas:**
  1. No console administrativo do WebSphere Application Server, clique em **Aplicativos** → **Novo aplicativo** → **Novo Aplicativo Corporativo**. Escolha o caminho **Detalhado** para criar o aplicativo e concluir as etapas iniciais do assistente.
  2. Na etapa **Configurações de gerenciamento de sessão do eXtreme Scale** do assistente, configure a grade de dados que deseja usar. Escolha a **Grade de dados do eXtreme Scale remota** ou a **Grade de dados do eXtreme Scale integrada**.
     - Para a opção **Grade de dados do eXtreme Scale remota**, escolha o domínio do serviço de catálogo que gerencia a grade de dados da sessão e escolha uma grade de dados na lista de grades de dados da sessão ativa.
     - Para a opção **Grade de dados do eXtreme Scale integrada**, escolha a configuração padrão do ObjectGrid ou especifique o local específico dos arquivos de configuração do ObjectGrid.
  3. Conclua as etapas do assistente para concluir a instalação do aplicativo.

  É possível também instalar o aplicativo com um script wsadmin. No exemplo a seguir, o parâmetro **-SessionManagement** cria a mesma configuração que é possível criar no console administrativo:

  Para a configuração remota de grade de dados eXtreme Scale:

```
AdminApp.install('C:/A.ear', '[  -nopreCompileJSPs -distributeApp
-nouseMetaDataFromBinary -nodeployejb -appname A -edition 8.0
-createMBeansForResources -noreloadEnabled -nodeployws -validateinstall
off -noprocessEmbeddedConfig -filepermission .*\.dll=755#.*\.so=755#.*\.a=755#.*\.sl=755
-buildVersion Unknown -noallowDispatchRemoteInclude -noallowServiceRemoteInclude
```

```
-asyncRequestDispatchType DISABLED -nouseAutoLink -SessionManagement [[true
XSRemoteSessionManagement cs0:!:grid0]]
-MapWebModToVH [[MicroWebApp microwebapp.war,WEB-INF/web.xml default_host] [MicroSipApp
microsipapp.war,WEB-INF/web.xml default_host] [MicroDG1App microdg1app.war,WEB-INF/web.xml
default_host] [MicroDG2App microdg2app.war,WEB-INF/web.xml default_host] [MicroSip2App
microsip2app.war,WEB-INF/web.xml default_host]]]')
```

Para o cenário integrado do eXtreme Scale com configuração padrão:

```
AdminApp.install('C:/A.ear', '[  -nopreCompileJSPs -distributeApp
-nouseMetaDataFromBinary -nodeployejb -appname A -edition 8.0
-createMBeansForResources -noreloadEnabled -nodeployws -validateinstall
off -noprocessEmbeddedConfig -filepermission .*\.dll=755#.*\.so=755#.*\.a=755#.*\.sl=755
-buildVersion Unknown -noallowDispatchRemoteInclude -noallowServiceRemoteInclude
-asyncRequestDispatchType DISABLED -nouseAutoLink -SessionManagement [[true
XSRemoteSessionManagement  :!: :!:default]]  -MapWebModToVH [[MicroWebApp microwebapp.war,
WEB-INF/web.xml default_host] [MicroSipApp
microsipapp.war,WEB-INF/web.xml default_host] [MicroDG1App microdg1app.war,WEB-INF/web.xml
default_host] [MicroDG2App microdg2app.war,WEB-INF/web.xml default_host] [MicroSip2App
microsip2app.war,WEB-INF/web.xml default_host]]]')
```

Para o cenário integrado do eXtreme Scale com uma configuração customizada:

```
AdminApp.install('C:/A.ear', '[  -nopreCompileJSPs -distributeApp
-nouseMetaDataFromBinary -nodeployejb -appname A -edition 8.0
-createMBeansForResources -noreloadEnabled -nodeployws -validateinstall
off -noprocessEmbeddedConfig -filepermission .*\.dll=755#.*\.so=755#.*\.a=755#.*\.sl=755
-buildVersion Unknown -noallowDispatchRemoteInclude -noallowServiceRemoteInclude
-asyncRequestDispatchType DISABLED -nouseAutoLink -SessionManagement [[true
XSRemoteSessionManagement  :!: :!:custom:!:c:\XS\objectgrid.xml:!:c:\XS\objectgriddeployment.xml]]
-MapWebModToVH [[MicroWebApp microwebapp.war,WEB-INF/web.xml default_host] [MicroSipApp
microsipapp.war,WEB-INF/web.xml default_host] [MicroDG1App microdg1app.war,WEB-INF/web.xml
default_host] [MicroDG2App microdg2app.war,WEB-INF/web.xml default_host] [MicroSip2App
microsip2app.war,WEB-INF/web.xml default_host]]]')
```

- **Para configurar o gerenciamento de sessões em um aplicativo existente no console administrativo do WebSphere Application Server:**
  1. No console administrativo do WebSphere Application Server, clique em **Aplicativos** → **Tipos de Aplicativos** → **Aplicativos corporativos do WebSphere** → *application_name* → **Propriedades do módulo da Web** → **Gerenciamento de sessões** → **Configurações de gerenciamento de sessões**.
  2. Atualize os campos para ativar a persistência de sessão para uma grade de dados.

  Também é possível atualizar o aplicativo com um script wsadmin. No exemplo a seguir, o parâmetro **-SessionManagement** cria a mesma configuração que é possível criar no console administrativo:

  Para a configuração remota de grade de dados eXtreme Scale:

  ```
  AdminApp.edit('DefaultApplication','[-SessionManagement[[[true
  XSRemoteSessionManagement cs0:!:grid0]]]')
  ```

  Para o cenário integrado do eXtreme Scale com configuração padrão:

  ```
  AdminApp.edit('DefaultApplication','[-SessionManagement[[[true
  XSEmbeddedSessionManagement :!: :!:default]]]')
  ```

  Para o cenário integrado do eXtreme Scale com uma configuração customizada:

  ```
  AdminApp.edit('DefaultApplication','[-SessionManagement[[[true
  XSEmbeddedSessionManagement :!: :!:
  custom:!:c:\XS\objectgrid.xml:!:c:\XS\objectgriddeployment.xml]]]')
  ```

  Quando você salva as mudanças, o aplicativo usa a grade de dados configurada para a persistência de sessão no dispositivo.

- **Para configurar o gerenciamento de sessões em um servidor existente:**
  1. No console administrativo do WebSphere Application Server, clique em **Servidores** → **Tipos de Servidor** → **WebSphere Application Servers** → *server_name* → **Configurações do contêiner** → **Configurações de gerenciamento de sessões do eXtreme Scale**.
  2. Atualize os campos para ativar a persistência de sessão.

  Também é possível configurar o gerenciamento de sessões em um servidor existente com os seguintes comandos de ferramenta wsadmin:

  Para a configuração remota de grade de dados eXtreme Scale:

```
AdminTask.configureServerSessionManagement('[-nodeName IBM-C77EE220EB6Node01 -serverName server1
-enableSessionManagement true -sessionManagementType XSRemoteSessionManagement -XSRemoteSessionManagement
[-catalogService cs0 -csGridName grid0]]')
```

  Para a configuração integrada do eXtreme Scale:

  – A configuração padrão, se você estiver usando os arquivos XML padrão:

```
AdminTask.configureServerSessionManagement('[-nodeName IBM-C77EE220EB6Node01 -serverName server1
-enableSessionManagement true -sessionManagementType XSEmbeddedSessionManagement
-XSEmbeddedSessionManagement [-embeddedGridType default -objectGridXML -objectGridDeploymentXML ]]')
```

  – A configuração customizada, se você estiver usando arquivos XML customizados:

```
AdminTask.configureServerSessionManagement('[-nodeName IBM-C77EE220EB6Node01 -serverName server1
-enableSessionManagement true -sessionManagementType XSEmbeddedSessionManagement
-XSEmbeddedSessionManagement
[-embeddedGridType custom -objectGridXML c:\XS\objectgrid.xml -objectGridDeploymentXML
c:\XS\objectgriddeployment.xml]]')
```

  Quando você salva as mudanças, o servidor usa a grade de dados configurada para a persistência de sessão com os aplicativos que estão em execução no servidor.
- Se desejar editar outros aspectos da configuração de sessão HTTP, será possível editar o arquivo `splicer.properties`.
  1. No console administrativo do WebSphere Application Server, clique em **Administração do sistema** → **Célula** → **Propriedades customizadas**.
  2. Edite a propriedade customizada `<app_name>,com.ibm.websphere.xs.sessionFilterProps`. O valor desta propriedade customizada fornece o local do arquivo `splicer.properties` a ser editado. O arquivo existe no gerenciador de implementação.
  3. Edite o arquivo `splicer.properties` que está no caminho para a propriedade customizada no perfil do gerenciador de implementação.
  4. Sincronize seus nós de forma que o arquivo `splicer.properties` atualizado seja propagado para seus nós. Clique em **Administração do Sistema** → **Nós**. Escolha os nós nos quais o aplicativo está instalado e clique em **Sincronizar**.

Resultados

Você configurou o gerenciador de sessões HTTP para persistir as sessões para uma grade de dados.

Arquivo splicer.properties:

O arquivo splicer.properties contém todas as opções de configuração para um gerenciador de sessões ObjectGrid baseado em filtro de servlet.

Propriedades do splicer de Amostra

```
# Arquivo propriedades que contém todas as opções de configuração que o
 gerenciador de sessão do ObjectGrid baseado no filtro de servlet pode ser configurado para usar.
#
# Esse arquivo de propriedades pode ser criado para conter todos os
# valores padrão a serem designados a essas definições de configuração, e
# configurações individuais podem ser substituídas com o uso de propriedades
# da Tarefa ANT, caso o arquivo de propriedades seja utilizado junto
# com a Tarefa ANT filtersplicer.

# Um valor de cadeia "REMOTE" ou "EMBEDDED". O padrão é REMOTE.

# Isso é para indicar se devemos iniciar o
# contêiner do WebSphere eXtreme Scale integrado dentro de qualquer
# processo do servidor de aplicativos, incluindo WebSphere,
 WebLogic, JBoss, ToimCat.

objectGridType= REMOTE

# Um valor de cadeia que define o nome da instância de ObjectGrid
# utilizada para um determinado aplicativo da Web.
O nome padrão
# é sessionmanager. A configuração desse parâmetro substitui esse
valor.# objectGridName =

# O Servidor de Catálogos pode ser contatado para você obter uma instância
# de ObjectGrid do lado do cliente.
O valor precisa ter
# o formato "host:port<,host:port>".

# Esta lista pode ser arbitrariamente longa e é utilizada apenas para
 autoinicialização.

# O primeiro endereço viável será utilizado. Isso é opcional dentro do WebSphere
# se catalog.services.cluster estiver configurado

catalogHostPort = host:port<,host:port>

# Um valor de número inteiro define o tempo em segundos entre
# gravações de sessões atualizadas em ObjectGrid. O padrão é 2 segundos.

replicationInterval = 1

# Um valor de número inteiro que define o número de referências de sessão
# mantido na memória. O padrão é
2000.

sessionTableSize =

# Um valor de cadeia "true" ou "false". O padrão é false.
# Ele serve para controlar se vamos armazenar os dados da sessão como
 uma entrada inteira
# ou armazenar cada atributo separadamente

fragmentedSession = false

# Diz ao gerenciador de sessões para considerar o uso de codificação de URL
# (vs. cookies). O padrão, se não estiver configurado, é false

useURLEncoding = false

# Uma cadeia de locais de arquivo para o arquivo xml objectgrid.
# Carregamos nosso arquivo xml integrado automaticamente se ele não estiver
 especificado
# e se objectGridType=EMBEDDED

objectGridXML =
```

```
# Uma cadeia de locais de arquivo para o arquivo xml da política de
implementação do objectGrid.
# Carregamos nosso arquivo xml integrado automaticamente se ele
não estiver especificado
# e se objectGridType=EMBEDDED

objectGridDeploymentXML =

# Uma cadeia de especificação de rastreio do IBM WebSphere,
# útil para todos os servidores de aplicativos, como WebLogic.

traceSpec=

# Uma cadeia de locais de arquivo de rastreio
# útil para todos os servidores de aplicativos, como WebLogic.

traceFile=
```

Usando o WebSphere eXtreme Scale para Gerenciamento de Sessão SIP:

É possível usar o WebSphere eXtreme Scale como um mecanismo de replicação Session Initiation Protocol (SIP) como uma alternativa confiável ao data replication service (DRS) para replicação de sessão de SIP.

Configuração do Gerenciamento de Sessão de SIP

Para usar o WebSphere eXtreme Scale como o mecanismo de replicação SIP, configure a propriedade customizada com.ibm.sip.ha.replicator.type. No console administrativo, selecione **Servidores de aplicativos** → ***my_application_server*** → **contêiner de SIP** → **Propriedades customizadas** para cada servidor para incluir a propriedade customizada. Digite `com.ibm.sip.ha.replicator.type` para o Nome e `OBJECTGRID` para o Valor.

Utilize as seguintes propriedades para customizar o comportamento do ObjectGrid utilizado para armazenar as sessões de SIP. No console administrativo, clique em **Servidores de aplicativos** → ***my_application_server*** → **Contêiner de SIP** → **Propriedades customizadas** para cada servidor para incluir a propriedade customizada. Digite o **Nome** e **Valor**. Cada servidor deve ter o mesmo conjunto de propriedades para funcionar apropriadamente.

Tabela 10. Propriedades Customizadas para Gerenciamento de Sessão SIP com ObjectGrid

Propriedade	Valor	Padrão
com.ibm.sip.ha.replicator.type	`OBJECTGRID`: use ObjectGrid como armazenamento de sessão SIP	
min.synchronous.replicas	Número mínimo de réplicas síncronas	0
max.synchronous.replicas	Número máximo de réplicas síncronas	0
max.asynchronous.replicas	Número máximo de réplicas assíncronas	1
auto.replace.lost.shards	Consulte "Configurando Implementações Distribuídas" na página 163 para obter informações adicionais.	true
development.mode	• `true` - permite que as réplicas sejam ativadas no mesmo nó das primárias • `false` - as réplicas devem estar em um nó diferente do das primárias	false

Configurando o Gerenciador de Sessões HTTP para Vários Servidores de Aplicativos

O WebSphere eXtreme Scale vem com uma implementação de gerenciamento de sessões que substitui o gerenciador de sessões padrão para um contêiner de Web e fornece opções de replicação de sessão, alta disponibilidade, melhor escalabilidade e configuração firme. É possível ativar o gerenciador de replicação de sessão do eXtreme Scale e a inicialização de contêiner de ObjectGrid integrada genérica.

Sobre Esta Tarefa

É possível usar o gerenciador de sessões HTTP com outros servidores de aplicativos que não estejam executando o WebSphere Application Server, como WebSphere Application Server Community Edition. Para configurar outros servidores de aplicativos para usar a grade de dados, você deve conectar seu aplicativo e incorporar arquivos JAR do WebSphere eXtreme Scale a ele.

Procedimento

1. Conecte seu aplicativo para que ele possa usar o gerenciador de sessões. Para usar o gerenciador de sessões, você deve incluir as declarações de filtro apropriadas nos descritores de implementação da Web para o aplicativo. Além disso, os parâmetros de configuração do gerenciador de sessões são passados no gerenciador de sessões no formato de parâmetros de inicialização de contexto do servlet nos descritores de implementação. Há três formas nas quais é possível introduzir essas informações no seu aplicativo:
   - **Script addObjectGridFilter**

     Utilize o script de linha de comandos fornecido junto com o eXtreme Scale para unir um aplicativo com declarações de filtro e configuração no formato de parâmetros de inicialização de contexto do servlet. Este script, `objectgridRoot/session/bin/addObjectGridFilter.sh` ou `objectgridRoot/session/bin/addObjectGridFilter.bat`, utiliza dois parâmetros: o caminho absoluto para o arquivo EAR (Enterprise Archive) do aplicativo que você deseja conectar e o caminho absoluto para o arquivo de propriedades do splicer que contém várias propriedades de configuração. O formato de uso deste script é o seguinte:

     Windows

     ```
     addObjectGridFilter.bat [location of ear file] [location of properties file]
     ```

     UNIX

     ```
     addObjectGridFilter.sh [location of ear file] [location of properties file]
     ```

     UNIX **Exemplo usando o eXtreme Scale instalado em um diretório independente no UNIX:**

     a. `cd objectgridRoot/session/bin`

     b. `addObjectGridFilter.sh /tmp/mySessionTest.ear objectgridRoot/session/samples/splicer.properties`

     O filtro do servlet que é dividido mantém os padrões para os valores de configuração. É possível substituir estes valores-padrão com opções de configuração especificados no arquivo de propriedades no segundo argumento. Para obter uma lista dos parâmetros que podem ser utilizados, consulte "Parâmetros de inicialização do contexto do servlet" na página 267.

     É possível modificar e usar o arquivo `splicer.properties` de amostra que é fornecido com a instalação do eXtreme Scale. Também é possível usar o

script addObjectGridServlet, que insere o gerenciador de sessão ao estender cada servlet. Entretanto, o script recomendados é o script addObjectGridFilter.

- **Script de construção Ant**

  O WebSphere eXtreme Scale é fornecido com um arquivo `build.xml` que pode ser utilizado pelo Apache Ant, que está incluído na pasta `wasRoot/bin` de uma instalação do WebSphere Application Server. O `build.xml` pode ser modificado para alterar as propriedades de configuração do gerenciador de sessões. As propriedades de configuração são idênticas aos nomes de propriedades no arquivo `splicer.properties`. Depois que o arquivo `build.xml` tiver sido modificado, chame o processo Ant executando `ant.sh`, `ws_ant.sh` (UNIX) ou `ant.bat`, `ws_ant.bat` (Windows).

- **Atualize o descritor da Web manualmente**

  Edite o arquivo `web.xml` que é empacotado com o aplicativo da Web para incorporar a declaração do filtro, seu mapeamento de servlet e os parâmetros de inicialização de contexto do servlet. Não use este método porque ele é propenso a erros.

Para obter uma lista dos parâmetros que podem ser utilizados, consulte "Parâmetros de inicialização do contexto do servlet".

2. Incorpore os arquivos JAR do gerenciador de replicação de sessão do WebSphere eXtreme Scale ao seu aplicativo. É possível integrar os arquivos ao diretório `WEB-INF/lib` do módulo aplicativo ou ao caminho de classe do servidor de aplicativos. Os arquivos JAR necessários variam dependendo do tipo de contêineres que você está usando:
   - Contêineres do eXtreme Scale remotos: `ogclient.jar` e `sessionobjectgrid.jar`
   - Contêineres do eXtreme Scale integrados: `objectgrid.jar` e `sessionobjectgrid.jar`
3. Opcional: Se você usar contêineres do eXtreme Scale remotos, inicie os contêineres. Veja detalhes em "Iniciando Processos do Contêiner" na página 331.
4. Implemente o aplicativo. Implemente o aplicativo com seu conjunto normal de etapas para um servidor ou cluster. Após implementar o aplicativo, é possível iniciar o aplicativo.
5. Acesse o aplicativo. É possível acessar o aplicativo, que interage com o gerenciador de sessões e o WebSphere eXtreme Scale.

O que Fazer Depois

É possível alterar a maioria dos atributos de conexão para o gerenciador de sessões quando você instrumenta seu aplicativo para utilizar o gerenciador de sessões. Esses atributos incluem variações para o tipo de replicação (síncrona ou assíncrona), para o tamanho da tabela de sessão na memória e assim por diante. Não considerando os atributos que podem ser alterados no momento da instrumentação do aplicativo, os únicos outros atributos de configuração que podem ser alterados após a implementação do aplicativo são os atributos que estão relacionados à topologia em cluster do servidor WebSphere eXtreme Scale e a maneira pela qual seus clientes (gerenciadores de sessões) se conectam a eles.

Parâmetros de inicialização do contexto do servlet

A lista de parâmetros de inicialização de contexto de servlet a seguir pode ser especificada no arquivo splicer properties conforme requerido no método de conexão escolhido.

Parâmetros

objectGridType

Um valor de cadeia de "REMOTE" ou "EMBEDDED". O padrão é REMOTE.

Se forem configurados para "REMOTE", os dados da sessão serão armazenados fora do servidor no qual o aplicativo da Web está sendo executado.

Se for configurado para "EMBEDDED", um contêiner integrado do eXtreme Scale será iniciado no processo do servidor de aplicativos no qual o aplicativo da Web está sendo executado.

objectGridName

Um valor de cadeia que define o nome da instância de ObjectGrid usada para um aplicativo da Web particular. O nome padrão é session.

Esta propriedade deve refletir o objectGridName nos arquivos XML da grade de objeto e de implementação usados para iniciar os contêineres do eXtreme Scale.

catalogHostPort

O servidor de catálogos pode ser contatado para obter uma instância de ObjectGrid do lado do cliente. O valor deve ter o formato `host:port<,host:port>`, no qual host é o host do listener no qual o servidor de catálogos está em execução e port é a porta listener para esse processo do servidor de catálogos. Esta lista pode ser arbitrariamente longa e é usada apenas para autoinicialização. O primeiro endereço viável é usado. Será opcional dentro do WebSphere Application Server se a propriedade catalog.services.cluster for configurada.

replicationInterval

Um valor de número inteiro (em segundos) que define o tempo entre a gravação de sessões atualizadas para ObjectGrid. O padrão é 2. Os valores possíveis vão de 0 a 60. 0 significa que as sessões atualizadas são gravadas no ObjectGrid no final da chamada de método de serviço de servlet para cada pedido. Um valor replicationInterval mais alto melhora o desempenho porque menos atualizações são gravadas na grade. No entanto, um valor mais alto torna a configuração menos tolerante a falhas.

Esta configuração aplica-se apenas quando objectGridType é configurado para "REMOTE".

sessionTableSize

Um valor de número inteiro que define o número de referências de sessão mantidas na memória. O padrão é 2000.

Esta configuração pertence apenas a uma topologia REMOTE porque a topologia EMBEDDED já tem os dados da sessão na mesma camada que o contêiner de Web.

As sessões são despejadas da tabela de memória com base na lógica Usada Menos Recentemente. Quando uma sessão é despejada da tabela de memória, ela é invalidada do contêiner de Web, mas os dados não são removidos da grade, portanto, os pedidos subsequentes para essa sessão ainda podem recuperar os dados.

fragmentedSession

Um valor de cadeia de "true" ou "false." O valor padrão é "true." Use esta configuração para controlar se o produto armazena dados da sessão como uma entrada inteira ou armazena cada atributo separadamente.

Configure fragmentedSession para "true" se a sessão de aplicativo da Web tiver muitos atributos ou atributos com tamanhos grandes. Configure fragmentedSession para "false" apenas se uma sessão tiver poucos atributos, pois todos os atributos são armazenados na mesma chave na grade.

Na implementação anterior, baseada em filtro, esta propriedade foi mencionada como persistenceMechanism, com os valores possíveis de ObjectGridStore (fragmentado) e ObjectGridAtomicSessionStore (não fragmentado).

securityEnabled

Um valor de cadeia de "true" ou "false." O valor padrão é "false." Esta configuração ativa a segurança do cliente do eXtreme Scale. Ela deve corresponder à configuração securityEnabled no arquivo de propriedades de servidor eXtreme Scale. Se as configurações não corresponderem, ocorrerá uma exceção.

credentialGeneratorClass

O nome da classe que implementa a interface com.ibm.websphere.objectgrid.security.plugins.CredentialGenerator. Essa classe é usada para obter credenciais para os clientes.

credentialGeneratorProps

As propriedades para a classe de implementação CredentialGenerator. As propriedades são configuradas para o objeto com o método **setProperties(String)**. O valor credentialGeneratorProps é usado apenas se o valor da propriedade **credentialGeneratorClass** não for nulo.

objectGridXML

O local de arquivo do arquivo `objectgrid.xml`. O arquivo XML integrado compactado na biblioteca do eXtreme Scale será carregado automaticamente se objectGridType=EMBEDDED e a propriedade **objectGridXML** não forem especificados.

objectGridDeploymentXML

O local do arquivo XML da política de implementação de objectGrid. O arquivo XML integrado compactado na biblioteca do eXtreme Scale será carregado automaticamente se objectGridType=EMBEDDED e a propriedade **objectGridDeploymentXML** não forem especificados.

traceSpec

A especificação de rastreio do IBM WebSphere, como um valor de cadeia. Use esta configuração para servidores de aplicativos diferentes do WebSphere Application Server.

traceFile

O local do arquivo de rastreio, como um valor de cadeia. Use esta configuração para servidores de aplicativos diferentes do WebSphere Application Server.

Configurando o Provedor de Cache Dinâmico para o WebSphere eXtreme Scale

A instalação e configuração do provedor de cache dinâmico para o eXtreme Scale depende de quais são seus requisitos e do ambiente configurado.

Antes de Iniciar

Instale o provedor de cache dinâmico.

Para usar o provedor de cache dinâmico, o WebSphere eXtreme Scale deve ser instalado além das implementações de nó do WebSphere Application Server, incluindo o nó do gerenciador de implementação. O provedor de cache dinâmico do eXtreme Scale é suportado nas seguintes versões do WebSphere Application Server.

- WebSphere Application Server 6.1.0.25 + PK85622 e superior
- WebSphere Application Server 7.0.0.3 + PK85622 e superior

Para obter instruções de instalação, consulte Instalando para 6.1 ou Instalando para 7.0.

Para obter informações sobre como usar o provedor de cache dinâmico do eXtreme Scale com o IBM WebSphere Commerce, consulte os tópicos a seguir na documentação do IBM WebSphere Commerce:

- Ativando o Serviço de Cache Dinâmico e o Armazenamento em Cache do Servlet
- Ativando o Cache de Dados do WebSphere Commerce

Sobre Esta Tarefa

Siga essas etapas para configurar o provedor de cache dinâmico do eXtreme Scale:

1. Ative o provedor de cache dinâmico do eXtreme Scale.

   No WebSphere Application Server 7.0, é possível configurar o serviço de cache dinâmico para usar o provedor de cache dinâmico do eXtreme Scale com o console administrativo ou com uma propriedade customizada.

   Depois de instalar o eXtreme Scale, o provedor de cache dinâmico do eXtreme Scale estará imediatamente disponível como uma opção "Provedor de Cache" no console administrativo. Para obter mais informações, consulte Selecionando um Provedor de Servido de Cache.

   Também é possível configurar o provedor de cache dinâmico do eXtreme Scale para uma instância de cache ao configurar os seguinte pares propriedade customizada e valor na instância. Essas propriedades customizadas são a única maneira de ativar o provedor eXtreme Scale no WebSphere Application Server 6.1 da seguinte forma.

   ```
   com.ibm.ws.cache.CacheConfig.cacheProviderName =
      "com.ibm.ws.objectgrid.dynacache.CacheProviderImpl"
   ```

   Se você não estiver direcionando especificamente seu armazenamento em cache para uma instância Cache do Objeto ou Cache do Servlet definida, é provável que as chamadas da API do Cache Dinâmico estejam sendo atendidas por baseCache. baseCache é a instância de cache padrão criada como parte do Serviço de Cache Dinâmico. As mesmas propriedades de configuração são usadas para configurar a instância baseCache para usar o eXtreme Scale como seu provedor de cache. No entanto, essas propriedades de configuração devem ser configuradas como Propriedades customizadas JVM para que elas afetem baseCache.

   Configurar variáveis de configuração como uma propriedade customizada JVM pode resultar em que outros caches as selecionem também. As instâncias Cache do Objeto e Cache do Servlet devem preferir as propriedades customizadas configuradas na instância de cache, mas se você descobrir que uma instância de

cache específica está usando o provedor do eXtreme Scale quando deveria estar usando o provedor padrão, a situação poderá ser corrigida usando-se as propriedades customizadas. Para fazer com que uma instância de cache use o provedor de cache dinâmico padrão, configure a propriedade do provedor de cache da seguinte maneira:

```
com.ibm.ws.cache.CacheConfig.cacheProviderName = "default"
```

Como as propriedades de configuração do provedor de cache dinâmico do eXtreme Scale configuradas para baseCache podem ser selecionadas por outras instâncias de cache, tenha cuidado em confiar nos valores padrão para as propriedades de configuração do cache.

2. Defina a configuração de replicação para o cache.

   Com o provedor de cache dinâmico do eXtreme Scale, é possível ter instâncias de cache locais e instâncias de cache replicadas. No caso de instâncias de cache locais, nenhuma outra configuração é necessária e o restante desta seção poderá ser ignorado.

   Os caches replicados podem ser criados de duas formas. A primeira forma é ativar a replicação de cache por meio do console de administração. Essa ativação pode ser feita a qualquer momento no WebSphere Application Server Versão 7.0, mas será necessário criar um domínio de replicação DRS no WebSphere Application Server Versão 6.1.

   A segunda maneira é usar o seguinte par propriedade customizada e valor para forçar o cache a informar que isso é um cache replicado, mesmo se o domínio de replicação DRS não tiver sido designado para ele.

   ```
   com.ibm.ws.cache.CacheConfig.enableCacheReplication = "true"
   ```

3. Configure a topologia para o serviço de cache dinâmico.

   O único parâmetro de configuração necessário para o provedor de cache dinâmico do eXtreme Scale é a topologia do cache. A seguinte variável deve ser configurada como uma Propriedade Customizada no serviço de cache dinâmico.

   ```
   com.ibm.websphere.xs.dynacache.topology
   ```

   A seguir há os três valores possíveis para essa propriedade.

   - integrado
   - integrado_particionado
   - remoto

   É necessário usar um dos valores permitidos.

4. Configure o número de contêineres iniciais para o serviço de cache dinâmico.

   É possível aumentar o desempenho dos caches que estão usado a topologia particionada integrada ao configurar o número de contêineres iniciais. A variável a seguir deve ser configurada como uma propriedade de sistema na WebSphere Application Server Java Virtual Machine.

   ```
   com.ibm.websphere.xs.dynacache.num_initial_containers
   ```

   O valor recomendado dessa propriedade de configuração é um número inteiro que seja igual ou pouco menor que o número total de instâncias do WebSphere Application Server que acessam essa instância de cache distribuída. Por exemplo, se um serviço de cache dinâmico for compartilhado entre os membros da grade, a variável deverá ser configurada para o número de membros da grade.

Para topologias integradas ou embedded_partitioned, você deve estar usando a Versão 7.0 do WebSphere Application Server. Configure a seguinte propriedade customizada no processo JVM para garantir que os contêineres iniciais estejam disponíveis imediatamente.

```
com.ibm.ws.cache.CacheConfig.createCacheAtServerStartup=true
```

5. Configurar a grade de serviço de catálogo do eXtreme Scale.

   Quando estiver usando o eXtreme Scale como o provedor de cache dinâmico para uma instância de cache distribuído, você deverá configurar um domínio do serviço de catálogo do eXtreme Scale.

   Um único domínio do serviço de catálogo pode atender vários provedores do serviço de cache dinâmico retornados pelo eXtreme Scale.

   7.1+ Um serviço de catálogo pode ser executado dentro ou fora de processos WebSphere Application Server. Iniciando com o eXtreme Scale Versão 7.1, quando você usar o console administrativo para configurar mecanismos de domínio para servidores de catálogos, o cache dinâmico usará estas configurações. Não é necessário executar etapas adicionais para configurar um serviço de catálogo. Para obter informações adicionais, consulte "Criando Domínios do Serviço de Catálogo no WebSphere Application Server" na página 344. Quando estiver executando um domínio do serviço de catálogo, você deverá configurar a propriedade customizada **catalog.services.cluster** para os terminais do serviço de catálogo.

6. Configurar os objetos de chave customizados.

   Quando estiver usando os objetos customizados como chaves, os objetos deverão implementar a interface Serializável ou Externalizável. Quando estiver usando as topologias integradas ou particionadas integradas, é necessário colocar os objetos no caminho de biblioteca compartilhado do WebSphere, como se eles estivessem sendo usados com o provedor de cache dinâmico padrão. Consulte Usando as interfaces DistributedMap e DistributedObjectCache para o cache dinâmico no centro de informações do WebSphere Application Server Network Deployment para obter mais detalhes.

   Se você estiver usando a topologia remota, você deverá colocar os objetos de chaves customizados no CLASSPATH para os contêineres do eXtreme Scale independentes.

7. Configure os servidores de contêiner do eXtreme Scale.

   Os dados em cache são armazenados nos contêineres do WebSphere eXtreme Scale. Os contêineres podem ser executados dentro ou fora dos processos do WebSphere Application Server. O provedor do eXtreme Scale cria automaticamente contêineres dentro do processo do WebSphere quando estiver usando topologias integradas ou particionadas integradas para uma instância de cache. Nenhuma configuração adicional é necessária para essas topologias.

   Quando estiver usando uma topologia remota, é necessário iniciar os contêineres eXtreme Scale independente antes que as instâncias do WebSphere Application Server que acessam a instância do cache seja inicializadas. Certifique-se de que todos os contêineres para um serviço de cache dinâmico específico apontem para os mesmos terminais de serviço de catálogo.

   Os arquivos de configuração XML para os contêineres independentes do provedor eXtreme Scale Dynamic Cache estão localizados no diretório `<install_root>/customLibraries/ObjectGrid/dynacache/etc` para instalações além do WebSphere Application Server ou no diretório `<install_root>/ObjectGrid/dynacache/etc` para instalações independentes. Os arquivos são chamados de `dynacache-remote-objectgrid.xml` e `dynacache-remote-definition.xml`. Faça cópias desses arquivos para editar e usar quando ativar

contêineres independentes para o provedor de cache dinâmico do eXtreme Scale. O parâmetro **numIntitialContainers** no arquivo **dynacache-remote-deployment.xml** deve ser configurado de acordo com o número de processos de contêineres que estão sendo executados. Observe que o atributo **numberOfPartitions** no arquivo `dynacache-remote-objectgrid.xml` tem um valor padrão de 47.

Nota: O conjunto de processos de contêineres precisa ter memória livre suficiente para atender a todas as instâncias de cache dinâmico configuradas para usar a topologia remota. Qualquer processo do WebSphere Application Server que compartilhar valores iguais ou equivalentes para **catalog.services.cluster** deve usar o mesmo conjunto de contêineres independentes. O número de contêineres e o número de máquinas nas quais eles residem devem ser redimensionados adequadamente. Consulte "Planejamento de Capacidade e Alta Disponibilidade (Armazenamento em Cache Dinâmico)" na página 13 para obter detalhes adicionais.

Um exemplo de entrada da linha de comandos UNIX que ativa um contêiner independente para o provedor de cache dinâmico do eXtreme Scale é mostrado no seguinte código:

```
startOgServer.sh container1 -objectGridFile ../dynacache/etc/dynacache-remote-objectgrid.xml -deploymentPolicyFile ../dynacache/etc/dynacache-remote-deployment.xml -catalogServiceEndpoints MyServer1.company.com:2809
```

8. Para topologias distribuídas ou integradas, ative o agente de dimensionamento para melhorar as estimativas de consumo de memória.

   O agente de dimensionamento faz a estimativa do consumo de memória (estatística usedBytes). O agente requer uma JVM Java 5 ou superior.

   Carregue o agente incluindo o seguinte argumento na linha de comandos da JVM:

   ```
   -javaagent:WXS lib directory/wxssizeagent.jar
   ```

   Para uma topologia integrada, inclua o argumento na linha de comandos do processo do WebSphere Application Server.

   Para uma topologia distribuída, inclua o argumento na linha de comandos dos processos (contêineres) do eXtreme Scale e do processo do WebSphere Application Server.

Configurando Integração de Spring

Arquivo XML descritor do Spring

Use um arquivo XML descritor do Spring para configurar e integrar o eXtreme Scale com o Spring.

Nas seguintes seções, cada elemento e atributo do arquivo `objectgrid.xsd` Spring é definido. O arquivo `objectgrid.xsd` Spring está no arquivo `ogspring.jar` e no espaço de nomes objectgrid `com/ibm/ws/objectgrid/spring/namespace`. Consulte o "Arquivo `objectgrid.xsd` Spring" na página 280 para obter um exemplo do esquema do XML descritor.

Elemento register

Utilize o elemento register para registrar o bean factory padrão para o ObjectGrid.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

id Especifica o nome do diretório bean padrão para um determinado ObjectGrid.

gridname
Especifica o nome da instância do ObjectGrid. O valor designado para esse atributo deve corresponder a um ObjectGrid válido configurado no arquivo descritor ObjectGrid.

```
<register
(1) id="register id"
(2) gridname="ObjectGrid name"
/>
```

Elemento server

Use o elemento server para definir um servidor eXtreme Scale, que pode hospedar um contêiner, um serviço de catálogo, ou ambos.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

id Especifica o nome do servidor eXtreme Scale.

tracespec
Indica o tipo de rastreio e permite o rastreio e a especificação de rastreio para o servidor.

tracefile
Fornece o caminho e o nome do traceFile a ser criado e usado.

statspec
Indica a especificação de estatísticas para o servidor.

jmxport
Designa o número de porta não-utilizada por meio da qual você deseja ativar as conexões JMX/RMI. O JMX ativa o monitoramento e o gerenciamento para os sistemas remotos.

isCatalog
Especifica se o servidor particular hospeda um serviço de catálogo. O valor padrão é `false`.

name
Especifica o nome do servidor.

haManagerPort
Configura o número da porta para o Gerenciador de Alta Disponibilidade (HA Manager).

listenerHost
Configura o nome do host ao qual o ORB deve se ligar.

listenerPort
Configura a porta à qual o ORB deve se ligar.

maximumThreadPoolSize
Configura o número máximo de encadeamentos no conjunto.

memoryThresholdPercentage
Configura o limite de memória (porcentagem de heap máx.) para o despejo baseado na memória.

minimumThreadPoolSize
Configura o número mínimo de encadeamentos no conjunto.

workingDirectory
A propriedade que define qual diretório o servidor ObjectGrid deve usar para todas as configurações padrão.

zoneName
Define a zona à qual este servidor pertence.

enableChannelFramework
Configura o TransportMode nas propriedades do ORB da IBM para ChannelFramework,

enableSystemStreamToFile
Define se SystemOut e SystemErr devem ser enviados para um arquivo.

enableMBeans
Determina se o ObjectGrid registrará ou não os MBeans neste processo.

serverPropertyFile
Carrega as propriedades de servidor de um arquivo.

catalogServerProperties
Especifica o servidor de catálogos que hospeda o servidor.

```
<server
(1) id="server id"
(2) tracespec="the server trace specification"
(3) tracefile="the server trace file"
(4) statspec="the server statistic specification"
(5) jmxport="JMX port number"
(6) isCatalog="true"|"false"
(7) name="the server name"
(8) haManagerPort="the haManager port"
(9) listenerHost="the orb binding host name"
(10) listenerPort="the orb binding listener port"
(11) maximumThreadPoolSize="the number of maximum threads"
(12) memoryThresholdPercentage="the memory threshold (percentage of max heap)"
(13) minimumThreadPoolSize="the number of minimum threads"
(14) workingDirectory="location for the working directory"
(15) zoneName="the zone name"
(16) enableChannelFramework="true"|"false"
(17) enableSystemStreamToFile="true"|"false"
(18) enableMBeans="true"|"false"
(19 serverPropertyFile="location of the server properties file."
(20) catalogServerProperties="the catalog server properties reference"
/>
```

Elemento catalog

Use o elemento catalog para rotear os servidores de contêineres na grade de dados.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

host
Especifica o nome do host da estação de trabalho onde o serviço de catálogo está em execução.

port
Especifica o número da porta unido ao nome do host para determinar a porta do serviço de catálogo para a qual o cliente pode se conectar.

```
<catalog
(1) host="catalog service host name"
(2) port="catalog service port number"
/>
```

Elemento catalog server

Use o elemento de propriedades catalog Server para definir um serviço do servidor de catálogos.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

catalogServerEndPoint
Especifica as propriedades de conexão para o servidor de catálogos.

enableQuorum
Determina se o quorum deve ser ativado.

heartBeatFrequencyLevel
Configura o nível de frequência de pulsação.

domainName
Define o nome de domínio usado para identificar exclusivamente esta grade do serviço de catálogo para clientes ao rotear para vários domínios.

foreignDomains
Uma conexão bidirecional será estabelecida para cada um dos domínios externos com o objetivo de trocar dados em um cenário multi-primário.

clusterSecurityURL
Configura o local do arquivo de segurança específico para o serviço de catálogo.

```
<catalog server
(1) catalogServerEndPoint="a catalog server endpoint reference "
(2) enableQuorum="true"|"false"
(3) heartBeatFrequencyLevel="
  HEARTBEAT_FREQUENCY_LEVEL_TYPICAL|
  HEARTBEAT_FREQUENCY_LEVEL_RELAXED|
  HEARTBEAT_FREQUENCY_LEVEL_AGGRESSIVE"
(4) domainName="the domain name used to uniquely identify this catalog service grid"
(5) domainEndpoints="a reference to the domain name endpoints"
(6) foreignDomains="the name of the foreign domain"
(7) clusterSecurityURL="the The cluster security file location."/>
```

Elemento Catalog Server Endpoint

Use o elemento Catalog Server EndPoint para criar um terminal do servidor de catálogos para ser usado por um elemento do servidor de catálogos.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

serverName
Especifica o nome que identifica o processo que você está ativando.

hostName
Especifica o nome do host para a máquina em que o servidor é ativado.

clientPort
Especifica a porta usada para comunicação do cluster do catálogo do peer.

peerPort
Especifica a porta usada para comunicação do cluster do catálogo do peer.

```
<catalogServerEndPoint
(1) name="catalog server endpoint name"
(2) host=""
(3) clientPort=""
(4) peerPort""
/>
```

Elemento container

Utilize o elemento container para armazenar os próprios dados.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

objectgridxml
Especifica o caminho e o nome do arquivo do XML descritor a ser usado para especificar as características do ObjectGrid, incluindo mapas, estratégia de bloqueio e plug-ins.

deploymentxml
Especifica o caminho e o nome do arquivo XML que é usado com o descritor XML para determinar o particionamento, a replicação, o número de contêineres iniciais e outras configurações.

server
Especifica o servidor no qual o contêiner está hospedado.

```
<server
(1) objectgridxml="the objectgrid descriptor XML file"
(2) deploymentxml ="the objectgrid deployment descriptor XML file "
(3) server="the server reference "
/>
```

Elemento JPALoader

Use o elemento JPALoader para sincronizar o cache ObjectGrid com um armazenamento de dados backend existente ao usar a API ObjectMap.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

entityClassName
Ativa o uso de JPAs, como EntityManager.persist e EntityManager.find. O atributo **entityClassName** é necessário para o JPALoader.

preloadPartition
Especifica a partição na qual o pré-carregamento do mapa será iniciado. Se o valor for menor que 0 ou maior que (totalNumberOfPartition – 1), o pré-carregamento do mapa não será iniciado.

```
<JPALoader
(1) entityClassName="the entity class name"
(2) preloadPartition ="int"
/>
```

Elemento JPATxCallback

Use o elemento JPATxCallback para coordenar as transações JPA e ObjectGrid.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

persistenceUnitName
Cria um JPA EntityManagerFactory e localiza os metadados da entidade JPA no arquivo persistence.xml. O atributo **persistenceUnitName** é necessário.

jpaPropertyFactory
Especifica o factory para criar um mapa de propriedade de persistência para substituir as propriedades de persistência padrão. Este atributo deve referenciar um bean.

exceptionMapper
Especifica o plug-in ExceptionMapper que pode ser usado para funções de mapeamento de exceção específicos do JPA ou específicos do banco de dados. Este atributo deve referenciar um bean.

```
<JPATxCallback
(1) persistenceUnitName="the JPA persistence unit name"
(2) jpaPropertyFactory ="JPAPropertyFactory bean reference"
(3) exceptionMapper="ExceptionMapper bean reference"
/>
```

Elemento JPAEntityLoader

Use o elemento JPAEntityLoader para sincronizar o cache ObjectGrid com um armazenamento de dados backend existente ao usar a API EntityManager.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

entityClassName
Ativa o uso de JPAs, como EntityManager.persist e EntityManager.find. O atributo **entityClassName** é opcional para o elemento JPAEntityLoader. Se o elemento não estiver configurado, a classe de entidade configurada no mapa de entidade ObjectGrid será usada. A mesma classe deve ser usada para o ObjectGrid EntityManager e para o provedor JPA.

preloadPartition
Especifica a partição na qual o pré-carregamento do mapa será iniciado. Se o valor for menor que 0 ou maior que (totalNumberOfPartition – 1), o pré-carregamento do mapa não será iniciado.

```
<JPAEntityLoader
(1) entityClassName="the entity class name"
(2) preloadPartition ="int"
/>
```

Elemento LRUEvictor

Use o elemento LRUEvictor para decidir quais entradas devem ser descartadas quando o mapa exceder o número máximo de entradas.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

maxSize
Especifica o total de entradas em uma fila até o evictor tiver que intervir.

sleepTime
Configura o tempo em segundos entre um tempo de acesso do evictor sobre as filas de mapa para determinar as ações necessárias no mapa.

numberOfLRUQueues
Permite configurar quantas filas devem ser varridas pelo evictor para evitar que uma única fila tenha o tamanho do mapa inteiro.

```
<LRUEvictor
(1) maxSize="int"
(2) sleepTime ="seconds"
(3) numberOfLRUQueues ="int"
/>
```

Elemento LFUEvictor

Use o elemento LFUEvictor para determinar quais entradas devem se descartadas quando o mapa exceder o número máximo de entradas.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

maxSize
Especifica o total de entradas permitidas em cada heap até o evictor tiver que atuar.

sleepTime
Configura o tempo em segundos entre o tempo de acesso do evictor sobre os heaps de mapa para determinar as ações necessárias no mapa.

numberOfHeaps
Permite configurar quantos heaps devem ser varridos pelo evictor para evitar que um único heap tenha o tamanho do mapa inteiro.

```
<LFUEvictor
(1) maxSize="int"
(2) sleepTime ="seconds"
(3) numberOfHeaps ="int"
```

Elemento HashIndex

Use o elemento HashIndex com a reflexão Java para examinar dinamicamente os objetos armazenados em um mapa quando eles forem atualizados.

- Número de ocorrências: Zero para muitas
- Elemento-filho: Nenhum

Atributos

name
Especifica o nome do índice, que deve ser exclusivo para cada mapa.

attributeName
Especifica o nome do atributo a ser indexado. Para índices acessados por campo, o nomes do atributo é equivalente ao nome de campo. Para índices acessados por propriedade, o nomes do atributo é o nome da propriedade compatível com JavaBean.

rangeIndex
Indica se a indexação do intervalo está ativada. O valor padrão é `false`.

fieldAccessAttribute
Usado para mapas sem entidade. O método getter é usado para acessar os dados. O valor padrão é `false`. Se você especificar o valor `true`, o objeto será acessado usando diretamente os campos.

POJOKeyIndex
Usado para mapas sem entidade. O valor padrão é `false`. Se você especificar o

valor como `true`, o índice examinará o objeto na parte chave do mapa, que é útil quando a chave for uma chave composta e o valor não tiver a chave integrada com ele. Se você não especificar o valor ou especificá-lo como `false`, então o índice examinará o objeto na parte do valor do mapa.

```
<HashIndex
(1) name="index name"
(2) attributeName="attribute name"
(3) rangeIndex ="true"|"false"
(4) fieldAccessAttribute ="true"|"false"

(5) POJOKeyIndex ="true"|"false"
/>
```

Arquivo `objectgrid.xsd` Spring

Use o arquivo `objectgrid.xsd` Spring para integrar o eXtreme Scale ao Spring para gerenciar as transações do eXtreme Scale e configurar os clientes e servidores.

Consulte o "Arquivo XML descritor do Spring" na página 273 para obter descrições dos elementos e atributos definidos no arquivo `objectgrid.xsd` Spring.

Arquivo `objectgrid.xsd` Spring

```
<xsd:schema xmlns="http://www.ibm.com/schema/objectgrid"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:beans="http://www.springframework.org/schema/beans"
    targetNamespace="http://www.ibm.com/schema/objectgrid"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

   <xsd:import namespace="http://www.springframework.org/schema/beans" />

 <xsd:element name="transactionManager">
      <xsd:complexType>
         <xsd:attribute name="id" type="xsd:ID" />
      </xsd:complexType>
   </xsd:element>

 <xsd:element name="register">
      <xsd:complexType>
         <xsd:attribute name="id" type="xsd:ID" />
         <xsd:attribute name="gridname" type="xsd:string" />
      </xsd:complexType>
   </xsd:element>

 <xsd:element name="server">
      <xsd:complexType>
  <xsd:choice minOccurs="0" maxOccurs="unbounded">
           <xsd:element ref="catalog" />
         </xsd:choice>
         <xsd:attribute name="id" type="xsd:ID" />
         <xsd:attribute name="tracespec" type="xsd:string" />
         <xsd:attribute name="tracefile" type="xsd:string" />
         <xsd:attribute name="statspec" type="xsd:string" />
         <xsd:attribute name="jmxport" type="xsd:integer" />
         <xsd:attribute name="isCatalog" type="xsd:boolean" />
         <xsd:attribute name="name" type="xsd:string" />
         <xsd:attribute name="haManagerPort" type="xsd:integer"/>
         <xsd:attribute name="listenerHost" type="xsd:string"/>
         <xsd:attribute name="listenerPort" type="xsd:integer"/>
         <xsd:attribute name="maximumThreadPoolSize" type="xsd:integer"/>
         <xsd:attribute name="memoryThresholdPercentage" type="xsd:integer"/>
         <xsd:attribute name="minimumThreadPoolSize" type="xsd:integer"/>
         <xsd:attribute name="workingDirectory" type="xsd:string"/>
         <xsd:attribute name="zoneName" type="xsd:string"/>
         <xsd:attribute name="enableChannelFramework" type="xsd:boolean"/>
         <xsd:attribute name="enableSystemStreamToFile" type="xsd:boolean"/>
         <xsd:attribute name="enableMBeans" type="xsd:boolean"/>
         <xsd:attribute name="serverPropertyFile" type="xsd:string"/>
         <xsd:attribute name="catalogServerProperties" type="xsd:string"/>
      </xsd:complexType>
 </xsd:element>

 <xsd:element name="catalog">
   <xsd:complexType>
     <xsd:attribute name="host" type="xsd:string" />
     <xsd:attribute name="port" type="xsd:integer" />
   </xsd:complexType>
 </xsd:element>
```

```
<xsd:element name="catalogServerProperties">
  <xsd:complexType>
 <xsd:choice minOccurs="0" maxOccurs="unbounded">
           <xsd:element ref="catalogServerEndPoint"/>
        </xsd:choice>
       <xsd:attribute name="id" type="xsd:ID" />
    <xsd:attribute name="enableQuorum" type="xsd:boolean"/>
    <xsd:attribute name="heartBeatFrequencyLevel" type="xsd:integer"/>
    <xsd:attribute name="domainName" type="xsd:string"/>
    <xsd:attribute name="domainEndpoints" type="xsd:string"/>
    <xsd:attribute name="foreignDomains" type="xsd:string"/>
    <xsd:attribute name="clusterSecurityURL" type="xsd:anyURI"/>
  </xsd:complexType>
</xsd:element>

<xsd:element name="catalogServerEndPoint">
  <xsd:complexType>
     <xsd:attribute name="name" type="xsd:string" />
    <xsd:attribute name="host" type="xsd:string" />
    <xsd:attribute name="clientPort" type="xsd:integer"/>
    <xsd:attribute name="peerPort" type="xsd:integer"/>
  </xsd:complexType>
</xsd:element>

<xsd:element name="container">
     <xsd:complexType>
        <xsd:attribute name="id" type="xsd:ID" />
        <xsd:attribute name="objectgridxml" type="xsd:string" />
        <xsd:attribute name="deploymentxml" type="xsd:string" />
        <xsd:attribute name="server" type="xsd:string" />
     </xsd:complexType>
  </xsd:element>

   <xsd:element name="JPALoader">
     <xsd:complexType>
        <xsd:attribute name="id" type="xsd:ID" />
        <xsd:attribute name="entityClassName" type="xsd:string" />
        <xsd:attribute name="preloadPartition" type="xsd:integer" />
     </xsd:complexType>
  </xsd:element>

  <xsd:element name="JPATxCallback">
     <xsd:complexType>
        <xsd:attribute name="id" type="xsd:ID" />
        <xsd:attribute name="persistenceUnitName" type="xsd:string" />
  <xsd:attribute name="jpaPropertyFactory" type="xsd:string" />
  <xsd:attribute name="exceptionMapper" type="xsd:string" />
  </xsd:complexType>
  </xsd:element>

  <xsd:element name="JPAEntityLoader">
     <xsd:complexType>
        <xsd:attribute name="id" type="xsd:ID" />
        <xsd:attribute name="entityClassName" type="xsd:string" />
        <xsd:attribute name="preloadPartition" type="xsd:integer" />
     </xsd:complexType>
  </xsd:element>

  <xsd:element name="LRUEvictor">
     <xsd:complexType>
        <xsd:attribute name="id" type="xsd:ID" />
        <xsd:attribute name="maxSize" type="xsd:integer" />
        <xsd:attribute name="sleepTime" type="xsd:integer" />
        <xsd:attribute name="numberOfLRUQueues" type="xsd:integer" />
        <xsd:attribute name="useMemoryUsageThresholdEviction" type="xsd:boolean" />
     </xsd:complexType>
  </xsd:element>

  <xsd:element name="LFUEvictor">
     <xsd:complexType>
        <xsd:attribute name="id" type="xsd:ID" />
        <xsd:attribute name="maxSize" type="xsd:integer" />
        <xsd:attribute name="sleepTime" type="xsd:integer" />
        <xsd:attribute name="numberOfHeaps" type="xsd:integer" />
        <xsd:attribute name="useMemoryUsageThresholdEviction" type="xsd:boolean" />
     </xsd:complexType>
  </xsd:element>

  <xsd:element name="HashIndex">
     <xsd:complexType>
        <xsd:attribute name="id" type="xsd:ID" />
        <xsd:attribute name="name" type="xsd:string" />
        <xsd:attribute name="attributeName" type="xsd:string" />
        <xsd:attribute name="rangeIndex" type="xsd:boolean" />
        <xsd:attribute name="fieldAccessAttribute" type="xsd:boolean" />
        <xsd:attribute name="POJOKeyIndex" type="xsd:boolean" />
     </xsd:complexType>
  </xsd:element>

</xsd:schema>
```

Beans de Extensão Spring e Suporte a Espaço de Nomes

O WebSphere eXtreme Scale fornece um recurso para declarar Plain Old Java Objects (POJOs) para uso como pontos de extensão no arquivo `objectgrid.xml`, além de uma maneira de nomear os beans e especificar o nome da classe. Normalmente, as instâncias da classe especificada são criadas, e tais objetos são usados como plug-ins. Agora, o eXtreme Scale pode delegar que Spring obtenha instâncias destes objetos plug-in. Se um aplicativo utiliza o Spring, então, normalmente, tais POJOs possuem um requisito de serem conectados ao resto do aplicativo.

Em alguns casos, você deve usar o Spring para configurar determinados objetos do plug-in. Use a configuração a seguir como exemplo:

```
<objectGrid name="Grid">
    <bean id="TransactionCallback" className="com.ibm.websphere.objectgrid.jpa.JPATxCallback">
        <property name="persistenceUnitName" type="java.lang.String"  value="employeePU" />
    </bean>
    ...
</objectGrid>
```

A implementação TransactionCallback integrada, classe com.ibm.websphere.objectgrid.jpa.JPATxCallback, é configurada como a classe TransactionCallback. Esta classe é configurada com a propriedade persistenceUnitName conforme mostrado no exemplo anterior. A classe JPATxCallback também tem o atributo JPAPropertyFactory, que é do tipo java.lang.Object. A configuração XML do ObjectGrid não pode suportar este tipo de configuração.

A integração Spring do eXtreme Scale soluciona este problema delegando a criação do bean à estrutura do Spring. A configuração revisada é a seguinte:

```
<objectGrid name="Grid">
    <bean id="TransactionCallback" className="{spring}jpaTxCallback"/>
    ...
</objectGrid>
```

O arquivo Spring para o objeto "Grid" contém as seguintes informações:

```
<bean id="jpaTxCallback" class="com.ibm.websphere.objectgrid.jpa.JPATxCallback" scope="shard">
    <property name="persistenceUnitName" value="employeeEMPU"/>
    <property name="JPAPropertyFactory" ref ="jpaPropFactory"/>
</bean>

<bean id="jpaPropFactory" class="com.ibm.ws.objectgrid.jpa.plugins.
JPAPropFactoryImpl" scope="shard">
</bean>
```

Aqui, o TransactionCallback está especificado como `{spring}jpaTxCallback`, e os beans jpaTxCallback e jpaPropFactory estão configurados no arquivo Spring como mostrado no exemplo anterior. A configuração Spring torna possível a configuração de um bena JPAPropertyFactory como um parâmetro do objeto JPATxCallback.

Bean factory Spring padrão

Quando o eXtreme Scale encontra um plug-in ou um bean de extensão (como um ObjectTransformer, utilitário de carga, TransactionCallback e assim por diante) com um valor className que inicia com o prefixo `{spring}`, o eXtreme Scale usará o restante do nome como um nome Spring Bean e obterá a instância do bean usando o Spring Bean Factory.

Pelo padrão, se nenhum bean factory tiver sido registrado para um determinado ObjectGrid, então ele tenta localizar um arquivo `ObjectGridName_spring.xml`. Por exemplo, se sua grade for chamada como "Grid", então o arquivo XML será chamado como `/Grid_spring.xml`. Este arquivo deve estar no caminho da classe ou

em um diretório `META-INF` que está no caminho da classe. Se este arquivo for encontrado, então o eXtreme Scale construi um ApplicationContext usando tal arquivo e construi beans a partir desse bean factory.

Bean factory Spring customizado

O WebSphere eXtreme Scale também fornece uma API ObjectGridSpringFactory para registrar uma instância do Spring Bean Factory para usar para um ObjectGrid específico nomeado. Esta API registra uma instância de BeanFactory com eXtreme Scale ussando o método estático a seguir:

```
void registerSpringBeanFactoryAdapter(String objectGridName, Object
springBeanFactory)
```

Suporte a Espaço de Nomes

Desde a versão 2.0, o Spring possui um mecanismo para extensão baseado em esquema para o formato XML do Spring básico para definição e configuração de beans. O ObjectGrid usa este novo recurso para definir e configurar beans ObjectGrid. Com a extensão de esquema XML do Spring, algumas das implementações integradas dos plug-ins do eXtreme Scale e alguns beans do ObjectGrid são predefinidos no espaço de nomes "objectgrid". Ao escrever os arquivos de configuração Spring, não é necessário especificar o nome completo de classe das implementações integradas. Em vez disso, é possível referenciar os beans predefinidos.

Além disso, com os atributos dos beans definidos no esquema XML, é menos provável que você forneça um nome de atributo errado. A validação XML baseada no esquema XML pode capturar estes tipos de erros anteriormente no ciclo de desenvolvimento.

Estes beans definidos nas extensões de esquema XML são:

- transactionManager
- register
- server
- catálogo
- catalogServerProperties
- contêiner
- JPALoader
- JPATxCallback
- JPAEntityLoader
- LRUEvictor
- LFUEvictor
- HashIndex

Estes beans são definidos no esquema XML objectgrid.xsd. Este arquivo XSD é enviado como arquivo `com/ibm/ws/objectgrid/spring/namespace/objectgrid.xsd` no arquivo `ogspring.jar`. Para obter descrições detalhadas do arquivo XSD e dos beans definidos no arquivo XSD, consulte as informações sobre o arquivo descritor Spring no *Guia de Administração*.

Continue usando o exemplo JPATxCallback da seção anterior. Na seção anterior, o bean JPATxCallback é configurado como o seguinte:

```
<bean id="jpaTxCallback" class="com.ibm.websphere.objectgrid.jpa.JPATxCallback" scope="shard">
    <property name="persistenceUnitName" value="employeeEMPU"/>
    <property name="JPAPropertyFactory" ref ="jpaPropFactory"/>
</bean>

<bean id="jpaPropFactory" class="com.ibm.ws.objectgrid.jpa.plugins.JPAPropFactoryImpl" scope="shard">
</bean>
```

Usando este recurso de espaço de nomes, a configuração XML do Spring pode ser escrita da seguinte forma:

```
<objectgrid:JPATxCallback id="jpaTxCallback" persistenceUnitName="employeeEMPU"
 jpaPropertyFactory="jpaPropFactory" />

<bean id="jpaPropFactory" class="com.ibm.ws.objectgrid.jpa.plugins.JPAPropFactoryImpl"
 scope="shard">
</bean>
```

Observe aque que em vez de especificar a classe "com.ibm.websphere.objectgrid.jpa.JPATxCallback" como no exemplo anterior, foi usado diretamente o bean "objectgrid:JPATxCallback" predefinido. Como pode ser visto, esta configuração é menos detalhada e mais amigável para verificação de erro.

Início com Servidor de Contêineres com Spring Extension Beans

Neste exemplo, será mostrado como iniciar um servidor ObjectGrid usando beans de extensão gerenciados Spring do ObjectGrid e suporte a espaço de nomes.

Arquivo XML do ObjectGrid

Primeiramente, defina um arquivo XML do ObjectGrid muito simples que contenha um "Grid" do ObjectGrid e uma mapa "Test". O ObjectGrid possui um plug-in ObjectGridEventListener chamado "partitionListener", e o mapa "Test" possui um Evictor conectado chamado "testLRUEvictor". Observe que ambos os plug-ins ObjectGridEventListener e Evictor são configurados usando Spring pois seus nomes contêm "{spring}".

```
<?xml version="1.0" encoding="UTF-8"?>
<objectGridConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/config ../objectGrid.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config">
    <objectGrids>
        <objectGrid name="Grid">
     <bean id="ObjectGridEventListener" className="{spring}partitionListener" />
            <backingMap name="Test" pluginCollectionRef="test" />
        </objectGrid>
    </objectGrids>

    <backingMapPluginCollections>
        <backingMapPluginCollection id="test">
            <bean id="Evictor" className="{spring}testLRUEvictor"/>
        </backingMapPluginCollection>
    </backingMapPluginCollections>
</objectGridConfig>
```

Arquivo XML de implementação do ObjectGrid

Agora, crie um arquivo XML de implementação simples do ObjectGrid da forma a seguir. Ele particiona o ObjectGrid em 5 partições, e nenhuma réplica é necessária.

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy ../deploymentPolicy.xsd"
 xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">
    <objectgridDeployment objectgridName="Grid">
        <mapSet name="mapSet" numInitialContainers="1" numberOfPartitions="5" minSyncReplicas="0"
            maxSyncReplicas="1" maxAsyncReplicas="0">
```

```
            <map ref="Test"/>
        </mapSet>
    </objectgridDeployment>
</deploymentPolicy>
```

Arquivo XML Spring do ObjectGrid

Agora serão usados tanto beans de extensão gerenciado Spring do ObjectGrid e recursos de suporte a espaço de nomes para configurar os beans ObjectGrid. O arquivo XML do Spring é nomeado como "Grid_spring.xml". Observe que estão incluídos dois esquemas no arquivo XML: spring-beans-2.0.xsd é para uso com benas gerenciados do Spring, e objectgrid.xsd é para uso com beans predefinidos no espaço de nomes objectgrid.

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:aop="http://www.springframework.org/schema/aop"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xmlns:objectgrid="http://www.ibm.com/schema/objectgrid"
       xsi:schemaLocation="
       http://www.ibm.com/schema/objectgrid
   http://www.ibm.com/schema/objectgrid/objectgrid.xsd
       http://www.springframework.org/schema/beans
   http://www.springframework.org/schema/beans/spring-beans-2.0.xsd">

    <objectgrid:register id="ogregister" gridname="Grid"/>

    <objectgrid:server id="server" isCatalog="true" name="server">
        <objectgrid:catalog host="localhost" port="2809"/>
    </objectgrid:server>

    <objectgrid:container id="container"
   objectgridxml="com/ibm/ws/objectgrid/test/springshard/objectgrid.xml"
       deploymentxml="com/ibm/ws/objectgrid/test/springshard/deployment.xml"
   server="server"/>

    <objectgrid:LRUEvictor id="testLRUEvictor" numberOfLRUQueues="31"/>

    <bean id="partitionListener"
   class="com.ibm.websphere.objectgrid.springshard.ShardListener" scope="shard"/>
</beans>
```

Há 6 beans definidos neste arquivo XML do Spring:

1. *objectgrid:register*: Isto registra o bean factory padrão para o "Grid" do ObjectGrid.
2. *objectgrid:server*: Isto define um servidor do ObjectGrid com o nome "server". Este servidor também fornece o serviço de catálogo desde que ele possua um bean objectgrid:catalog aninhado nele.
3. *objectgrid:catalog*: Isto define um terminal de serviço de catálogo ObjectGrid, que está configurado como "localhost:2809".
4. *objectgrid:container*: Isto define um contêiner ObjectGrid com o arquivo XML objectgrid especificado e o arquivo XML de implementação como discutido anteriormente. A propriedade de servidor especifica em qual servidor este contêiner está hospedado.
5. *objectgrid:LRUEvictor*: Isto define um LRUEvictor com a quantidade de filas LRU para usar configurada como 31.
6. *bean partitionListener*: Isto define um plug-in ShardListener. É necessário fornecer uma implementação para este plug-in, caso contrário, ele não poderá usar os beans predefinidos. Além disso, esse escopo do bean é configurado como "shard", o que significa que existe apenas uma instância desse ShardListener por shard ObjectGrid.

Início do servidor

O fragmento a seguir inicia o servidor ObjectGrid, que hospeda tanto o serviço de contêiner e o serviço de catálogo. Como podemos ver, o único método que precisamos chamar para iniciar o servidor é para obter um "contêiner" de bean do bean factory. Isto simplifica a complexidade de programação pela movimentação da maioria da lógica na configuração do Spring.

```
public class ShardServer extends TestCase
{
    Container container;
    org.springframework.beans.factory.BeanFactory bf;

    public void startServer(String cep)
    {
        try
        {
            bf = new org.springframework.context.support.ClassPathXmlApplicationContext(
                "/com/ibm/ws/objectgrid/test/springshard/Grid_spring.xml", ShardServer.class);
            container = (Container)bf.getBean("container");
        }
        catch(Exception e)
        {
            throw new ObjectGridRuntimeException("Cannot start OG container", e);
        }
    }

    public void stopServer()
    {
        if(container != null)
            container.teardown();
    }
}
```

Configurando o Serviço de Dados REST

Use os links a seguir para localizar informações sobre como administrar o serviço de dados REST. Consulte também as informações da interface de programação de aplicativos sobre o Mbean RestService.

Arquivo de Propriedades do Serviço de Dados REST

Para configurar o serviço de dados REST, edite o arquivo de propriedades para REST e defina o esquema de entidade necessário para uma grade do WebSphere eXtreme Scale.

O arquivo de propriedades do serviço de dados REST é o arquivo de configuração principal para o serviço de dados REST do eXtreme Scale. Este arquivo é um arquivo de propriedades Java típico com pares de valor da chave. Por padrão, o tempo de execução do serviço de dados REST procura um arquivo `wxsRestService.properties` nomeado no caminho de classe. O arquivo também pode ser definido explicitamente por meio do uso da propriedade de sistema: `wxs.restservice.props`.

```
-Dwxs.restservice.props=/usr/configs/dataservice.properties
```

Quando o serviço de dados REST é carregado, o arquivo de propriedades utilizado é exibido nos arquivos de log:

```
CWOBJ4004I: Os arquivos de propriedades do serviço de dados REST do eXtreme
Scale foram carregados: [/usr/configs/RestService.properties]
```

O arquivo de propriedades do serviço de dados REST suporta as seguintes propriedades:

Tabela 11. Propriedades para o Serviço de Dados REST

Propriedade	Descrição

catalogServiceEndPoints	A lista requerida delimitada por vírgula de hosts e portas de um domínio do serviço de catálogo no formato: <host:port>. Isso é opcional se estiver usando o WebSphere Application Server integrado ao eXtreme Scale para hospedar o serviço de dados REST. Consulte a documentação do produto WebSphere eXtreme Scale para obter detalhes sobre como configurar e iniciar um serviço de catálogo.
	
	`catalogServiceEndPoints=`
	
	`server1:2809,server2:2809`

objectGridNames	Os nomes necessários de ObjectGrids para expor ao serviço REST. Pelo menos um nome de ObjectGrid é necessário. Separe nomes de ObjectGrid múltiplos utilizando uma vírgula: `ECommerceGrid,InventoryGrid`
objectGridClientXML	O nome opcional do arquivo XML de substituição de cliente do ObjectGrid. O nome especificado aqui será carregado do caminho de classe. O padrão é: /META-INF/objectGridClient.xml. Consulte a documentação do produto WebSphere eXtreme Scale para obter detalhes sobre como configurar um cliente do eXtreme Scale.
objectGridNames	Os nomes necessários de ObjectGrids para expor ao serviço REST. Pelo menos um nome de ObjectGrid é necessário. Separe nomes de ObjectGrid múltiplos utilizando uma vírgula: `ECommerceGrid,InventoryGrid`
objectGridClientXML	O nome opcional do arquivo XML de substituição de cliente do ObjectGrid. O nome especificado aqui será carregado do caminho de classe. O padrão é: /META-INF/objectGridClient.xml. Consulte a documentação do produto WebSphere eXtreme Scale para obter detalhes sobre como configurar um cliente do eXtreme Scale.

Tabela 11. Propriedades para o Serviço de Dados REST (continuação)

Propriedade	Descrição
ogClientPropertyFile	O nome opcional do arquivo de propriedades do cliente do ObjectGrid. Este arquivo contém propriedades de segurança que são necessárias para ativar a segurança do cliente do ObjectGrid. Se o atributo "securityEnabled" for configurado no arquivo de propriedades, a segurança será ativada no cliente do ObjectGrid usado pelo serviço REST. O "credentialGeneratorProps" também deve ser configurado no arquivo de propriedades para um valor no formato "user:pass" ou um valor de {xor_encoded user:pass}
loginType	O tipo de autenticação usado pelo Serviço REST quando a segurança do cliente do ObjectGrid for ativada. Se a segurança do cliente do ObjectGrid não for ativada, esta propriedade será ignorada. Se a segurança do cliente do ObjectGrid for ativada e loginType for configurado para 'basic', o serviço REST irá: • Usar as credenciais especificadas na propriedade 'credentialGeneratorProps' do arquivo de propriedades do cliente do ObjectGrid para operações do ObjectGrid na inicialização do serviço. • Usar a autenticação HTTP BASIC para operações de sessões do ObjectGrid por pedido Se a segurança do cliente do ObjectGrid for ativada e loginType for configurado para 'none', o serviço REST irá: • Usar as credenciais especificadas na propriedade 'credentialGeneratorProps' do arquivo de propriedades do cliente do ObjectGrid para operações do ObjectGrid na inicialização do serviço. • Usar as credenciais especificadas na propriedade 'credentialGeneratorProps' do arquivo de propriedades do cliente do ObjectGrid para operações de sessões do ObjectGrid por pedido.
traceFile	O nome opcional do arquivo para o qual redirecionar a saída de rastreio. O padrão é logs/trace.log.
traceSpec	A especificação de rastreio opcional que o servidor de runtime do eXtreme Scale deve usar inicialmente. O padrão é *=all=disabled. Para rastrear o serviço de dados REST inteiro, utilize: ObjectGridRest*=all=enabled

Tabela 11. Propriedades para o Serviço de Dados REST (continuação)

Propriedade	Descrição
verboseOutput	Se configurado como true, os clientes do serviço de dados REST receberão informações de diagnóstico adicionais quando ocorrer uma falha. O padrão é false. Esse valor opcional deve ser configurado como false para ambientes de produção conforme informações sensíveis forem reveladas.
maxResultsPerCollection	O número máximo opcional de resultados que será retornado em uma consulta. O valor padrão é ilimitado e um valor válido é um número inteiro positivo.
wxsRestAccessRightsFile	O nome opcional do arquivo de propriedades de direitos de acesso do serviço REST do eXtreme Scale que especifica os direitos de acesso para as operações de serviço e as entidades do ObjectGrid. Se esta propriedade for especificada, o serviço REST tentará carregar o arquivo a partir do caminho especificado e tentará também carregar o arquivo a partir de seu caminho de classe.

Configuração do WebSphere eXtreme Scale

O serviço de dados REST do eXtreme Scale interage com oeXtreme Scale usando a API EntityManager. Um esquema de entidade é definido para uma grade do eXtreme Scale e os metadados para as entidades são automaticamente consumidos pelo serviço de dados REST. Consulte Definindo um esquema de entidade para obter detalhes sobre como configurar um esquema de entidade.

Por exemplo, é possível definir uma entidade representando uma Pessoa em uma grade do eXtreme Scale como a seguir:

```
@Entity
public class Person {
  @Id String taxId;
  String firstName;
  String lastName;
}
```

Dica: As anotações utilizadas aqui estão no pacote `com.ibm.websphere.projector.annotations`.

O serviço REST automaticamente cria um Modelo de Dados de Entidade ADO.NET para o documento Serviços de Dados (EDMX), que está disponível usando a URI $metadata:

```
http://localhost:8080/wxsrestservice/restservice/NorthwindGrid/$metadata
```

Depois que a grade do eXtreme Scale for configurada e estiver em execução, um cliente do eXtreme Scale deverá ser configurado e compactado. Para obter detalhes sobre como configurar o pacote do cliente do serviço de dados REST do eXtreme Scale, consulte as informações de compactação e implementação no “Instalando o Serviço de Dados REST” na página 300.

Modelo de Entidade

As entidades do WebSphereeXtreme Scale são modeladas usando as anotações da entidade ou um arquivo descritor de metadados da entidade. Para obter detalhes sobre como configurar um esquema de entidade do eXtreme Scale, consulte as informações sobre como definir um esquema de entidade no *Guia de Programação*. O serviço REST do eXtreme Scale usa os metadados de entidade para criar automaticamente um modelo EDMX para o serviço de dados.

Esta versão do serviço de dados REST do WebSphere eXtreme Scale tem as seguintes restrições de esquema:

- Ao definir entidades em uma grade particionada, todas as entidades devem ter uma associação com valor único direta ou indireta com a entidade raiz (uma associação chave). O tempo de execução do cliente WCF Data Service deve ter capacidade de acessar cada entidade diretamente por meio de seu endereço canônico. Portanto, a chave da entidade raiz que é utilizada para o roteamento de partição (a raiz do esquema) deve fazer parte da chave na entidade-filha.

  Por exemplo:

  ```
  @Entity(schemaRoot=true)
  public class Person {
      @Id String taxId;
      String firstName;
      String lastName;
      @OneToMany(mappedBy="person")
      List<Address> addresses;

  }

  @Entity
  public class Address {
      @Id int addrId;
      @Id @ManyToOne Person person;
      String street;
  }
  ```

- Associações bidirecionais e unidirecionais são suportadas. Entretanto, as associações unidirecionais podem nem sempre funcionar a partir de um cliente Microsoft® WCF Data Services, já que elas só podem ser navegadas em uma direção e a especificação da Microsoft requer que todas as associações sejam bidirecionais.
- Restrições de referência não são suportadas. O tempo de execução do eXtreme Scale não valida chaves entre entidades. As associações entre as entidades devem ser gerenciadas pelo cliente.
- Tipos complexos não são suportados. A API EntityManager do eXtreme Scale não suporta atributos integráveis. Todos os atributos devem ser do tipo simples (consulte os tipos de atributos simples listados abaixo). Os atributos de tipo não simples são tratados como um objeto binário a partir da perspectiva do cliente.
- A herança de entidade não é suportada. A API EntityManager do eXtreme Scale não suporta herança.
- Recursos de Mídia e Links de Mídia não são suportados. O atributo HasStream de EntityType no Conceptual Schema Definition Language Document for Data Services nunca é utilizado.

Mapeamento entre Tipos de Dados EDM e Tipos de Dados Java

O protocolo OData define a seguinte lista de tipos Entity Data Model (EDM) em seu sistema de tipo abstrato. Os tópicos a seguir descrevem como o adaptador

REST do eXtreme Scale escolhe o tipo EDM com base no tipo básico definido na entidade. Para obter detalhes sobre os tipos EDM, consulte: Biblioteca MSDN: Sistema de Tipo Abstrato.

Os tipos EDM a seguir estão disponíveis no WCF Data Services:

- Edm.Binary
- Edm.Boolean
- Edm.Byte
- Edm.DateTime
- Edm.Time
- Edm.Decimal
- Edm.Double
- Edm.Single
- Edm.Float
- Edm.Guid *
- Edm.Int16
- Edm.Int32
- Edm.Int64
- Edm.SByte
- Edm.String

O tipo EDM: Edm.Guid não é suportado pelo serviço de dados REST do eXtreme Scale.

Mapeando Tipos Java para Tipos EDM

O serviço de dados REST do eXtreme Scale converterá automaticamente os tipos de entidade básicos para tipos EDM. O mapeamento de tipo pode ser visto por meio da exibição do documento de metadados do Entity Data Model Extensions (EDMX) utilizando o URI $metadata. O tipo EDM é aquele que é utilizado pelos clientes para ler e gravar dados no serviço de dados REST.

Tabela 12. Tipos Java Mapeados para Tipos EDM. A tabela mostra o mapeamento do tipo Java definido para uma entidade para o tipo de dado EDM. Com a recuperação de dados utilizando uma consulta, os dados serão representados com estes tipos:

Tipo Java	**Tipo EDM**
boolean java.lang.Boolean	Edm.Boolean
byte java.lang.Byte	Edm.SByte
short java.lang.Short	Edm.Int16
int java.lang.Integer	Edm.Int32
long java.lang.Long	Edm.Int64
float java.lang.Float	Edm.Single
double java.lang.Double	Edm.Double
java.math.BigDecimal	Edm.Decimal
java.math.BigInteger	java.math.BigInteger
java.lang.String	Edm.String
char	char
java.lang.Character	java.lang.Character

Tabela 12. Tipos Java Mapeados para Tipos EDM (continuação). A tabela mostra o mapeamento do tipo Java definido para uma entidade para o tipo de dado EDM. Com a recuperação de dados utilizando uma consulta, os dados serão representados com estes tipos:

Tipo Java	**Tipo EDM**
Char[]	Char[]
java.lang.Character[]	java.lang.Character[]
java.util.Calendar	Edm.DateTime
java.util.Date	java.util.Date
java.sql.Date	java.sql.Date
java.sql.Timestamp	java.sql.Timestamp
java.sql.Time	java.sql.Time
Outros tipos	Edm.Binary

Mapeando de Tipos EDM para Tipos Java

Para pedidos Update e Insert, a carga útil especifica os dados a serem atualizados ou inseridos no serviço de dados REST do eXtreme Scale. O serviço pode converter automaticamente tipos de dado compatíveis em tipos de dados definidos no documento EDMX. O serviço de dados REST converte as representações de cadeia codificadas em XML do valor no tipo correto utilizando o seguinte processo de duas etapas:

1. Uma verificação de tipo é feita para garantir que o tipo EDM seja compatível com o tipo Java. Um tipo EDM é compatível com um tipo Java se os dados suportados pelo tipo EDM forem um subconjunto dos dados suportados pelo tipo Java. Por exemplo, o tipo Edm.int32 é compatível com um tipo Java long, mas o tipo Edm.int32 não é compatível com um tipo Java short.
2. Será criado um objeto de tipo Java de destino que representa o valor de cadeia na carga útil.

Tabela 13. Tipo EDM compatível com o tipo Java

Tipo EDM	**Tipo Java**
Edm.Boolean	boolean java.lang.Boolean

Tabela 13. Tipo EDM compatível com o tipo Java (continuação)

Tipo EDM	Tipo Java
Edm.SByte	byte java.lang.Byte short java.lang.Short int java.lang.Integer long java.lang.Long float java.lang.Float double java.lang.Double java.math.BigDecimal java.math.BigInteger char java.lang.Character
Edm.Byte, Edm.Int16	short java.lang.Short int java.lang.Integer long java.lang.Long float java.lang.Float double java.lang.Double java.math.BigDecimal java.math.BigInteger char java.lang.Character

Tabela 13. Tipo EDM compatível com o tipo Java (continuação)

Tipo EDM	Tipo Java
Edm.Int32	int java.lang.Integer long java.lang.Long float java.lang.Float double java.lang.Double java.math.BigDecimal java.math.BigInteger
Edm.Int64	long java.lang.Long double java.lang.Double java.math.BigDecimal java.math.BigInteger
Edm.Double	double java.lang.Double java.math.BigDecimal
Edm.Decimal	double java.lang.Double java.math.BigDecimal java.math.BigInteger
Edm.Single	float java.lang.Float double java.lang.Double java.math.BigDecimal

Tabela 13. Tipo EDM compatível com o tipo Java (continuação)

Tipo EDM	Tipo Java
Edm.String	java.lang.String char java.lang.Character Char[] java.lang.Character[] java.math.BigDecimal java.math.BigInteger
Edm.DateTime	java.util.Calendar java.util.Date java.sql.Date java.sql.Time java.sql.Timestamp
Edm.Time	java.sql.Time java.sql.Timestamp

Tipos Temporais de Mapeamento

O Java inclui cinco tipos temporais para armazenamento de data, hora ou ambas: java.util.Date, java.sql.Date, java.sql.Time, java.sql.Timestamp e java.util.Calendar. Todos esses tipos são expressos no modelo de dados da entidade como Edm.DateTime. O serviço REST do eXtreme Scale converte automaticamente e normaliza os dados dependendo do tipo Java. Este tópico descreve vários problemas dos quais os desenvolvedores devem estar cientes ao utilizarem qualquer tipo temporal.

Diferenças de fuso horário

No WCF Data Services, as descrições de valores de hora no tipo Edm.DateTime são sempre expressas utilizando o padrão Coordinated Universal Time (UTC), que é o nome internacionalmente reconhecido de Horário de Greenwich (GMT). O Coordinated Universal Time é a hora conforme medição a zero graus de longitude, o ponto de origem do UTC. O horário de verão não se aplica ao UTC.

Convertendo entre entidade e tipos EDM

Quando um cliente envia um pedido ao serviço de dados REST, a data e a hora são representadas como um horário do fuso horário GMT, como no exemplo a seguir:

```
"2000-02-29T21:30:30.654123456"
```

O serviço de dados REST constrói a instância de tipo temporal Java apropriada e a insere na entidade na grade.

Quando um cliente solicita uma propriedade que é um tipo temporal Java do serviço de dados REST do eXtreme Scale, o valor é sempre normalizado como uma valor do fuso horário GMT. Por exemplo, se uma entidade java.util.Date for construída da seguinte forma:

```
Calendar c = Calendar.getInstance();
c.clear();
c.set(2000, 1, 29, 21, 30, 30);
Date d = c.getTime();
```

A data e a hora são representadas com o uso do fuso horário padrão do processo Java, pois Calendar.getInstance() criará um objeto Calendar com o fuso horário local. Se o fuso horário local for CST, a data, quando recuperada do serviço de dados REST, será a representação GMT da hora: `"2000-03-01T03:30:30"`

Normalização de java.sql.Date

Uma entidade do eXtreme Scale pode definir um atributo com tipo Java java.sql.Date. Esse tipo de dado não inclui a hora e é normalizado pelo serviço de dados REST. Isso significa que o tempo de execução do eXtreme Scale não armazena nenhuma informação sobre horas, minutos, segundos ou milissegundos no atributo java.sql.Date. Independentemente do deslocamento de fuso horário, a data é sempre representada como uma data local.

Por exemplo, se o cliente atualizar uma propriedade java.sql.Date com o valor “2009-01-01T03:00:00”, o serviço de dados REST, que está no fuso horário CST (-06:00), criará simplesmente uma instância java.sql.Date da qual a hora é configurada como “2009-01-01T00:00:00” do horário CST local. Não é feita nenhuma conversão de fuso horário para se criar o valor java.sql.Date. Quando o cliente do serviço REST recuperar o valor desse atributo, ele será exibido como “2009-01-01T00:00:00Z”. Se uma conversão de fuso horário fosse feita, o valor seria exibido como tendo a data de “2008-12-31”, que seria incorreta.

Normalização de java.sql.Time

Semelhantes ao java.sql.Date, os valores de java.sql.Time são normalizados e não incluem informações de data. Isso significa que o tempo de execução do eXtreme Scale não armazena o ano, mês ou dia. A hora é armazenada com o uso do horário GMT da época de 1 de janeiro de 1970, que está consistente com a implementação java.sql.Time.

Por exemplo, se o cliente atualizar uma propriedade java.sql.Time com o valor "2009-01-01T03:00:00", o serviço de dados REST criará uma instância java.sql.Time com os milissegundos configurados como 3*60*60*1000, o que é igual a 3 horas. Quando o serviço REST recuperar o valor, ele será exibido como "1970-01-01:03:00:00Z".

Associações

Associações definem o relacionamento entre duas entidades peer. O serviço REST do eXtreme Scale reflete as associações modeladas com entidades definidas com entidades anotadas do eXtreme Scale ou entidades definidas usando um arquivo XML descritor de entidade.

Manutenção de associação

O serviço de dados REST do eXtreme Scale não suporta restrições de integridade referencial. O cliente deve garantir que as referências sejam atualizadas quando entidades forem removidas ou incluídas. Se uma entidade de destino de uma associação for removida da grade, mas o link entre a entidade de origem e de destino não for removido, o link será quebrado. A API do serviço de dados REST e EntityManager do eXtreme Scale tolera links quebrados e os registrará como avisos CWPRJ1022W. As associações quebradas serão simplesmente removidas da carga útil do pedido.

Utilize um pedido em lote para agrupar atualizações de associação em uma única transação para evitar links quebrados. Consulte a seção para obter detalhes sobre pedidos em lote.

O elemento ReferentialConstraint do Modelo de Dados de Entidade ADO.NET não é usado pelo serviço de dados REST do eXtreme Scale.

Multiplicidade de associação

Entidades podem ter associações com diversos valores ou associações com valor único. As associações com diversos valores, ou coletas, são associações um-para-muitos ou muitos-para-muitos. As associações com valor único são associações um-para-um ou muitos-para-um.

Em uma grade particionada, todas as entidades devem ter um caminho de associação chave com valor único para uma entidade raiz. Outra seção deste tópico mostra como definir uma associação chave. Como a entidade raiz é utilizada para particionar a entidade, as associações muitos-para-muitos não são permitidas para grades particionadas. Para obter um exemplo de como modelar um esquema de entidade relacional para uma grade particionada, consulte Modelo de dados escalável no eXtreme Scale.

O exemplo a seguir descreve como os tipos de associação da API EntityManager, modelados com o uso de classes Java anotadas, são mapeados para o ADO.NET Entity Data Model:

```
@Entity
public class Customer {
    @Id String customerId;
    @OneToOne TaxInfo taxInfo;
    @ManyToOne Address homeAddress;
    @OneToMany Collection<Order> orders;
    @ManyToMany Collection<SalesPerson> salespersons;
}

<Association Name="Customer_TaxInfo">
    <End Type="Model1.Customer" Role="Customer" Multiplicity="1"
/>
    <End Type="Model1.TaxInfo " Role="TaxInfo" Multiplicity="1"
/>
</Association>
<Association Name="Customer_Address">
    <End Type="Model1.Customer" Role="Customer" Multiplicity="1"
/>
    <End Type="Model1.Address" Role="TaxInfo" Multiplicity="*"
/>
</Association>
<Association Name="Customer_Order">
    <End Type="Model1.Customer" Role="Customer" Multiplicity="*" />
    <End Type="Model1.Order" Role="TaxInfo" Multiplicity="1" />
</Association>
```

```
<Association Name="Customer_SalesPerson">
    <End Type="Model1.Customer" Role="Customer" Multiplicity="*" />
    <End Type="Model1.SalesPerson" Role="TaxInfo" Multiplicity="*" />
</Association>
```

Associações bidirecionais e unidirecionais

As associações de entidades podem ser unidirecionais ou bidirecionais. Ao especificar o atributo "mappedBy" na anotação @OneToOne, @OneToMany ou @ManyToMany ou o atributo "mapped-by" na tag do atributo XML um-para-um, um-para-muitos ou muitos-para-muitos, a entidade se torna bidirecional. O protocolo OData requer atualmente que todas as entidades sejam bidirecionais, permitindo que clientes gerem caminhos de navegação em ambas as direções. A API EntityManager do eXtreme Scale permite modelar associações unidirecionais que podem salvar memória e simplificar a manutenção das associações. Se uma associação unidirecional for utilizada, o cliente de serviço de dados REST só deverá navegar por meio da associação utilizando a associação definida.

Por exemplo: Se uma associação muitos-para-um unidirecional for definida entre Address e Country, o seguinte URI não será permitido:

```
/restservice/CustomerGrid/Country('USA')/addresses
```

Associações chave

Associações com valor único (um-para-um e muitos-para-um) também podem ser incluídas como um todo ou como parte da chave de entidades. Isso é conhecido como associação chave.

As associações chave são necessárias quando se usa uma grade particionada. A associação chave deve ser definida para todas as entidades-filhas em um esquema de entidade particionada. O protocolo OData requer que todas as entidades sejam diretamente endereçáveis. Isso significa que a chave na entidade-filha deve incluir a chave utilizada para o particionamento.

No exemplo a seguir, Customer tem uma associação um-para-muitos com Order. A entidade Customer é a entidade raiz e o atributo customerId é utilizado para particionar a entidade. Order incluiu Customer como parte de sua identidade:

```
@Entity(schemaRoot="true")
public class Customer {
    @Id String customerId;
    @OneToMany(mappedBy="customer") Order orders
}

@Entity
public class Order {
    @Id int orderId;
    @Id @ManyToOne Customer customer;
    java.util.Date orderDate;
}
```

Quando o serviço de dados REST gera o documento EDMX para esse modelo, os campos-chave Customer são automaticamente incluídos como parte da entidade Order:

```
<EntityType Name="Order">
<Key>
    <PropertyRef Name="orderId"/>
    <PropertyRef Name="customer_customerId"/>
</Key>
```

```
<Property Name="orderId" Type="Edm.Int64" Nullable="false"/>
<Property Name="customer_customerId" Type="Edm.String"
    Nullable="false"/>
<Property Name="orderDate" Type="Edm.DateTime" Nullable="true"/>
<NavigationProperty Name="customer"
    Relationship="NorthwindGridModel.Customer_orders"
    FromRole="Order" ToRole="Customer"/>

<NavigationProperty Name="orderDetails"
    Relationship="NorthwindGridModel.Order_orderDetails"
    FromRole="Order" ToRole="OrderDetail"/>
</EntityType>
```

Quando uma entidade for criada, a chave nunca deverá mudar. Isso significa que se a associação chave entre uma entidade-filha e seu pai tiver que mudar, a entidade-filha deverá ser removida e recriada com um pai diferente. Em uma grade particionada, isso vai exigir dois conjuntos de mudanças em lote diferentes, já que a mudança provavelmente envolverá mais de uma partição.

Operações em cascata

A API EntityManager permite uma política de cascata flexível. Associações podem ser marcadas para formar uma cascata das operações persist, remove, invalidate ou merge. Tais operações de cascata podem acontecer em um ou ambos os lados de uma associação bidirecional.

O protocolo OData só permite operações delete em cascata no lado um da associação. A anotação CascadeType.REMOVE ou o atributo XML cascade-remove não podem ser definidos em ambos os lados de uma associação bidirecional um-para-um ou no lado muitos de uma associação um-para-muitos. O exemplo a seguir ilustra uma associação bidirecional Cascade.REMOVE válida:

```
@Entity(schemaRoot="true")
public class Customer {
    @Id String customerId;
    @OneToMany(mappedBy="customer", cascade=CascadeType.REMOVE)
    Order orders
}

@Entity
public class Order {
    @Id int orderId;
    @Id @ManyToOne Customer customer;
    java.util.Date orderDate;
}
```

A associação EDMX resultante é semelhante à seguinte:

```
<Association Name="Customer_orders">
    <End Type="NorthwindGridModel.Customer" Role="Customer"
        Multiplicity="1">
        <OnDelete Action="Cascade"/>
    </End>
    <End Type="NorthwindGridModel.Order" Role="Order"
        Multiplicity="*"/>
</Association>
```

Administrando o Serviço de Dados REST

Sobre Esta Tarefa

Use os links a seguir para localizar informações sobre como administrar o serviço de dados REST. Consulte também as informações de Mbean RestService.

Instalando o Serviço de Dados REST

Este tópico descreve como instalar o serviço de dados REST do WebSphere eXtreme Scale em um servidor da Web.

Antes de Iniciar

Requisitos de Software

O serviço de dados REST do eXtreme Scale é um aplicativo da Web Java que pode ser implementado para qualquer servidor de aplicativos que suporte a especificação de servlet Java, Versão 2.3 e um ambiente de tempo de execução Java, Versão 5 ou posterior.

O software a seguir é necessário:

- Java Standard Edition 5 ou posterior

  Restrição: Embora o eXtreme Scale suporte o Java Standard Edition 1.4 ou posterior, o serviço de dados REST requer o Java Standard Edition 5 ou posterior.
- Contêiner do servlet da Web, Versão 2.3 ou posterior, que inclui uma das opções a seguir:
  - WebSphere Application Server Versão 6.1.0.25 ou posterior
  - WebSphere Application Server Versão 7.0.0.5 ou posterior
  - WebSphere Community Edition Versão 2.1.1.3 ou posterior
  - Apache Tomcat Versão 5.5 ou posterior
- eXtreme Scale, Versão 7.1 ou posterior (incluindo o teste)

Sobre Esta Tarefa

O serviço de dados REST do eXtreme Scale inclui um único arquivo WAR `wxsrestservice.war`. O arquivo `wxsrestservice.war` inclui um único servlet que age como um gateway entre os aplicativos cliente WCF Data Services ou qualquer outro cliente HTTP REST e uma grade do eXtreme Scale.

O serviço de dados REST inclui uma amostra que permite que você crie rapidamente uma grade do eXtreme Scale e interaja com ela usando um cliente do eXtreme Scale ou o serviço de dados REST. Consulte Amostra e tutorial dos serviços de dados REST para obter detalhes sobre como usar a amostra.

Quando o eXtreme Scale 7.1 é instalado ou o teste do eXtreme Scale Versão 7.1 é extraído, os seguintes diretórios e arquivos são incluídos:

- `restservice_home/lib`

  O diretório lib contém esses arquivos:
  - `wxsrestservice.ear` – O archive do aplicativo corporativo do serviço de dados REST para uso com WebSphere Application Server e WebSphere Application Server CE.

– `wxsrestservice.war` – O módulo da Web de serviço de dados REST para uso com Apache Tomcat.

O arquivo `wxsrestservice.ear` inclui o arquivo `wxsrestservice.war` e os dois são estreitamente acoplados ao tempo de execução do WebSphere eXtreme Scale. Se for feito o upgrade do eXtreme Scale para uma nova versão um se um fix pack for aplicado, o arquivo `wxsrestservice.war` ou o arquivo `wxsrestservice.ear` precisarão ter seu upgrade feito manualmente para a versão instalada neste diretório.

- `restservice_home/gettingstarted`

  O diretório `gettingstarted` contém uma amostra simples que demonstra como usar o serviço de dados REST do eXtreme Scale com uma grade do eXtreme Scale.

Procedimento

Empacote e implemente o serviço de dados REST.
O serviço de dados REST é projetado como um módulo WAR autocontido. Para configurar o serviço de dados REST, você deve primeiro empacotar a configuração do serviço de dados REST e os arquivos de configuração opcionais do eXtreme Scale em um arquivo ou diretório JAR. Essa compactação de aplicativo é referida pelo tempo de execução do servidor de contêiner de Web. O diagrama a seguir ilustra os arquivos usados pelo serviço de dados REST do eXtreme Scale.

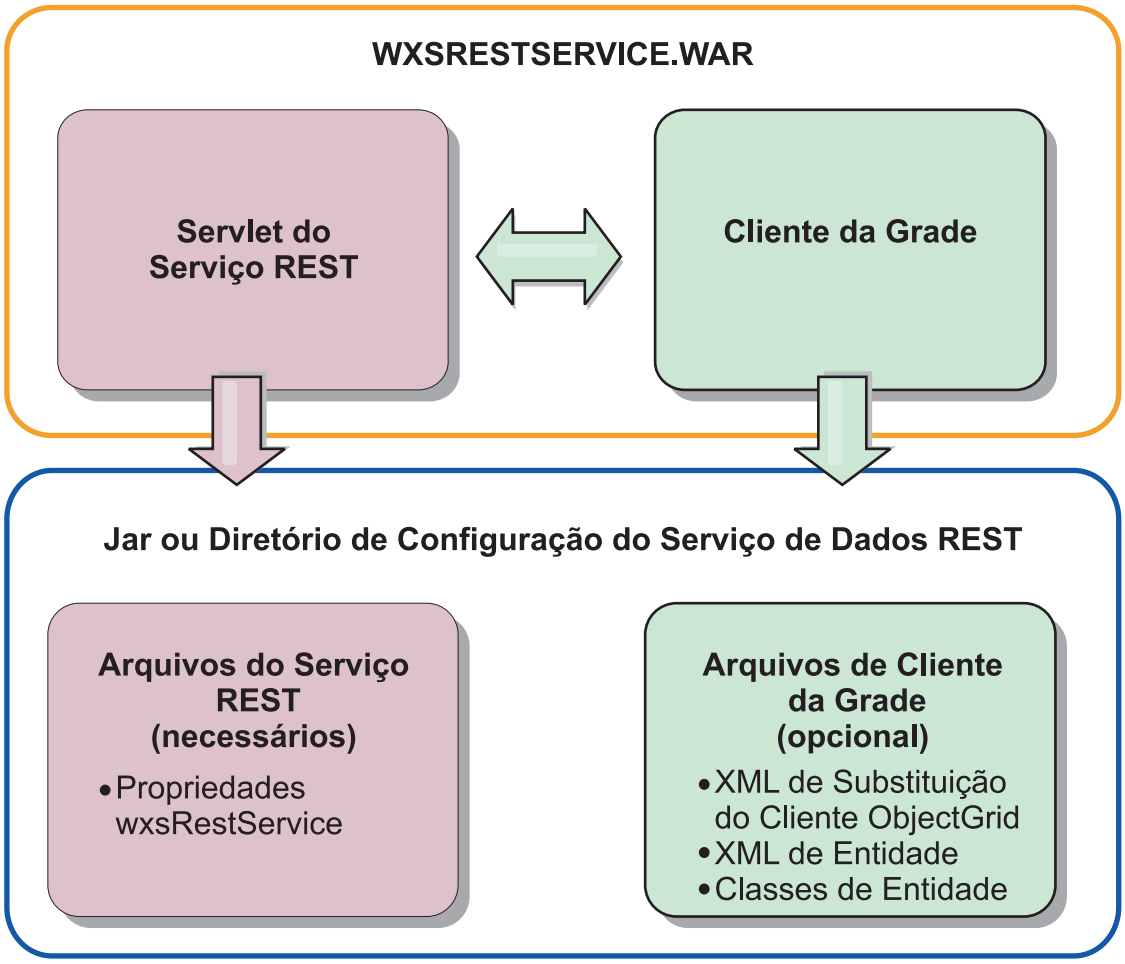


Figura 18. Arquivos do Serviço de Dados REST do WebSphere eXtreme Scale

O diretório ou JAR de configuração do serviço REST deve conter o seguinte arquivo:
`wxsRestService.properties`: O arquivo `wxsRestService.properties` inclui as opções de configuração para o serviço de dados REST. Isso inclui terminais de serviço de catálogo, nomes de ObjectGrid para expor, opções de rastreio e mais. Consulte "Arquivo de Propriedades do Serviço de Dados REST" na página 286.
Os arquivos de cliente do ObjectGrid a seguir são opcionais:

- `META-INF/objectGridClient.xml`: O arquivo XML de substituição do cliente do ObjectGrid é usado para conectar à grade do eXtreme Scale remoto. Por padrão, esse arquivo não é necessário. Se este arquivo não estiver presente, o serviço REST usará a configuração do servidor, desativando o cache próximo.

  O nome do arquivo pode ser substituído com o uso da propriedade de configuração do serviço de dados REST do objectGridClientXML. Se fornecido, esse arquivo XML deverá incluir:

  1. Incluir quaisquer ObjectGrids que você queira expor para o serviço de dados REST.
  2. Inclua uma referência para o arquivo XML do descritor de entidade associado a cada configuração de ObjectGrid.
- `META-INF/`*`arquivos XML do descritor de entidades`*: Um ou mais arquivos XML do descritor de entidades serão necessários apenas se o cliente precisar substituir

a definição de entidade do cliente. O arquivo XML do descritor de entidade deve ser utilizado junto com o arquivo descritor XML de substituição de cliente do ObjectGrid.

Para obter detalhes sobre os arquivos de configuração do eXtreme Scale, consulte o eXtreme Scale *Guia de Administração*..

- **Classes de entidade** Classes de entidade anotadas ou um arquivo XML de descritor de entidade podem ser utilizados para descrever os metadados da entidade. O serviço REST exigirá classes de entidades no caminho da classe apenas se os servidores eXtreme Scale forem configurados com classes de metadados de entidades e um descritor XML de entidade de substituição do cliente não for usado.

  Um exemplo com o arquivo de configuração requerido mínimo, no qual as entidades são definidas em XML nos servidores:

  ```
  restserviceconfig.jar:
  wxsRestService.properties
  ```

  O arquivo de propriedades contém:

  ```
  catalogServiceEndPoints=localhost:2809
  objectGridNames=NorthwindGrid
  ```

  Um exemplo com uma entidade, arquivos XML de substituição e classes de entidade:

  ```
  restserviceconfig.jar:
  wxsRestService.properties
  ```

  O arquivo de propriedades contém:

  ```
  catalogServiceEndPoints=localhost:2809
  objectGridNames=NorthwindGrid
  ```

  ```
  com/acme/entities/Customer.class
  META-INF/objectGridClient.xml
  ```

  O arquivo XML do descritor de ObjectGrid cliente contém:

  ```
  <objectGrid
  name="CustomerGrid" entityMetadataXMLFile="emd.xml"/>
  META-INF/emd.xml
  ```

  O arquivo XML do descritor de metadados da entidade contém:

  ```
  <entity
  class-name="com.acme.entities.Customer" name="Customer"/>
  ```

  Para obter detalhes sobre a API EntityManager e sobre como configurar um cliente e servidor do eXtreme Scale, consulte o *Guia de Administração*.

Implementando o Serviço de Dados REST no WebSphere Application Server

Este tópico descreve como configurar o serviço de dados REST do eXtreme Scale no WebSphere Application Server ou no WebSphere Network Deployment Versão 6.1.0.25 ou posterior. Estas instruções também se aplicam a implementações nas quais o WebSphere eXtreme Scale é integrado à implementação do WebSphere Application Server.

Antes de Iniciar

Você deve ter um dos seguintes ambientes no seu sistema para configurar e implementar o serviço de dados REST para o WebSphere eXtreme Scale.

- WebSphere Application Server com o cliente do eXtreme Scale independente:
  - O WebSphere eXtreme Scale de Teste Versão 7.1 com o serviço de dados REST é transferido por download e extraído ou o produto WebSphere eXtreme Scale 7.1.0.0 com correção acumulativa 2 é instalado em um diretório independente.

– WebSphere Application Server Versão 6.1.0.25 ou 7.0.0.5 ou posterior está instalado e em execução.

- WebSphere Application Server integrado ao WebSphere eXtreme Scale:

  O WebSphere eXtreme Scale Versão 7.1.0.0 com correção acumulativa 2 é instalado na parte superior do WebSphere Application Server Versão 6.1.0.25 ou 7.0 (ou posterior).

  Dica: O serviço de dados REST do eXtreme Scale requer apenas que a opção do cliente do eXtreme Scale seja instalada. O perfil não precisa ser aumentado.

  Leia sobre como ativar a segurança do Java 2 no centro de informações do WebSphere Application Server.

Procedimento

1. Configure e inicie uma grade eXtreme Scale.
   a. Para obter detalhes sobre como configurar uma grade do eXtreme Scale para usar com o serviço de dados REST, consulte Capítulo 6, "Configurando o Ambiente de Implementação", na página 97.
   b. Verifique se um cliente do eXtreme Scale pode conectar e acessar entidades na grade. Para obter um exemplo, consulte a seção Iniciação deste documento.
2. Construa a JAR da configuração do serviço eXtreme Scale REST ou o diretório. Consulte as informações sobre como empacotar e implementar o serviço REST no "Instalando o Serviço de Dados REST" na página 300.
3. Inclua o diretório ou JAR da configuração do serviço de dados REST no caminho de classe do servidor de aplicativos:
   a. Abra o console administrativo do WebSphere
   b. Navegue para **Ambiente** → **Bibliotecas Compartilhadas**
   c. Clique em **Novo**
   d. Inclua as seguintes entradas nos campos apropriados:
      - Nome: `extremescale_rest _configuration`
      - Caminho de classe: <diretório ou jar da configuração do serviço REST>
   e. Clique em **OK**
   f. Salve as mudanças na configuração principal
4. Se o eXtreme Scale estiver integrado à instalação do WebSphere Application Server, ignore esta etapa e vá para a etapa 5. Caso contrário, continue:

   Inclua o JAR de tempo de execução do cliente do WebSphere eXtreme Scale, wsogclient.jar, e o JAR ou o diretório de configuração do serviço de dados REST no caminho de classe do servidor de aplicativos:
   a. Abra o console administrativo do WebSphere
   b. Navegue para **Ambiente** → **Bibliotecas Compartilhadas**
   c. Clique em **Novo**
   d. Inclua as seguintes entradas nos campos:
      - Nome: `extremescale_client_v71`
      - Caminho da classe: `wxs_home/lib/wsogclient.jar`
   e. Clique em OK
   f. Salve as mudanças na configuração principal
5. Instale o arquivo EAR do serviço de dados REST, wxsrestservice.ear, no WebSphere Application Server utilizando o console administrativo do WebSphere:

a. Abra o console administrativo do WebSphere
b. Navegue para Aplicativos -> Novo Aplicativo
c. Navegue para o arquivo /lib/wxsrestservice.ear no sistema de arquivos, selecione-o e clique em **Avançar**.
   - Se estiver utilizando o WebSphere Application Server versão 7.0, clique em Avançar.
   - Se estiver utilizando o WebSphere Application Server versão 6.1, insira um valor de Raiz de Contexto com o nome: /wxsrestservice e continue na próxima etapa.
d. Escolha uma opção de instalação detalhada e clique em Avançar.
e. Na tela de avisos de segurança do aplicativo, clique em Continuar.
f. Escolha as opções de instalação padrão e clique em Avançar.
g. Escolha um servidor para o qual mapear o aplicativo e clique em Avançar.
h. Na página de recarregamento de JSP, utilize os padrões e clique em Avançar.
i. Na página de bibliotecas compartilhadas, mapeie o módulo "wxsrestservice.war" para as seguintes bibliotecas compartilhadas definidas nas etapas 3 e 4:
   - extremescale_rest _configuration
   - extremescale_client_v71

   Dica: Esta biblioteca compartilhada será necessária apenas se o eXtreme Scale não estiver integrado ao WebSphere Application Server.
j. Na página de relacionamento da biblioteca compartilhada do mapa, utilize os padrões e clique em Avançar.
k. Na página de hosts virtuais do mapa, utilize os padrões e clique em Avançar.
l. Na página de raízes de contexto do mapa, configure a raiz de contexto como: wxsrestservice
m. Na página Resumo, clique em Concluir para concluir a instalação.
n. Salve as alterações na configuração principal.

6. Inicie o aplicativo do serviço de dados REST do eXtreme Scale "wxsrestservice":
   a. Escolha um aplicativo
      - Se estiver utilizando o WebSphere Application Server versão 7.0: No console administrativo, clique em **Aplicativos → Tipos de Aplicativo → Aplicativos WebSphere**
      - Se estiver utilizando o WebSphere Application Server versão 6.1: No console administrativo, clique em **AplicativosAplicativos Corporativos**.
   b. Marque a caixa de opção próxima do aplicativo "wxsrestservice " e clique em **Iniciar**.
   c. Revise SystemOut.log para o perfil do servidor de aplicativos. Quando o serviço de dados REST for iniciado com sucesso, a seguinte mensagem será exibida no SystemOut.log para o perfil do servidor:

      ```
      CWOBJ4000I: O serviço de dados REST do WebSphere eXtreme Scale foi iniciado.
      ```

7. Verifique se o serviço de dados do REST está funcionando: O número da porta pode ser localizado em SystemOut.log dentro do diretório de logs do perfil do servidor de aplicativos examinando-se a primeira porta exibida para o identificador de mensagem: SRVE0250I. A porta padrão é 9080.

Por exemplo: http://localhost:9080/wxsrestservice/restservice/NorthwindGrid/ Resultado: O documento de serviço AtomPub é exibido.

Implementando o Serviço de Dados REST no WebSphere Application Server Community Edition

Este tópico descreve como configurar o serviço de dados REST do eXtreme Scale no WebSphere Application Server Community Edition Versão 2.1.1.3 ou posterior.

Antes de Iniciar

- Um IBM (recomendado) ou Sun JRE ou JDK, Versão 5 ou posterior está instalado e a variável de ambiente JAVA_HOME está configurada.
- Faça download e instale o WebSphere Application Server Community Edition Versão 2.1.1.3 ou posterior no diretório `wasce_root`, por exemplo, o diretório `/opt/IBM/wasce`. Leia as instruções de instalação para obter informações sobre o versão 2.1.1 ou outras versões.
- O eXtreme Scale de Teste Versão 7.1 com o serviço de dados REST é transferido por download e extraído ou o produto WebSphere eXtreme Scale 7.1.0.0 com correção acumulativa 2 é instalado em um diretório independente.

Procedimento

1. Configure e inicie uma grade eXtreme Scale.
   a. Para obter detalhes sobre como configurar uma grade do eXtreme Scale para usar com o serviço de dados REST, leia sobre o Capítulo 6, “Configurando o Ambiente de Implementação”, na página 97.
   b. Verifique se um cliente do eXtreme Scale pode conectar e acessar entidades na grade. Para obter um exemplo, consulte Amostra e tutorial dos serviços de dados REST.
2. Construa a JAR da configuração do serviço eXtreme Scale REST ou o diretório. Consulte as informações de empacotamento e implementação no tópico “Instalando o Serviço de Dados REST” na página 300 para obter detalhes.
3. Inicie o servidor WebSphere Application Server Community Edition:
   a. Para iniciar o servidor sem a segurança Java SE ativada, execute o seguinte comando:

      UNIX Linux `wasce_root/bin/startup.sh`

      Windows `wasce_root/bin/startup.bat`

   b. Para iniciar o servidor com a segurança Java SE ativada, siga estas etapas:

      UNIX Linux

      1) Abra uma janela de linha de comandos ou de terminal e execute o seguinte comando de cópia (ou copie o conteúdo do arquivo de política especificado na política existente): `cp restservice_home/gettingstarted/wasce/geronimo.policy wasce_root/bin`
      2) Edite o arquivo `wasce_root/bin/setenv.sh`
      3) Depois da linha que contém `"WASCE_JAVA_HOME="`, inclua o seguinte:
      ```
      export JAVA_OPTS="-Djava.security.manager
      -Djava.security.policy=geronimo.policy"
      ```

      Windows

      1) Abra uma janela de linha de comandos e execute o seguinte comando de cópia (ou copie o conteúdo do arquivo de políticas especificado na política existente):
      ```
      copy restservice_home\gettingstarted\wasce\geronimo.policy\bin
      ```

2) Edite o arquivo `wasce_root\bin\setenv.bat`
3) Após a linha contendo "set WASCE_JAVA_HOME=", inclua o seguinte:

```
set JAVA_OPTS="-Djava.security.manager
-Djava.security.policy=geronimo.policy"
```

4. Inclua o JAR de tempo de execução do cliente ObjectGrid ao repositório WebSphere Application Server Community Edition:
   a. Abra o console administrativo WebSphere Application Server Community Edition e efetue login. A URL padrão é: http://localhost:8080/console, o ID do usuário padrão é "system" e a senha é "manager".
   b. Clique no link **Repositório** no lado esquerdo da janela do console, na pasta **Serviços**.
   c. Na seção **Incluir Archive no Repositório**, preencha o seguinte nas caixas de texto de entrada:

Tabela 14. Incluir Archive no Repositório

Caixa de texto	Valor
File	`wxs_home/lib/ogclient.jar`
Grupo	com.ibm.websphere.xs
Artefato	ogclient
Versão	7.1
Tipo	JAR

   d. Clique no botão Instalar

   Consulte a seguinte nota técnica para obter detalhes sobre as diferentes maneiras pelas quais as dependências de classe e de biblioteca podem ser configuradas:Especificando dependências externas para aplicativos em execução no WebSphere Application Server Community Edition.
5. Implemente e inicie o módulo do serviço de dados REST, o arquivo `wxsrestservice.war`, para o servidor WebSphere Application Server Community Edition.
   a. Copie e edite o arquivo XML do plano de implementação de amostra: `restservice_home/gettingstarted/wasce/geronimo-web.xml` para incluir dependências do caminho para o JAR ou o diretório de configuração do seu serviço de dados REST. Consulte a seção para obter um exemplo sobre como configurar o caminho de classe para incluir seu arquivo `wxsRestService.properties` e outros arquivos de configuração e classes de metadados.
   b. Abra o console administrativo WebSphere Application Server Community Edition e efetue login.

      Dica: A URL padrão é: http://localhost:8080/console. O ID padrão do usuário é "system" e a senha é "manager".
   c. Clique no link **Implementar Novo** no lado esquerdo da janela do console.
   d. Na página **Instalar Novos Aplicativos**, insira os seguintes valores nas caixas de texto:

Tabela 15. Instalar Novos Aplicativos

Caixa de texto	Valor
Archive	restservice_home/lib/wxsrestservice.war
Plano	restservice_home/gettingstarted/wasce/geronimo-web.xml

Dica: Use o caminho para o arquivo `geronimo-web.xml` que você copiou e editou na etapa 3.

e. Clique no botão Instalar. A página do console indica que o aplicativo foi instalado e iniciado com sucesso.

f. Examine o log de saída do sistema ou o console do WebSphere Application Server Community Edition para verificar se o serviço de dados REST foi iniciado com êxito verificando se a seguinte mensagem está presente:

```
CWOBJ4000I: O serviço de dados REST do WebSphere eXtreme Scale foi iniciado.
```

6. Inicie o servidor WebSphere Application Server Community Edition executando o seguinte comando:
   - UNIX Linux `wasce_root/bin/startup.sh`
   - Windows `wasce_root/bin/startup.bat`
7. Instale o serviço de dados REST do eXtreme Scale e a amostra fornecida no servidor WebSphere Application Server Community Edition:
   a. Inclua o JAR de tempo de execução do cliente ObjectGrid ao repositório WebSphere Application Server Community Edition:
      1) Abra o console administrativo WebSphere Application Server Community Edition e efetue login. (As configurações padrão são http://localhost:8080/console/ com ID do usuário system e senha manager.)
      2) Clique no link **Repositório** no lado esquerdo da janela do console, na pasta Serviços.
      3) Na seção **Incluir Archive no Repositório**, preencha o seguinte nas caixas de texto de entrada:

Tabela 16. Incluir Archive no Repositório

Caixa de texto	Valor
File	wxs_home/lib/ogclient.jar
Grupo	com.ibm.websphere.xs
Artefato	ogclient
Versão	7.1
Tipo	JAR

      4) Clique no botão Instalar.

         Dica: Consulte a seguinte nota técnica para obter detalhes sobre as diferentes maneiras pelas quais as dependências de classe e de biblioteca podem ser configuradas: Especificando dependências externas para aplicativos em execução no WebSphere Application Server Community Edition

   b. Implemente o módulo de serviço de dados REST: `wxsrestservice.war` para o servidor WebSphere Application Server Community Edition.
      1) Edite o arquivo XML de implementação `restservice_home/gettingstarted/wasce/geronimo-web.xml` de amostra para incluir dependências do caminho nos diretórios do caminho de classe de amostra de introdução:
         - Altere "classesDirs" para os dois GBeans do cliente de introdução:

O caminho "classesDirs" para o GBean GettingStarted_Client_SharedLib deve ser configurado para: `restservice_home/gettingstarted/restclient/bin`

O caminho "classesDirs" para o GBean GettingStarted_Common_SharedLib deve ser configurado para: `restservice_home/gettingstarted/common/bin`

2) Abra o console administrativo WebSphere Application Server Community Edition e efetue login.
3) Clique no link **Implementar Novo** no lado esquerdo da janela do console.
4) Na página **Instalar Novos Aplicativos**, insira os seguintes valores nas caixas de texto:

Tabela 17. Instalar Novos Aplicativos

Caixa de texto	Valor
Archive	*restservice_home*/lib/wxsrestservice.war
Plano	*restservice_home*/gettingstarted/wasce/geronimo-web.xml

5) Clique no botão **Instalar**.

   A página do console indica que o aplicativo foi instalado e iniciado com sucesso.
6) Examine o log de saída do sistema WebSphere Application Server Community Edition para verificar se o serviço de dados REST foi iniciado com êxito verificando se a seguinte mensagem está presente:

   `CWOBJ4000I: O serviço de dados REST do WebSphere eXtreme Scale REST foi iniciado.`

8. Verifique se o serviço de dados REST está funcionando:

   Abra um navegador da Web e navegue para a seguinte URL: `http://<host>:<port>/<context root>/restservice/<Grid Name>`

   A porta padrão para WebSphere Application Server Community Edition é 8080 e é definida usando a propriedade "HTTPPort" no arquivo `/var/config/config-substitutions.properties`.

   Por exemplo: http://localhost:8080/wxsrestservice/restservice/NorthwindGrid/

Resultados

O documento de serviço AtomPub é exibido.

Implementando o Serviço de Dados REST no Apache Tomcat

Este tópico descreve como configurar o serviço de dados REST do WebSphere eXtreme Scale no Apache Tomcat Versão 5.5 ou posterior.

Sobre Esta Tarefa

- Um IBM ou Sun JRE ou JDK, Versão 5 ou posterior está instalado e uma variável de ambiente JAVA_HOME especificada.
- Apache Tomcat Versão 5.5 ou posterior está instalado. Consulte Apache Tomcat para obter detalhes sobre como instalar o Tomcat.
- O eXtreme Scale de Teste Versão 7. com o serviço de dados REST é transferido por download e extraído ou o WebSphere eXtreme Scale Versão 7.1.0.0 com o produto de correção acumulativa 2 é instalado em um diretório independente.

Procedimento

1. Se estiver utilizando Sun JRE ou JDK, instale IBM ORB no Tomcat:
   a. Tomcat versão 5.5:
      Copie todos os arquivos JAR de:
      o diretório *wxs_home*/lib/endorsed
      para:
      o diretório *tomcat_root*/common/endorsed
   b. Tomcat versão 6.0:
      Crie um diretório "endorsed":
      UNIX Linux `mkdir tomcat_root/endorsed`
      Windows `md tomcat_root/endorsed`
      Copie todos os arquivos JAR de:
      `wxs_home/lib/endorsed`
      para:
      `tomcat_root/common/endorsed`
2. Configure e inicie uma grade eXtreme Scale.
   a. Para obter detalhes sobre como configurar uma grade do eXtreme Scale para usar com o serviço de dados REST, consulte Capítulo 6, “Configurando o Ambiente de Implementação”, na página 97.
   b. Verifique se um cliente do eXtreme Scale pode conectar e acessar entidades na grade. Para obter um exemplo, consulte Amostra e tutorial dos serviços de dados REST.
3. Construa a JAR da configuração do serviço eXtreme Scale REST ou o diretório. Consulte as informações de empacotamento e implementação no “Instalando o Serviço de Dados REST” na página 300 para obter detalhes.
4. Implemente o módulo de serviço de dados REST: wxsrestservice.war para o servidor Tomcat.
   Copie o arquivo wxsrestservice.war de:
   `restservice_home/lib`
   para:
   `tomcat_root/webapps`
5. Inclua o JAR de tempo de execução do cliente ObjectGrid e o JAR do aplicativo no caminho de classe compartilhado no Tomcat:
   a. Edite o arquivo `tomcat_root/conf/catalina.properties`
   b. Anexe os seguintes nomes de caminhos no final da propriedade shared.loader, separando cada nome de caminho por uma vírgula:
      - `wxs_home/lib/ogclient.jar`
      - `restservice_home/gettingstarted/restclient/bin`
      - `restservice_home/gettingstarted/common/bin`
6. Se você estiver usando a segurança Java 2, inclua permissões de segurança para o arquivo de políticas tomcat:
   - Se estiver utilizando Tomcat versão 5.5:
     Mescle o conteúdo do arquivo 5.5 `catalina policy` de amostra localizado em `restservice_home/gettingstarted/tomcat/catalina-5_5.policy` com o arquivo `tomcat_root/conf/catalina.policy`.
   - Se estiver utilizando Tomcat versão 6.0:
     Mescle o conteúdo do arquivo 6.0 `catalina policy` de amostra localizado em

`restservice_home/gettingstarted/tomcat/catalina-6_0.policy` com o arquivo `tomcat_root/conf/catalina.policy`.

7. Inicie o servidor Tomcat:
   - **Se estiver usando Tomcat 5.5 no UNIX ou Windows ou a distribuição ZIP de Tomcat 6.0:**
     a. `cd tomcat_root/bin`
     b. Inicie o servidor:
        - Sem a segurança Java 2 ativada:

          UNIX Linux `./catalina.sh run`

          Windows `catalina.bat run`
        - Com a segurança Java 2 ativada:

          UNIX Linux `./catalina.sh run -security`

          Windows `catalina.bat run -security`

     c. Os logs do Apache Tomcat são exibidos no console. Quando o serviço de dados REST for iniciado com êxito, a seguinte mensagem será exibida no console administrativo:

        ```
        CWOBJ4000I: O serviço de dados REST do WebSphere eXtreme Scale foi iniciado.
        ```
   - **Se estiver usando Tomcat 6.0 no Windows usando a distribuição do Windows Installer:**
     a. cd /bin
     b. Inicie a ferramenta de configuração do Apache Tomcat 6:

        ```
        tomcat6w.exe
        ```
     c. Para ativar a segurança Java 2 (opcional):

        Inclua as seguintes entradas em Opções Java na guia Java na janela de propriedades do Apache Tomcat 6:

        ```
        -Djava.security.manager
        -Djava.security.policy=\conf\catalina.policy
        ```
     d. Clique no botão Iniciar na janela de propriedades do Apache Tomcat 6 para iniciar o servidor Tomcat.
     e. Revise os seguintes logs para verificar se o servidor Tomcat foi iniciado com sucesso:
        - `tomcat_root/bin/catalina.log`

          Exibe o status do mecanismo do servidor Tomcat
        - `tomcat_root/bin/stdout.log`

          Exibe o log de saída do sistema

     f. Quando o serviço de dados REST for iniciado com sucesso, a seguinte mensagem será exibida no log de saída do sistema:

        ```
        CWOBJ4000I: O serviço de dados REST do WebSphere eXtreme Scale foi iniciado.
        ```
8. Verifique se o serviço de dados REST está funcionando.

   Abra um navegador da Web e navegue para a seguinte URL:

   `http://`*`host:port/context_root`*`/restservice/`*`grid_name`*

   A porta padrão para Tomcat é 8080 e é configurada no arquivo `tomcat_root/conf/server.xml` no elemento <Connector>.

   Por exemplo:

   ```
   http://localhost:8080/wxsrestservice/restservice/NorthwindGrid/
   ```

Resultados

O documento de serviço AtomPub é exibido.

Protegendo o Serviço de Dados REST

Proteja vários aspectos do serviço de dados REST. O acesso ao serviço de dados REST do eXtreme Scale pode ser protegido por meio da autenticação e autorização. O acesso também pode ser controlado por regras de configuração com escopo definido de serviço, conhecidas como regras de acesso. A segurança de transporte é a terceira consideração.

Sobre Esta Tarefa

O acesso ao serviço de dados REST do eXtreme Scale pode ser protegido por meio da autenticação e autorização. A autenticação e a autorização são realizadas fazendo a integração com a segurança do eXtreme Scale.

O acesso também pode ser controlado por regras de configuração com escopo definido de serviço, conhecidas como regras de acesso. Existem dois tipos de regras de acesso: direitos de operação de serviço, que controla as operações CRUD que são permitidas pelo serviço; e direitos de acesso de entidade, que controlam as operações CRUD que são permitidas para um tipo de entidade particular.

A segurança de transporte é fornecida pela configuração de contêiner do hosting (para Web client para conexões de serviço REST) e pela configuração de cliente do eXtreme Scale (para serviço REST para conexões da grade do eXtreme Scale).

Procedimento

- Autenticação e autorização de controle.

  O acesso ao serviço de dados REST do eXtreme Scale pode ser protegido por meio da autenticação e autorização. A autenticação e a autorização são realizadas fazendo a integração com a segurança do eXtreme Scale.

  O serviço de dados REST do eXtreme Scale usa a segurança do eXtreme Scale (autenticação e autorização) para controlar quais usuários podem acessar o serviço e as operações que um usuário tem permissão para executar por meio do serviço. O serviço de dados REST do eXtreme Scale usa uma credencial global configurada (usuário e senha) ou uma credencial derivada de um desafio HTTP BASIC que é enviado com cada transação para a grade do eXtreme Scale onde a autenticação e a autorização são executadas.

  1. Configure a autenticação e a autorização do cliente do eXtreme Scale na grade Consulte “Integração de Segurança com Provedores Externos” na página 368 para obter detalhes sobre como configurar a autenticação e a autorização do cliente do eXtreme Scale.
  2. Configure o cliente do eXtreme Scale (usado pelo serviço REST) para segurança.

     O serviço de dados REST do eXtreme Scale chama a biblioteca do cliente do eXtreme Scale ao se comunicar com a grade do eXtreme Scale. Portanto, o cliente do eXtreme Scale deve ser configurado para a segurança do eXtreme Scale.

     A autenticação de cliente do eXtreme Scale é ativada por meio de propriedades no arquivo de propriedades do cliente objectgrid. No mínimo, os seguintes atributos devem ser ativados ao usar a segurança do cliente com o serviço REST:

```
securityEnabled=true
credentialAuthentication=Supported [-or-] Required
credentialGeneratorProps=user:pass [-or-] {xor encoded user:pass}
```

Observe, o usuário e a senha especificados na propriedade credentialGeneratorProps devem ser mapeados para um ID no registro de autenticação e devem ter direitos de políticas de ObjectGrid suficientes para conectar e criar ObjectGrids.

Um arquivo de políticas do cliente de objectgrid de amostra está localizado em `wxsrest_home/security/security.ogclient.properties`. Consulte também “Arquivo de Propriedades do Cliente” na página 202.

3. Configure o serviço de dados REST do eXtreme Scale para segurança.

   O arquivo de propriedades de configuração do serviço de dados REST do eXtreme Scale deve conter as seguintes entradas para ser integrado com a segurança do eXtreme Scale:

   ```
   ogClientPropertyFile=file_name
   ```

   O ogClientPropertyFile é o local do arquivo de propriedades que contém propriedades do cliente de ObjectGrid mencionadas na etapa anterior. O serviço REST usa este arquivo para inicializar o cliente do eXtreme Scale para falar com a grade quando a segurança estiver ativada.

   ```
   loginType=basic [-or-] none
   ```

   A propriedade loginType configura o serviço REST para o tipo de login. Se um valor de "none" for especificado, o ID e a senha de usuário “global” definidos por credentialGeneratorProps serão enviados para a grade para cada transação. Se um valor de “basic” for especificado, o serviço REST apresentará um desafio HTTP BASIC para o cliente solicitando credenciais que ele enviará em cada transação ao se comunicar com a grade.

   Para obter mais informações sobre as propriedades ogClientPropertyFile e loginType, consulte “Arquivo de Propriedades do Serviço de Dados REST” na página 286.

- Aplique as regras de acesso.

  O acesso também pode ser controlado por regras de configuração com escopo definido de serviço, conhecidas como regras de acesso. Existem dois tipos de regras de acesso: direitos de operação de serviço, que controla as operações CRUD que são permitidas pelo serviço; e direitos de acesso de entidade, que controlam as operações CRUD que são permitidas para um tipo de entidade particular.

  O serviço de dados REST do eXtreme Scale permite, opcionalmente, regras de acesso que podem ser configuradas para restringir o acesso ao serviço e a entidades no serviço. Essas regras de acesso são especificadas no arquivo de propriedades de direitos de acesso do serviço REST. O nome deste arquivo é especificado no arquivo de propriedades do serviço de dados REST pela propriedade “wxsRestAccessRightsFile”, conforme mostrado na Seção 5.1. Este arquivo é um arquivo de propriedades Java típico com pares de chave e valor. Existem dois tipos de regras de acesso: direitos de operação de serviço, que controlam as operações CRUD que são permitidas pelo serviço; e direitos de acesso de entidade, que controlam as operações CRUD que são permitidas para um tipo de entidade particular.

  1. Configure os direitos de operação de serviço.

     Os direitos de operações de serviço especificam direitos de acesso que se aplicam a todos os ObjectGrids expostos por meio do serviço REST ou a todas as entidades de um ObjectGrid individual, conforme especificado.

     Use a seguinte sintaxe.

```
serviceOperationRights=service_operation_right
serviceOperationRights.grid_name -OR- *=service_operation_right
```

em que

– serviceOperationRights pode ser uma das opções a seguir: [NONE, READSINGLE, READMULTIPLE, ALLREAD, ALL]

– serviceOperationRights.*grid_name -OR-* * implica que o direito de acesso se aplica a todos os ObjectGrids, também o nome de um ObjectGrid específico pode ser fornecido.

Por exemplo:

```
serviceOperationsRights=ALL
serviceOperationsRights.*=NONE
serviceOperationsRights.EMPLOYEEGRID=READSINGLE
```

O primeiro exemplo especifica que todas as operações de serviço são permitidas para todos os ObjectGrids expostos por este Serviço REST. O segundo exemplo é semelhante ao primeiro, uma vez que também se aplica a todos os ObjectGrids expostos pelo serviço REST; no entanto, ele especifica o direito de acesso como NONE, o que significa que nenhuma das operações de serviço são permitidas nos ObjectGrids. O último exemplo especifica como controlar as operações de serviço para uma grade específica, aqui apenas Reads, que resultam em um único registro, são permitidas para todas as entidades do EMPLOYEEGRID.

O padrão assumido pelo serviço REST é serviceOperationsRights=ALL, que significa que todas as operações são permitidas para todos os ObjectGrids expostos por este serviço. Isto é diferente da implementação Microsoft, na qual eles escolhem o padrão para NONE, assim nenhuma operação é permitida no Serviço REST.

Nota: Os direitos de operações de serviço são avaliados na ordem em que são especificados neste arquivo, portanto, o último direito especificado substituirá os direitos que o precedem.

2. Configure os direitos de acesso da entidade.

   Os direitos do conjunto de entidades especifica os direitos de acesso que se aplicam a entidades do ObjectGrid específicas expostas por meio do serviço REST. Esses direitos fornecem uma maneira de impor um controle de acesso melhor e mais refinado sobre entidades do ObjectGrid individuais que o comparado aos direitos de Operações de Serviço.

   Use a seguinte sintaxe.

   ```
   entitySetRights.grid_name.entity_name=entity_set_right
   ```

   sendo que

   – *entity_set_right* pode ser um dos direitos a seguir.

Tabela 18. Direitos de Acesso de Entidade. Valores suportados.

Direito de acesso	Descrição
NENHUM	Nega todos os direitos para acessar dados
READSINGLE	Permite a leitura de itens de dados únicos
READMULTIPLE	Permite a leitura de conjuntos de dados
ALLREAD	Permite a leitura de um único ou de vários conjuntos de dados
WRITEAPPEND	Permite a criação de novos itens de dados em conjuntos de dados

Tabela 18. Direitos de Acesso de Entidade (continuação). Valores suportados.

Direito de acesso	Descrição
WRITEREPLACE	Permite a substituição de dados
WRITEDELETE	Permite a exclusão de itens de dados dos conjuntos de dados
WRITEMERGE	Permite a mesclagem de dados
ALLWRITE	Permite a gravação (isto é, criação, substituição, mesclagem ou exclusão) de dados
ALL	Permite a criação, leitura, atualização e exclusão de dados

  - *entity_name* é o nome de um ObjectGrid específico dentro do serviço REST.
  - *grid_name* é o nome de uma entidade específica dentro do ObjectGrid especificado.

  Nota: Se os direitos de operação de serviço e os direitos do conjunto de entidades forem especificados para um ObjectGrid respectivo e suas entidades, o mais restritivo desses direitos será reforçado, conforme ilustrado nos exemplos a seguir. Observe também que os direitos do conjunto de entidades são avaliados na ordem em que são especificados no arquivo. O último direito especificado substituirá os direitos que o precedem.

  Exemplo 1: Se serviceOperationsRights.NorthwindGrid=READSINGLE e entitySetRights.NorthwindGrid.Customer=ALL forem especificados. READSINGLE será reforçado para a entidade Cliente.

  Exemplo 2: Se serviceOperationsRights.NorthwindGrid=ALLREAD for especificado e entitySetRights.NorthwindGrid.Customer=ALLWRITE for especificado, apenas Reads será permitido para todas as entidades de NorthwindGrid. No entanto, para Cliente, seus direitos do conjunto de entidades impedirão quaisquer Reads (uma vez que especificou ALLWRITE) e, consequentemente, com efeito, a entidade Cliente terá o direito de acesso como NONE.

- Transportes seguros.

  A segurança de transporte é fornecida pela configuração de contêiner do hosting (para Web client para conexões de serviço REST) e pela configuração de cliente do eXtreme Scale (para serviço REST para conexões da grade do eXtreme Scale).

  1. Proteja a conexão do cliente e do serviço REST. A segurança de transporte para esta conexão é fornecida pelo ambiente de contêiner do hosting, não no eXtreme Scale.
  2. Proteja a conexão do serviço REST e da grade do eXtreme Scale. A segurança de transporte para esta conexão é configurada no eXtreme Scale. Consulte "Transport Layer Security e Secure Sockets Layer" na página 363.

Capítulo 7. Operando o Ambiente de Implementação

Operar o ambiente do produto inclui iniciar e parar servidores no modo independente ou no WebSphere Application Server. Também é possível utilizar WebSphere eXtreme Scale como um gerenciador de sessões em um ambiente do WebSphere Application Server.

Terminologia Administrativa

Antes de administrar o WebSphere eXtreme Scale, fique ciente desses fatos.

Sobre Esta Tarefa

Tipos de Servidor

O WebSphere eXtreme Scale possui dois tipos de servidores: *servidores de catálogo* e *servidores de contêiner*. Os servidores de catálogo controlam o posicionamento de shards e descobrem e monitoram os servidores de contêiner. Múltiplos servidores de catálogo juntos compõem o *serviço de catálogo*. Os servidores de contêiner são o Java Virtual Machines que armazenam os dados do aplicativo para a grade.

Modo independente

O modo independente é uma configuração do WebSphere eXtreme Scale que está em execução sozinha, sem nenhum outro produto do servidor de aplicativos.

Executando dentro do WebSphere Application Server

Ao executar o WebSphere eXtreme Scale na parte superior do WebSphere Application Server, os servidores de catálogo automaticamente iniciam nos servidores do WebSphere Application Server. Para iniciar os servidores de contêiner, você deve compactar e implementar o seu aplicativo com arquivos XML do ObjectGrid.

Contêineres, Partições e Shards

O contêiner é um serviço que armazena dados do aplicativo para a grade. Estes dados geralmente são divididos em partes, que são chamadas de partições e hospedadas em vários contêineres. Cada contêiner, por sua vez, hospeda um subconjunto de dados completos. Uma JVM pode hospedar um ou mais contêineres e cada contêiner pode hospedar vários shards.

Lembre-se: Planeje o tamanho do heap para os contêineres, que hospedam todos os dados. Configure as configurações de heap apropriadamente.

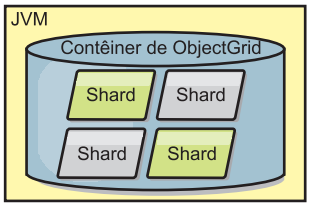


Figura 19. Contêiner

Partições hospedam um subconjunto de dados na grade. O WebSphere eXtreme Scale automaticamente coloca várias partições em um único contêiner e propaga as partições à medida que mais contêineres se tornam disponíveis.

Importante: Escolha o número de partições cuidadosamente antes da implementação final já que o número de partições não pode ser alterado dinamicamente. Um mecanismo hash é utilizado para localizar partições na rede e o eXtreme Scale não pode re-hash o conjunto de dados inteiro após ele ser implementado. Como uma regra geral, é possível superestimar o número de partições

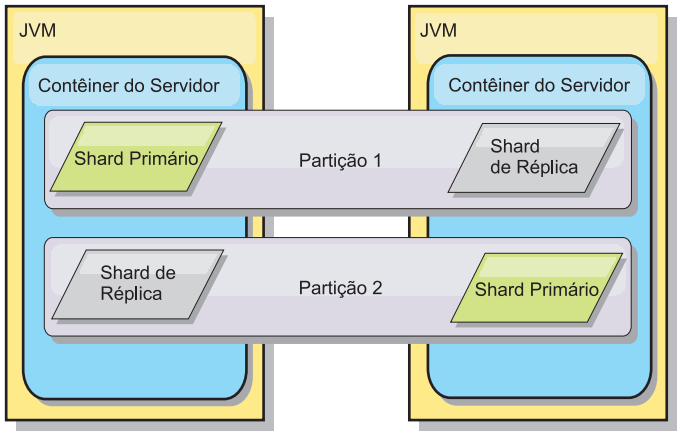


Figura 20. Partição

Os shards são instâncias de partições e possuem uma das duas funções: primário ou de réplica. O fragmento primário e suas réplicas constituem a manifestação física da partição. Cada partição tem vários shards que hospedam, cada um deles, todos os dados contidos nessa partição. Um shard é o primário e os outros são réplicas, que são cópias redundantes dos dados no primeiro shard. Um fragmento primário é a única instância da partição que permite que as transações sejam gravadas no cache. Um fragmento de réplica é uma instância "espelhada" da partição. Ele recebe atualizações de forma síncrona e assíncrona do fragmento primário. O fragmento de réplica permite apenas a leitura das transações do cache. As réplicas nunca são hospedadas no mesmo contêiner que as primárias e normalmente não são hospedadas na mesma máquina que as primárias.

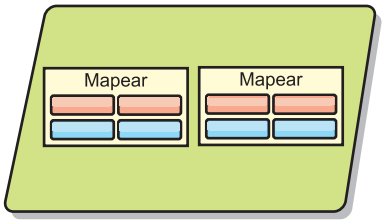


Figura 21. Shard

Para aumentar a disponibilidade dos dados, ou aumentar garantias de persistência, replique os dados. Entretanto, a replicação inclui custo na transação e troca desempenho em retorno para a disponibilidade. Com o eXtreme Scale, é possível controlar o custo já que ambas as replicações síncrona e assíncrona são suportadas, bem como modelos de replicação híbrida utilizando os modos de replicação síncrona e assíncrona. O shard da réplica síncrona recebe atualizações como parte da transação do shard primário para garantir consistência de dados. Uma réplica síncrona pode duplicar o tempo de resposta porque a transação precisa executar commit na réplica primária e na réplica síncrona antes da transação ser concluída. Um shard de réplica assíncrono recebe atualizações após o commit da transação para limitar o impacto no desempenho, mas introduz a possibilidade de perda de dados enquanto que a réplica assíncrona pode ser diversas transações atrás do shard primário.

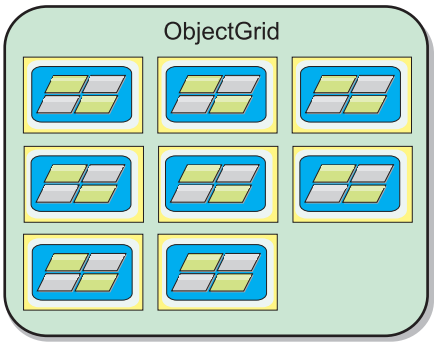


Figura 22. ObjectGrid

Serviços de Catálogos (Servidores de Catálogos)

O serviço de catálogo hospeda lógica que deve estar inativa durante um estado estável e tem pouca influência na escalabilidade. O serviço de catálogo é construído para atender a centenas de contêineres que se tornam disponíveis simultaneamente e executa serviços para gerenciar os contêiners.

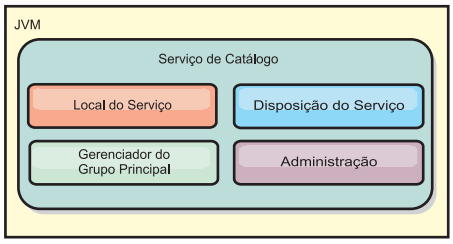


Figura 23. Serviço de Catálogo

As responsabilidades de catálogo consistem nos seguintes serviços:

Serviço de local
O serviço de local fornece localidade para clientes que estão procurando aplicativos de hosting de contêineres e para contêineres que estão procurando registrar aplicativos hospedados com o serviço de disposição. O serviço de local é executado em todos os membros da grade para efetuar o scale out desta função.

Serviço de disposição
O serviço de disposição é o sistema nervoso central para a grade e é responsável por alocar shards individuais para seu contêiner de host. O serviço de posicionamento é executado como Um de N serviços eleitos no cluster. Como a política Um de N é usada, há sempre exatamente uma instância do serviço de posicionamento em execução. Se tal instância deve ser interrompida, outro processo assume. Todos os estados do serviço de catálogo são replicados em todos os servidores hospedando o serviço de catálogo para redundância.

Gerenciador de grupo principal
O gerenciador de grupo principal gerencia agrupamento peer para monitoramento de funcionamento, organiza contêineres em grupos menores de servidores e automaticamente federa os grupos de servidores. Quando um contêiner entra em contato com o serviço de catálogo pela primeira vez, ele aguarda para ser designado a um grupo novo ou existente de várias Java virtual machines (JVM). Cada grupo das Java virtual machines monitora a disponibilidade de cada um dos membros por meio da pulsação. Um destes membros do grupo retransmite as informações de disponibilidade ao serviço de catálogo para permitir reação a falhas por meio de realocação e encaminhamento de rotas.

Administração
Os quatro estágios de administração do seu ambiente do WebSphere eXtreme Scale são planejamento, implementação, gerenciamento e monitoramento. Consulte o *Guia de Administração* para obter mais informações sobre cada estágio.

Para disponibilidade, configure um domínio do serviço de catálogo. Um domínio do serviço de catálogo consiste em várias Java virtual machines, incluindo uma JVM principal e várias Java virtual machines de backup.

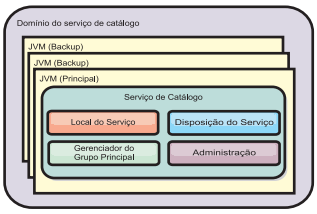


Figura 24. Domínio do Serviço de Catálogo

Configurando a Disponibilidade de um ObjectGrid

O estado de disponibilidade de uma instância do ObjectGrid determina quais pedidos podem ser processados em qualquer momento específico.

Há quatro estados de disponibilidade:

- ONLINE
- QUIESCE
- OFFLINE
- PRELOAD

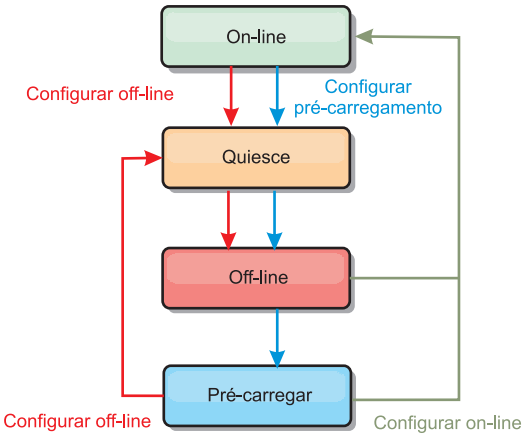


Figura 25. Estados de Disponibilidade de um ObjectGrid

Configurando o Estado de Disponibilidade

O estado padrão de disponibilidade de um ObjectGrid é ONLINE. Um ObjectGrid ONLINE é capaz de processar qualquer pedido de um cliente típico do eXtreme Scale. Entretanto, os pedidos de um cliente de pré-carregamento são rejeitados enquanto o ObjectGrid está ONLINE.

O estado QUIESCE é um estado de transição. Um ObjectGrid que está em QUIESCE, em breve, estará no estado OFFLINE. Enquanto em QUIESCE, um ObjectGrid terá permissão para processar transações pendentes. Entretanto, qualquer nova transação será rejeitada. Um ObjectGrid pode permanecer em QUIESCE por até 30 segundos. Depois deste tempo, o estado de disponibilidade será alterado para OFFLINE.

Um ObjectGrid no estado OFFLINE rejeitará todas as transações.

O estado PRELOAD pode ser utilizado para carregar dados em um ObjectGrid a partir de um cliente de pré-carregamento. Enquanto o ObjectGrid está em um estado PRELOAD, apenas um cliente de pré-carregamento poderá executar o commit em transações junto ao ObjectGrid. Todas as outras transações serão rejeitadas.

Use a interface StateManager para configurar o estado de disponibilidade de um ObjectGrid. Para configurar o estado de disponibilidade de um ObjectGrid em execução nos servidores, transmita um cliente ObjectGrid correspondente para a interface StateManager. O código a seguir demonstra como alterar o estado de disponibilidade de um ObjectGrid.

```
ClientClusterContext client = ogManager.connect("localhost:2809", null, null);
ObjectGrid myObjectGrid = ogManager.getObjectGrid(client, "myObjectGrid");
StateManager stateManager = StateManagerFactory.getStateManager();
stateManager.setObjectGridState(AvailabilityState.OFFLINE, myObjectGrid);
```

Cada shard de ObjectGrid muda para o estado desejado quando o método setObjectGridState é chamado na interface StateManager. Quando o método retornar, todos os shards no ObjectGrid deverão estar no estado adequado.

Use um plug-in ObjectGridEventListener para alterar o estado de disponibilidade de um ObjectGrid do lado do servidor. Altere o estado de disponibilidade de um ObjectGrid do lado do servidor apenas quando o ObjectGrid tiver uma única partição. Se o ObjectGrid tiver múltiplas partições, o método shardActivated será chamado em cada primário, o que resultará em chamadas desnecessárias para alterar o estado do ObjectGrid

```
public class OGListener implements ObjectGridEventListener,
ObjectGridEventGroup.ShardEvents {
    public void shardActivated(ObjectGrid grid) {
            StateManager stateManager = StateManagerFactory.getStateManager();
            stateManager.setObjectGridState(AvailabilityState.PRELOAD, grid);
    }
}
```

Como QUIESCE é o estado transicional, não é possível usar a interface StateManager para colocar um ObjectGrid em um estado QUIESCE. Um ObjectGrid transmite passa por este estado em seu caminho para o estado OFFLINE.

Recuperando o Estado de Disponibilidade

Use o método getObjectGridState da interface StateManager para recuperar o estado de disponibilidade de um ObjectGrid específico.

```
StateManager stateManager = StateManagerFactory.getStateManager();
AvailabilityState state = stateManager.getObjectGridState(inventoryGrid);
```

O método getObjectGridState escolhe um shard primário aleatório no ObjectGrid e retorna seu AvailabilityState. Como todos os shards de um ObjectGrid devem estar no mesmo estado de disponibilidade ou em transição para o mesmo estado de disponibilidade, este método fornece um resultado aceitável para o estado de disponibilidade atual do ObjectGrid.

Estados de Disponibilidade Apropriados para Vários Pedidos

Um pedido será rejeitado se um ObjectGrid não estiver no estado de disponibilidade apropriado para suportar tal pedido. Uma exceção AvailabilityException resulta sempre na rejeição de um pedido.

O atributo initialState

é possível usar o atributo initialState em um ObjectGrid para indicar seu estado de inicialização. Normalmente, quando um ObjectGrid conclui sua inicialização, ele está disponível para roteamento. O estado pode ser posteriormente alterado para evitar o tráfego de roteamento para um ObjectGrid. Se o ObjectGrid precisar ser inicializado, mas não ficar imediatamente disponível, é possível utilizar o atributo initialState.

O atributo initialState é configurado no arquivo XML de configuração do ObjectGrid. O estado padrão é ONLINE. Os valores válidos incluem:

- ONLINE (padrão)
- PRELOAD
- OFFLINE

Conuslte a documentação da API AvailabilityState para obter informações adicionais.

Se initialState estiver configurado em um ObjectGrid, o estado deve ser explicitamente configurado de volta como online ou o ObjectGrid permanecerá indisponível. Ele lançará AvailabilityExceptions.

Utilizando o atributo initialState para pré-carregamento

Se o ObjectGrid for pré-carregado com dados, pode haver um período de tempo entre quando o ObjectGrid está disponível e a mudança para um estado pré-carregado para bloquear o tráfego do cliente. Para evitar este período de tempo, o estado inicial em um ObjectGrid pode ser configurado como PRELOAD. O ObjectGrid ainda conclui todas as inicializações necessárias, mas ele bloqueia o tráfego até que o estado mude e permite que o pré-carregamento ocorra.

Ambos os estados PRELOAD e OFFLINE bloqueiam o tráfego, mas você deve utilizar o estado PRELOAD se deseja iniciar um pré-carregamento.

Comportamento de failover e equilíbrio

Se uma réplica for promovida para um primário, ela não utilizará a configuração initialState. Se o shard primário for movido para um reequilíbrio, a configuração initialState não será usada porque os dados são copiados para o novo local do shard primário antes de concluir a movimentação. Se a replicação não for configurada, então o primário assume o valor initialState se ocorrer failover e um novo primário tiver que ser posicionado.

Utilizando a API do Servidor Integrado

Com o WebSphere eXtreme Scale, é possível usar uma API programática para gerenciar o ciclo de vida de servidores e contêineres integrados. É possível configurar programaticamente o servidor com qualquer uma das opções que também podem ser configuradas com as opções de linha de comandos ou com as propriedades de servidor baseadas em arquivo. É possível configurar o servidor integrado para que seja um servidor de contêiner, um serviço de catálogo ou ambos.

Antes de Iniciar

É necessário ter um método para executar o código a partir de um Java Virtual Machine já existente. As classes doeXtreme Scale deverão estar disponíveis na árvore do carregador de classes.

Sobre Esta Tarefa

É possível executar muitas tarefas de administração com a API de Administração. Um uso comum da API é como um servidor interno para armazenar o estado do aplicativo da Web. O servidor da Web pode iniciar um servidorWebSphere eXtreme Scale integrado, relatar o servidor de contêiner para o serviço de catálogo e o servidor será, em seguida, incluído como um membro de uma grade distribuída maior. Esse uso pode fornecer escalabilidade e alta disponibilidade para um armazém de dados voláteis do contrário.

É possível controlar programaticamente o ciclo de vida completo de um servidor eXtreme Scale integrado. Os exemplos são genéricos o máximo possível e mostram apenas exemplos de código diretos para etapas realçadas.

Procedimento

1. Obtenha o objeto ServerProperties da classe ServerFactory e configure quaisquer opções necessárias.

   Cada servidor eXtreme Scale possui um conjunto de propriedades configuráveis.Quando um servidor é iniciado na linha de comandos, essas propriedades são configuradas para os padrões, mas é possível substituir várias propriedades ao fornecer uma origem ou arquivo externo. No escopo integrado, é possível configurar diretamente as propriedades com um objeto ServerProperties. Essas propriedades devem ser configuradas antes de obter uma instância do servidor a partir da classe ServerFactory. O seguinte exemplo de fragmento de código obtém um objeto ServerProperties, configura o campo CatalogServiceBootStrap e inicializa várias configurações do servidor opcionais. Consulte a documentação da API para obter uma lista de definições configuráveis.

   ```
   ServerProperties props = ServerFactory.getServerProperties();
   props.setCatalogServiceBootstrap("host:port"); // required to connect to specific catalog service
   props.setServerName("ServerOne"); // name server
   props.setTraceSpecification("com.ibm.ws.objectgrid=all=enabled"); // Sets trace spec
   ```

2. Se desejar que o servidor seja um serviço de catálogo, obtenha o objeto CatalogServerProperties.

   Cada servidor integrado pode ser um serviço de catálogo, um servidor de contêiner ou ambos. O seguinte exemplo obtém o objeto CatalogServerProperties, ativa a opção de serviço de catálogo e define várias configurações de serviço de catálogo.

   ```
   CatalogServerProperties catalogProps = ServerFactory.getCatalogProperties();
   catalogProps.setCatalogServer(true); // false by default, it is required to set as a catalog service
   catalogProps.setQuorum(true); // enables / disables quorum
   ```

3. Obtenha a instância Servidor a partir da classe ServerFactory. A instância doServidor é um singleton de escopo definido de processo responsável por gerenciar a associação na grade. Depois que essa instância for inicializada, esse processo será conectado e altamente disponibilizado com outros servidores na grade. A seguir há um exemplo de como criar a instância Servidor:

   ```
   Server server = ServerFactory.getInstance();
   ```

   Revisando o exemplo anterior, a classeServerFactory fornece um método estático que retorna uma instância Servidor. A classe ServerFactory deve ser a

única interface para obter uma instância Servidor. Portanto, a classe garante que a instância seja um singleton ou uma instância para cada JVM ou carregador de classes isolado. O método getInstance inicializa a instância Servidor. É necessário configurar todas as propriedades do servidor antes de inicializar a instância. A classe Servidor é responsável por criar novas instâncias do Contêiner. É possível usar ambas as classes ServerFactory e Servidor para gerenciar o ciclo de vida da instância do Servidor integrada.

4. Inicie uma instância Contêiner usando a instância do Servidor.

   Antes que os shards possam ser colocados em um servidor integrado, é necessário criar um contêiner no servidor. A interface Servidor possui um método createContainer que utiliza um argumentoDeploymentPolicy. O seguinte exemplo utiliza a instância do servidor obtida para criar um contêiner usando um arquivo DeploymentPolicy criado. Note que os contêineres requerem um carregador de classes que possui os binários do aplicativo disponíveis para serialização. É possível tornar esses binários disponíveis ao chamar o métodocreateContainer com o carregador de classes de contexto Encadeamento para configurar o carregador de classes que deseja usar.

   ```
   DeploymentPolicy policy = DeploymentPolicyFactory.createDeploymentPolicy
   (new URL("file://urltodeployment.xml"),
    new URL("file://urltoobjectgrid.xml"));
   Container container = server.createContainer(policy);
   ```

5. Remover e limpar um contêiner.

   É possível remover e limpar um servidor de contêiner usando o método teardown em execução na instância do Contêiner obtida. Executar o método teardown adequadamente em um contêiner limpa o contêiner e remove o contêiner do servidor integrado.

   O processo de limpeza do contêiner inclui o movimento e a remoção de todos os shards que são colocados dentro desse contêiner. Cada servidor pode conter muitos contêineres e shards. Limpar um contêiner não afeta o ciclo de vida da instância do Servidor pai. O seguinte exemplo demonstra como executar o método teardown em um servidor. O método teardown é disponibilizado por meio da interface ContainerMBean. Ao usar a interface ContainerMBean, se você não tiver mais acesso programático nesse contêiner, ainda poderá remover e limpar o contêiner com o MBean. Um métodoterminate também existe na interface Contêiner, mas não use-o a menos que realmente seja necessário. Esse método é mais forte e não coordena o movimento e a limpeza apropriados dos shards.

   ```
   container.teardown();
   ```

6. Parar o servidor integrado.

   Ao parar um servidor integrado, todos os contêineres e shards que estiverem em execução também são parados no servidor. Ao parar um servidor integrado, é necessário limpar todas as conexões abertas e mover ou remover todos os shards. O seguinte exemplo mostra como parar um servidor e o uso do métodowaitFor na interface do Servidor para garantir que a instância do Servidor seja parada completamente. Da mesma forma que o exemplo de contêiner, o método stopServer é disponibilizado por meio da interface ServerMBean. Com essa interface, é possível parar um servidor com o bean gerenciado (MBean) correspondente.

   ```
   ServerFactory.stopServer();  // Uses the factory to kill the Server singleton
    // or
   server.stopServer(); // Uses the Server instance directly
   server.waitFor(); // Returns when the server has properly completed its shutdown procedures
   ```

   Código de exemplo completo:

   ```
    import java.net.MalformedURLException;
   import java.net.URL;
   ```

```
import com.ibm.websphere.objectgrid.ObjectGridException;
import com.ibm.websphere.objectgrid.deployment.DeploymentPolicy;
import com.ibm.websphere.objectgrid.deployment.DeploymentPolicyFactory;
import com.ibm.websphere.objectgrid.server.Container;
import com.ibm.websphere.objectgrid.server.Server;
import com.ibm.websphere.objectgrid.server.ServerFactory;
import com.ibm.websphere.objectgrid.server.ServerProperties;

public class ServerFactoryTest {

    public static void main(String[] args) {

        try {

            ServerProperties props = ServerFactory.getServerProperties();
            props.setCatalogServiceBootstrap("catalogservice-hostname:catalogservice-port");
            props.setServerName("ServerOne"); // name server
            props.setTraceSpecification("com.ibm.ws.objectgrid=all=enabled"); // TraceSpec

            /*
             * Na maioria dos casos, o servidor atuará apenas como um servidor de contêiner
             * e será conectado a um serviço de catálogo externo. Essa é a forma mais
             * disponibilizada de fazer as coisas. O trecho de código comentado a seguir
             * ativará esse Servidor como um serviço de catálogo.
             *
             *
             * CatalogServerProperties catalogProps =
             * ServerFactory.getCatalogProperties();
             * catalogProps.setCatalogServer(true); // enable catalog service
             * catalogProps.setQuorum(true); // enable quorum
             */

            Server server = ServerFactory.getInstance();

            DeploymentPolicy policy =
DeploymentPolicyFactory.createDeploymentPolicy
     (new URL("url to deployment xml"),  new URL("url to objectgrid
xml file"));
            Container container = server.createContainer(policy);

            /*
             * O shard será agora colocado nesse contêiner se os requisitos de
implementação forem atendidos.
             * Isso inclui a criação de servidor e de contêiner integrado.
             *
             * As linhas abaixo demonstrarão simplesmente a chamada
de métodos de limpeza
             */

            container.teardown();
            server.stopServer();
            int success = server.waitFor();

        } catch (ObjectGridException e) {
            // Container failed to initialize
        } catch (MalformedURLException e2) {
            // invalid url to xml file(s)
        }

    }

}
```

API do Servidor Integrado

O WebSphere eXtreme Scale inclui interfaces de programação de aplicativos (APIs) e interfaces de programação do sistema para integrar clientes e servidores eXtreme Scale dentro de seus aplicativos Java existentes. O tópico a seguir descreve as APIs de servidor integradas disponíveis.

Instanciando o Servidor eXtreme Scale

É possível utilizar várias propriedades para configurar a instância do servidor eXtreme Scale, que pode ser recuperada do método ServerFactory.getServerProperties. O objeto ServerProperties é um singleton, portanto, cada chamada para o método getServerProperties recupera a mesma instância.

É possível criar um novo servidor com o seguinte código.

```
Server server = ServerFactory.getInstance();
```

Todas as propriedades configuradas antes da primeira chamada de getInstance são utilizadas para inicializar o servidor.

Configurando Propriedade de Servidor

É possível configurar as propriedades do servidor até que ServerFactory.getInstance seja chamado pela primeira vez. A primeira chamada do método getInstance instancia o servidor eXtreme Scale e lê todas as propriedades configuradas. Configurar as propriedades após a criação não tem efeito. O exemplo a seguir mostra como configurar propriedades antes de criar uma instância do Servidor.

```
// Get the server properties associated with this process.
ServerProperties serverProperties = ServerFactory.getServerProperties();

// Set the server name for this process.
serverProperties.setServerName("EmbeddedServerA");

// Set the name of the zone this process is contained in.
serverProperties.setZoneName("EmbeddedZone1");

// Set the end point information required to bootstrap to the catalog service.
serverProperties.setCatalogServiceBootstrap("localhost:2809");

// Set the ORB listener host name to use to bind to.
serverProperties.setListenerHost("host.local.domain");

// Set the ORB listener port to use to bind to.
serverProperties.setListenerPort(9010);

// Turn off all MBeans for this process.
serverProperties.setMBeansEnabled(false);

Server server = ServerFactory.getInstance();
```

Integrando o Serviço de Catálogo

Qualquer configuração JVM que é sinalizada pelo método CatalogServerProperties.setCatalogServer pode hospedar o serviço de catálogo para o eXtreme Scale. Este método indica ao tempo de execução do servidor do eXtreme Scale para instanciar o serviço de catálogo quando o servidor for iniciado. O código a seguir mostra como instanciar o servidor de catálogos do eXtreme Scale:

```
CatalogServerProperties catalogServerProperties =
 ServerFactory.getCatalogProperties();
catalogServerProperties.setCatalogServer(true);

Server server = ServerFactory.getInstance();
```

Integrando o Contêiner do eXtreme Scale

Emita o método Server.createContainer para qualquer JVM para hospedar vários contêineres do eXtreme Scale. O código a seguir mostra como instanciar um contêiner do eXtreme Scale:

```
Server server = ServerFactory.getInstance();
DeploymentPolicy policy = DeploymentPolicyFactory.createDeploymentPolicy(
    new
```

```
File("META-INF/embeddedDeploymentPolicy.xml").toURI().toURL(),
    new File("META-INF/embeddedObjectGrid.xml").toURI().toURL());
Container container = server.createContainer(policy);
```

Processo do Servidor Autocontido

É possível iniciar todos os serviços juntos, o que é útil para o desenvolvimento e também prático na produção. Ao iniciar os serviços juntos, um único processo faz todo o seguinte: inicia o serviço de catálogo, inicia um conjunto de contêineres e executa a lógica de conexão do cliente. A inicialização dos serviços desta forma classifica os problemas de programação antes da implementação em um ambiente distribuído. O código a seguir mostra como instanciar um servidor eXtreme Scale autocontido:

```
CatalogServerProperties catalogServerProperties =
 ServerFactory.getCatalogProperties();
catalogServerProperties.setCatalogServer(true);

Server server = ServerFactory.getInstance();
DeploymentPolicy policy = DeploymentPolicyFactory.createDeploymentPolicy(
    new
File("META-INF/embeddedDeploymentPolicy.xml").toURI().toURL(),
    new File("META-INF/embeddedObjectGrid.xml").toURI().toURL());
Container container = server.createContainer(policy);
```

Incorporando o eXtreme Scale no WebSphere Application Server

A configuração para o eXtreme Scale é definida automaticamente quando você instalar o WebSphere Extended Deployment DataGrid em um ambiente WebSphere Application Server. Você não precisa configurar quaisquer propriedades antes de acessar o servidor para criar um contêiner. O código a seguir mostra como instanciar um servidor do eXtreme Scale no WebSphere Application Server:

```
Server server = ServerFactory.getInstance();
DeploymentPolicy policy = DeploymentPolicyFactory.createDeploymentPolicy(
    new
File("META-INF/embeddedDeploymentPolicy.xml").toURI().toURL(),
    new File("META-INF/embeddedObjectGrid.xml").toURI().toURL);
Container container = server.createContainer(policy);
```

Para obter um exemplo passo a passo de como iniciar um serviço de catálogo integrado e contêiner programaticamente, consulte “Utilizando a API do Servidor Integrado” na página 323.

Iniciando Servidores WebSphere eXtreme Scale Independentes

Ao executar uma configuração do WebSphere eXtreme Scale independente, o ambiente será composto de servidores de catálogo, servidores de contêineres e processos do cliente do eXtreme Scale. Os servidores eXtreme Scale também podem ser integrados a aplicativos Java existentes usando a API do servidor integrado. É necessário configurar e iniciar estes processos manualmente.

Antes de Iniciar

É possível iniciar os servidores do WebSphere eXtreme Scale em um ambiente que não possua o WebSphere Application Server instalado. Se você estiver utilizando o WebSphere Application Server, consulte “Administrando o WebSphere eXtreme Scale com o WebSphere Application Server” na página 342.

O que Fazer Depois

Pare seus processos do eXtreme Scale. Consulte "Parando Servidores eXtreme Scale Independentes" na página 337 para obter mais informações.

Iniciando o Serviço de Catálogo em um Ambiente Independente

Você deve iniciar o serviço de catálogo manualmente quando estiver utilizando um ambiente do WebSphere eXtreme Scale distribuído que não esteja em execução no WebSphere Application Server.

Antes de Iniciar

Se você estiver usando o WebSphere Application Server, o serviço de catálogo inicia automaticamente dentro de um dos processos existentes. Consulte Iniciando o Serviço de Catálogo no WebSphere Application Server para obter mais informações.

Sobre Esta Tarefa

O serviço de catálogo pode ser executado em um único processo ou pode incluir vários servidores de catálogos para formar o domínio do serviço de catálogo. Uma grade do servidor de catálogo é necessária em um ambiente de produção para alta disponibilidade. Para obter mais informações sobre como configurar um domínio do serviço de catálogo, consulte as informações sobre os domínios do serviço de catálogo no *Visão Geral do Produto*. O serviço de catálogo, esteja ele posicionado em uma grade ou em um processo único, é iniciado usando o script `startOgServer`. Também é possível especificar parâmetros adicionais para o script vincular o Object Request Broker (ORB) a um host e porta específicos, especificar o domínio ou ativar a segurança.

Ao chamar o comando iniciar, use o script `startOgServer.sh` em plataformas Unix ou `startOgServer.bat` no Windows.

Procedimento

- **Iniciando um único processo de servidor de catálogos**

  Para iniciar um servidor de catálogo único, digite os seguintes comandos a partir da linha de comandos:

  1. Navegue até o diretório bin:
     ```
     cd objectgridRoot/bin
     ```
  2. Execute o comando startOgServer:
     ```
     startOgServer.bat|sh catalogServer
     ```

  Para obter uma lista de todos os parâmetros de linha de comandos disponíveis, consulte "Script startOgServer" na página 334. Não utilize uma única Java Virtual Machine (JVM) para executar o serviço de catálogo em um ambiente de produção. Se o serviço de catálogo falhar, nenhum novo cliente poderá rotear para o eXtreme Scale implementado, e nenhuma nova instância ObjectGrid poderá ser incluída no domínio. Por esses motivos, você deve iniciar um conjunto de Java Virtual Machines para executar um domínio do serviço de catálogo.

- **Iniciando um domínio do serviço de catálogo multi-processos**

  Para iniciar um conjunto de servidores para executar um serviço de catálogo, é necessário utilizar a opção **-catalogServiceEndPoints** no script `startOgServer`.

Este argumento aceita uma lista de terminais de serviço de catálogo no formato de `serverName:hostName:clientPort:peerPort`. Cada atributo é definido conforme a seguir:

serverName
Especifica o nome para identificar o processo que você está ativando.

hostName
Especifica o nome do host para o computador onde o servidor é ativado.

clientPort
Especifica a porta que é usada para a comunicação da grade do catálogo peer.

peerPort
É o mesmo que haManagerPort. Especifica a porta que é usada para a comunicação da grade do catálogo peer.

O exemplo a seguir mostra como iniciar a primeira das três Java Virtual Machines para hospedar um serviço de catálogo:

1. Navegue até o diretório bin:
   ```
   cd objectgridRoot/bin
   ```
2. Execute o comando startOgServer:
   ```
   startOgServer.bat|sh cs1 -catalogServiceEndpoints
   cs1:MyServer1.company.com:6601:6602,
   cs2:MyServer2.company.com:6601:6602,
   cs3:MyServer3.company.com:6601:6602
   ```

   Neste exemplo, o servidor `cs1` no host `MyServer1.company.com` é iniciado. Este nome do servidor é o primeiro argumento que é passado para o script. Durante a inicialização do servidor `cs1`, os parâmetros catalogServiceEndpoints são examinados para determinar quais portas estão alocadas para este processo. A lista também é utilizada para permitir que o servidor `cs1` aceite conexões de outros servidores: `cs2` e `cs3`.
3. Para iniciar os servidores de catálogos restantes na lista, transmita os argumentos a seguir para o script startOgServer. Iniciando o servidor `cs2` no host `MyServer2.company.com`:
   ```
   startOgServer.bat|sh cs2 -catalogServiceEndpoints
   cs1:MyServer1.company.com:6601:6602,
   cs2:MyServer2.company.com:6601:6602,
   cs3:MyServer3.company.com:6601:6602
   ```

   Iniciando `cs3` no `MyServer3.company.com`:
   ```
   startOgServer.bat|sh cs3 -catalogServiceEndpoints
   cs1:MyServer1.company.com:6601:6602,
   cs2:MyServer2.company.com:6601:6602,
   cs3:MyServer3.company.com:6601:6602
   ```

   Importante: Inicie todos os servidores de catálogo em paralelo.

   Você deve iniciar os servidores de catálogos que estão em uma grade em paralelo, porque cada servidor faz uma pausa para aguardar até que os outros servidores de catálogo juntem-se ao grupo principal. Um servidor de catálogos que está configurado para uma grade não inicia até identificar outros membros no grupo. O servidor de catálogos eventualmente expira se outros servidores tornam-se disponíveis.

- **Ligando o ORB a um host e porta específicos**

Não considerando as portas definidas no argumento **catalogServiceEndpoints**, cada serviço de catálogo também utiliza um Object Request Broker (ORB) para aceitar conexões de clientes e contêineres. Por padrão, o ORB atende na porta `2809` do host local. Se desejar vincular o ORB a um host específico em um JVM de serviço de catálogo, use os argumentos **-listenerHost** e **-listenerPort**. O exemplo a seguir mostra como iniciar um único servidor de catálogos JVM com seu ORB ligado à porta 7000 no MyServer1.company.com:

```
startOgServer.sh catalogServer -listenerHost MyServer1.company.com
-listenerPort 7000
```

Cada contêiner e cliente do eXtreme Scale deve ser fornecidos com dados do terminal ORB do serviço de catálogo. Os clientes precisam apenas de um subconjunto destes dados, mas você deve utilizar pelo menos dois terminais para alta disponibilidade.

- **Nomeando o domínio**

  Um nome de domínio não é requerido quando iniciar um serviço de catálogo. O nome de domínio padrão é `defaultDomain`. Para dar um nome ao seu domínio, use a opção **-domain**. O exemplo a seguir demonstra como iniciar uma única JVM de serviço de catálogo com o nome de domínio`myDomain`.

  ```
  startOgServer.sh catalogServer -domain myDomain
  ```

- **Iniciar um serviço de catálogo seguro**

  É possível iniciar um serviço de catálogo seguro fornecendo os seguintes argumentos:

  -clusterSecurityFile e -clusterSecurityUrl
  : Estes argumentos especificam o arquivo `objectGridSecurity.xml`, que descreve as propriedades de segurança que são comuns a todos os servidores (incluindo servidores de catálogos e servidores de contêineres). Um dos exemplos de propriedade é a configuração do autenticador que representa o registro do usuário e o mecanismo de autenticação.

  -serverProps
  : Especifica o arquivo de propriedades do servidor, que contém as propriedades de segurança específicas do servidor. O nome do arquivo especificado para esta propriedade está em formato de caminho de arquivo simples, tal como `c:/tmp/og/catalogserver.props`.

  Para obter um exemplo de como iniciar um serviço de catálogo seguro, consulte a etapa 2 do do Tutorial de Segurança Java SE no *Visão Geral do Produto*.. Para obter um exemplo do arquivo `objectGridSecurity.xml`, consulte "Arquivo XML Descritor de Segurança" na página 373.

Iniciando Processos do Contêiner

É possível iniciar o eXtreme Scale a partir da linha de comandos utilizando uma topologia de implementação ou utilizando um arquivo`server.properties`.

Sobre Esta Tarefa

Para iniciar um processo de contêiner, é necessário um arquivo XML ObjectGrid. O arquivo XML ObjectGrid especifica quais servidores eXtreme Scale o contêiner hospeda. Assegure-se de que o contêiner esteja equipado para hospedar cada ObjectGrid no XML transmitido para ele. Todas as classes que estes ObjectGrids requerem devem estar no caminho de classe para o contêiner. Para obter mais

informações sobre o arquivo XML do ObjectGrid, consulte "Arquivo objectGrid.xsd" na página 158.

Procedimento

- **Inicie o processo do contêiner a partir da linha de comandos.**
  1. A partir da linha de comandos, navegue até o diretório bin:
     ```
     cd objectgridRoot/bin
     ```
  2. Execute o seguinte comando:
     ```
     startOgServer.sh c0 -objectGridFile ../xml/companyGrid.xml
     -catalogServiceEndPoints MyServer1.company.com:2809
     ```

     Importante: No contêiner, a opção **-catalogServiceEndPoints** é utilizada para fazer referência ao host do Object Request Broker (ORB) e porta no serviço de catálogo. O serviço de catálogo utiliza as opções**-listenerHost** e **-listenerPort** para especificar o host e a porta para ligação ORB ou aceita a ligação padrão. Quando estiver iniciando um contêiner, use a opção**-catalogServiceEndPoints** para referenciar os valores que são transmitidos para as opções **-listenerHost** e **-listenerPort** no serviço de catálogo. Se as opções **-listenerHost** e **-listenerPort** não forem utilizadas quando o serviço de catálogo é iniciado, o ORB se conecta à porta 2809 no host local para o serviço de catálogo. Não use a opção **-catalogServiceEndPoints** para referenciar os hosts e portas que foram transmitidos para a opção **-catalogServiceEndPoints** no serviço de catálogo. No serviço de catálogo, a opção **-catalogServiceEndPoints** é usada para especificar portas necessárias para a configuração de servidor estático.
     Esse processo é identificado por c0, o primeiro argumento transmitido para o script. Utilize `companyGrid.xml` para iniciar o contêiner. Se o seu ORB do servidor de catálogos estiver em execução em um host diferente do seu contêiner ou estiver usando uma porta não-padrão, deverá usar o argumento **-catalogServiceEndPoints** para se conectar com o ORB. Para este exemplo, assuma que um único serviço de catálogo está em execução na porta `2809` no `MyServer1.company.com`
- **Inicie o contêiner usando uma política de implementação.**

  Embora não seja necessário, uma política de implementação é recomendada durante a inicialização do contêiner. A política de implementação é utilizada para configurar o particionamento e a replicação para o eXtreme Scale. A política de implementação também pode ser utilizada para influenciar o comportamento de disposição. Como o exemplo anterior não forneceu um arquivo de política de implementação, o exemplo recebe todos os valores-padrão com relação a replicação, particionamento e disposição. Portanto, os mapas no CompanyGrid estão em um mapSet. O mapSet não é particionado ou replicado. Para obter mais informações sobre os arquivos de política de implementação, consulte "Arquivo Descritor XML de Política de Implementação" na página 173. O exemplo a seguir utiliza o arquivo `companyGridDpReplication.xml` para iniciar uma JVM do contêiner, a JVM c0:
  1. A partir da linha de comandos, navegue até o diretório bin:
     ```
     cd objectgridRoot/bin
     ```
  2. Execute o seguinte comando:
     ```
     startOgServer.sh c0 -objectGridFile ../xml/companyGrid.xml
     -deploymentPolicyFile ../xml/companyGridDpReplication.xml
     -catalogServiceEndPoints MyServer1.company.com:2809
     ```

     Nota: Se você tiver classes Java armazenadas em um diretório específico, em vez de alterar o script StartOgServer, é possível ativar o servidor com argumentos semelhantes aos seguintes: `-jvmArgs -cp C:\ . . . \DirectoryPOJOs\POJOs.jar`

. No arquivo companyGridDpReplication.xml, um único conjunto de mapas contém todos os mapas. Esse mapSet é dividido em 10 partições. Cada partição possui uma réplica síncrona e nenhuma réplica assíncrona. Qualquer contêiner que utiliza a política de implementação companyGridDpReplication.xml em par com o arquivo XML do ObjectGrid companyGrid.xml também está apto a hospedar shards do CompanyGrid. Inicie outra JVM do contêiner, a JVM c1:

1. A partir da linha de comandos, navegue até o diretório bin:

   ```
   cd objectgridRoot/bin
   ```

2. Execute o seguinte comando:

   ```
   startOgServer.sh c1 -objectGridFile ../xml/companyGrid.xml
   -deploymentPolicyFile ../xml/companyGridDpReplication.xml
   -catalogServiceEndPoints MyServer1.company.com:2809
   ```

Cada política de implementação contém um ou mais elementos objectgridDeployment. Quando um contêiner é iniciado, ele publica sua política de implementação no serviço de catálogo. O serviço de catálogo examina cada elemento objectgridDeployment. Se o atributo objectgridName corresponde ao atributo objectgridName de um elemento objectgridDeployment anteriormente recebido, o elemento objectgridDeployment mais recente é ignorado. O primeiro elemento objectgridDeployment recebido para um atributo objectgridName especifico é usado como o elemento principal. Por exemplo, assuma que a JVM c2 utiliza uma política de implementação que divide o mapSet em um número diferente de partições:

companyGridDpReplicationModified.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<deploymentPolicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://ibm.com/ws/objectgrid/deploymentPolicy
  ../deploymentPolicy.xsd"
    xmlns="http://ibm.com/ws/objectgrid/deploymentPolicy">

    <objectgridDeployment objectgridName="CompanyGrid">
        <mapSet name="mapSet1" numberOfPartitions="5"
            minSyncReplicas="1" maxSyncReplicas="1"
            maxAsyncReplicas="0">
            <map ref="Customer" />
            <map ref="Item" />
            <map ref="OrderLine" />
            <map ref="Order" />
        </mapSet>
    </objectgridDeployment>

</deploymentPolicy>
```

Agora, é possível iniciar uma terceira JVM, a JVM c2:

1. A partir da linha de comandos, navegue até o diretório bin:

   ```
   cd objectgridRoot/bin
   ```

2. Execute o seguinte comando:

   ```
   startOgServer.sh c2 -objectGridFile ../xml/companyGrid.xml
   -deploymentPolicyFile ../xml/companyGridDpReplicationModified.xml
   -catalogServiceEndPoints MyServer1.company.com:2809
   ```

O contêiner na JVM c2 é iniciado com uma política de implementação que especifica 5 partições para mapSet1. No entanto, o serviço de catálogo já mantém a cópia principal do objectgridDeployment para o CompanyGrid. Quando a JVM c0 foi iniciada, ela especificou que 10 partições existem para esse mapSet. Como ela foi o primeiro contêiner a iniciar e publicar a política de implementação, sua política de implementação se tornou o principal. Portanto, qualquer valor de

atributo objectgridDeployment que seja igual ao `CompanyGrid` em uma política de implementação subsequente será ignorado.

- **Inicie um contêiner utilizando um arquivo de propriedades do servidor.**

  É possível usar um arquivo de propriedades de servidor para configurar o rastreio e configurar a segurança em um contêiner. Execute os seguintes comandos para iniciar o contêiner c3 com um arquivo de propriedades do servidor:

  1. A partir da linha de comandos, navegue até o diretório bin:

     ```
     cd objectgridRoot/bin
     ```

  2. Execute o seguinte comando:

     ```
     startOgServer.sh c3 -objectGridFile ../xml/companyGrid.xml
     -deploymentPolicyFile ../xml/companyGridDpReplicationModified.xml
     -catalogServiceEndPoints MyServer1.company.com:2809
     -serverProps ../serverProps/server.properties
     ```

  A seguir há um exemplo do arquivo `server.properties`:

  ```
  server.properties
  workingDirectory=
  traceSpec=*=all=disabled
  systemStreamToFileEnabled=true
  enableMBeans=true
  memoryThresholdPercentage=50
  ```

  Este é um arquivo de propriedades de servidor básico que não possui a segurança ativada. Para obter mais informações sobre o arquivo `server.properties`, consulte “Arquivo de Propriedades do Servidor” na página 183.

Script startOgServer

O script startOgServer inicia servidores. É possível utilizar uma variedade de parâmetros quando você inicia seu servidores para ativar o rastreio, especificar números de porta e assim por diante.

Finalidade

É possível utilizar o script startOgServer para iniciar servidores.

Local

O script startOgServer está no diretório bin do diretório-raiz, por exemplo:

```
cd objectgridRoot/bin
```

Nota: Se você tiver classes Java armazenadas em um diretório específico, em vez de alterar o script startOgServer, é possível ativar o servidor com argumentos como os seguintes: `-jvmArgs -cp C:\ . . . \DirectoryPOJOs\POJOs.jar`

.

Uso para Servidores de Catálogos

Para iniciar um servidor de catálogos:

Windows

```
startOgServer.bat <server> [options]
```

UNIX

```
startOgServer.sh <server>[options]
```

Para iniciar um servidor de catálogos configurado padrão, utilize os seguintes comandos:

Windows

```
startOgServer.bat catalogServer
```

UNIX

```
startOgServer.sh catalogServer
```

Opções para Iniciar Servidores de Catálogos

Parâmetros para iniciar um servidor de catálogos:

-catalogServiceEndPoints <server:serverHost:clientPort:peerPort,server:serverHost:clientPort:peerPort>
No contêiner, a opção **-catalogServiceEndpoints** é utilizada para fazer referência ao host do ORB (Object Request Broker) e porta no serviço de catálogo.

-clusterSecurityFile <arquivo xml de segurança do cluster>
Especifica o local do arquivo XML do descritor de segurança.

-clusterSecurityUrl <URL do xml de segurança do cluster>

-domain <nome de domínio>

-listenerHost <nome do host>
Padrão: `localhost`

Especifica o host do listener para comunicação com Internet Inter-ORB Protocol (IIOP).

-listenerPort <porta>
Padrão: `2809`

Especifica a porta listener para comunicação com o IIOP.

-haManagerPort <porta>
Padrão: É o mesmo que porta de peer. Se essa propriedade não estiver configurada, o serviço de catálogo gerará automaticamente uma porta disponível.

Especifica o número da porta usada pelo gerenciador de alta disponibilidade.

-serverProps <arquivo de propriedades do servidor>
Especifica o arquivo de propriedades do servidor que contém as propriedades de segurança específicas do servidor. O nome do arquivo especificado para esta propriedade está simplesmente no formato de caminho de arquivo simples, tal como `c:/tmp/og/catalogserver.props`.

-JMXServicePort <porta>
Padrão: `1099`

Especifica o número da porta para comunicação com Java Management Extensions (JMX).

-traceSpec <especificação de rastreio>
Especifica uma cadeia que especifica o escopo do rastreio que é ativado quando o servidor inicia.

Exemplo:

- `ObjectGrid=all=enabled`
- `ObjectGrid*=all=enabled`

-traceFile <arquivo de rastreio>
Especifica o caminho de um arquivo no qual salvar informações de rastreio.

Exemplo: `../logs/c4Trace.log`

-timeout <segundos>
Especifica um número de segundos antes do servidor expirar.

-script <arquivo de script>
Cria um script customizado para os comandos que você especifica para iniciar servidores de catálogo ou contêineres e depois parametriza ou edita conforme sua necessidade.

-jvmArgs <argumentos JVM>
Especifica um conjunto de argumentos JVM. Cada parâmetro após o parâmetro **-jvmArgs** é utilizado para iniciar a Java virtual machine (JVM) do servidor. Quando o parâmetro **-jvmArgs** é utilizado, certifique-se de que este seja o último argumento de script opcional especificado.

Exemplo: `-jvmArgs -Xms256M -Xmx1G`

Uso dos Servidores de Contêiner Windows

```
startOgServer.bat <server> -objectgridFile <xml file>
-deploymentPolicyFile <arquivo xml> [options]
```

Windows

```
startOgServer.bat <servidor> -objectgridUrl <URl xml> -deploymentPolicyUrl
<URL
xml> [options]
```

UNIX

```
startOgServer.sh <server> -objectgridFile <xml file>
-deploymentPolicyFile <arquivo xml> [options]
```

UNIX

```
startOgServer.sh <servidor> -objectgridUrl <URL xml> -deploymentPolicyUrl <URL xml> [options]
```

Opções para Servidores de Contêiner

-catalogServiceEndPoints<hostName:port,hostName:port>
Padrão: localhost:2809

-deploymentPolicyFile <arquivo xml da política de implementação>
Especifica o caminho para o arquivo de políticas de implementação.

Exemplo: `../xml/SimpleDP.xml`

-deploymentPolicyUrl <deployment policy url>

-objectgridFile <ObjectGrid descriptor xml file>
Especifica o caminho para o arquivo descritor ObjectGrid.

-objectgridUrl <ObjectGrid descriptor url>

-listenerHost <nome do host>
Padrão: `localhost`

Especifica o host do listener para comunicação com Internet Inter-ORB Protocol (IIOP).

-listenerPort <porta>
Padrão: 2809

Especifica a porta listener para comunicação com o IIOP.

-serverProps <arquivo de propriedades do servidor>
Especifica o caminho para o arquivo de propriedades do servidor.

Exemplo:../security/server.props

-zone <nome da zona>
Especifica a zona a utilizar para todos os contêineres no servidor.

-traceSpec <especificação de rastreio>
Especifica uma cadeia que especifica o escopo do rastreio que é ativado quando o servidor inicia.

Exemplo:

- ObjectGrid=all=enabled
- ObjectGrid*=all=enabled

-traceFile <arquivo de rastreio>
Especifica o caminho de um arquivo no qual salvar informações de rastreio.

Exemplo: ../logs/c4Trace.log

-timeout <segundos>
Especifica um número de segundos antes do servidor expirar.

-script <arquivo de script>
Cria um script customizado para os comandos que você especifica para iniciar servidores de catálogo ou contêineres e depois parametriza ou edita conforme sua necessidade.

-jvmArgs <argumentos JVM>
Especifica um conjunto de argumentos JVM. Cada parâmetro após o parâmetro **-jvmArgs** é utilizado para iniciar a Java virtual machine (JVM) do servidor. Quando o parâmetro **-jvmArgs** é utilizado, certifique-se de que este seja o último argumento de script opcional especificado.

Exemplo:**-jvmArgs** -Xms256M -Xmx1G

Parando Servidores eXtreme Scale Independentes

É possível utilizar o script stopOgServer para parar processos do servidor.

Procedimento

- Parar processos do eXtreme Scale.
  1. A partir da linha de comandos, navegue até o diretório bin.

     cd *objectGridRoot*/bin
  2. Execute o script stopOgServer para parar o servidor. O seguinte exemplo para o servidor c0:

     stopOgServer *c0*
     -catalogServiceEndPoints *MyServer1.company.com:2809*

     Utilize o mesmo script para parar servidores múltiplos com uma lista separada por vírgulas da seguinte forma:

     stopOgServer c0,c1,c2 -catalogServiceEndPoints
     MyServer1.company.com:2809

Atenção: A opção **-catalogServiceEndPoints** deve corresponder à opção **-catalogServiceEndPoints** usada para iniciar o contêiner. Se um **-catalogServiceEndPoints** não tiver sido usado para iniciar o contêiner, os valores padrão serão, provavelmente, localhost ou hostname e 2809 para a porta ORB para conectar ao serviço de catálogo. Caso contrário, use os valores transmitidos para **-listenerHost** e **-listenerPort** no serviço de catálogo. Se as opções **-listenerHost** e **-listenerPort** não são utilizadas ao iniciar o serviço de catálogo, o ORB conecta-se à porta 2809 no host local para o serviço de catálogo.

- Parar processos do serviço de catálogo do eXtreme Scale.
  1. A partir da linha de comandos, navegue até o diretório bin.
     ```
     cd objectGridRoot/bin
     ```
  2. Execute o script stopOgServer para parar o servidor.
     ```
     stopOgServer.sh catalogServer -catalogServiceEndPoints MyServer1.company.com:2809
     ```

  Atenção: Ao parar um catalogService, a opção **-catalogServiceEndPoints** é usada para fazer referência ao host e à porta do Object Request Broker (ORB) no serviço de catálogo. O serviço de catálogo utiliza as opções **-listenerHost** e **-listenerPort** para especificar o host e a porta para a ligação ORB ou aceita a ligação padrão. Se as opções**-listenerHost** e **-listenerPort** não são utilizadas ao iniciar o serviço de catálogo, o ORB conecta-se à porta 2809 no host local para o serviço de catálogo. A opção **-catalogServiceEndPoints** será diferente ao parar um serviço de catálogo e ao iniciá-lo.

  Iniciar um serviço de catálogo requer portas de acesso peer e portas de acesso do cliente, se as portas padrão não tiverem sido usadas. Parar um serviço de catálogo requer apenas a porta ORB.
- Ativar o rastreio para o processo de interrupção do servidor. Se um contêiner falhar ao parar, é possível ativar o rastreio para ajudar com a depuração do problema. Para ativar o rastreio durante a interrupção de um servidor, inclua os parâmetros **-traceSpec** e **-traceFile** nos comandos de interrupção. O parâmetro **-traceSpec** especifica o tipo de rastreio a ser ativado e o parâmetro **-traceFile** especifica o caminho e o nome do arquivo a ser criado e utilizado para os dados de rastreio.
  1. A partir da linha de comandos, navegue até o diretório bin.
     ```
     cd objectGridRoot/bin
     ```
  2. Execute o script stopOgServer com o rastreio ativado.
     ```
     stopOgServer.sh c4 -catalogServiceEndPoints MyServer1.company.com:2809
      -traceFile ../logs/c4Trace.log -traceSpec ObjectGrid=all=enabled
     ```

     Após o rastreio ser obtido, consulte os erros relacionados aos conflitos de porta, às classes ausente, aos arquivos XML ausentes ou incorretos ou a quaisquer rastreios de pilha. As especificações de rastreio de inicialização sugeridas são:

     – `ObjectGrid=all=enabled`

     – `ObjectGrid*=all=enabled`

     Para todas as opções de especificação de rastreio, consulte “Opções de Rastreio” na página 447.

Script stopOgServer

O script stopOgServer para servidores.

Finalidade

Ao fornecer seu nome e catalogServiceEndpoints, é possível utilizar o script stopOgServer para parar um servidor.

Local

O script stopOgServer está no diretório bin do diretório-raiz, por exemplo:

```
cd objectgridRoot/bin
```

USO NÃO-RAIZ

Para parar um servidor de catálogos:

Windows

```
stopOgServer.bat <server> -catalogServiceEndPoints
<csHost:csListenerPort,csHost:csListenerPort> [opções]
```

UNIX

```
stopOgServer.sh <server> -catalogServiceEndPoints
<csHost:csListenerPort,csHost:csListenerPort> [opções]
```

Para parar um servidor de contêineres do ObjectGrid:

Windows

```
stopOgServer.bat <server> -catalogServiceEndPoints
<csHost:csListenerPort,csHost:csListenerPort> [opções]
```

UNIX

```
startOgServer.sh -catalogServiceEndPoints
<csHost:csListenerPort,csHost:csListenerPort> [opções]
```

Opções

-clientSecurityFile <arquivo de propriedades de segurança>

-traceSpec <especificação de rastreio>
Especifica uma cadeia que especifica o escopo do rastreio que é ativado quando o servidor inicia.

Exemplo:

- `ObjectGrid=all=enabled`
- `ObjectGrid*=all=enabled`

-traceFile <arquivo de rastreio>
Especifica o caminho de um arquivo no qual salvar informações de rastreio.

Exemplo: `../logs/c4Trace.log`

-jvmArgs <argumentos JVM>
Todos os parâmetros após a opção -jvmArgs são utilizados para iniciar a Java virtual machine (JVM) do servidor. Quando a opção -jvmArgs é utilizada, assegure-se de que seja o último argumento de script opcional especificado.

Início e Encerramento de Servidores Seguros do eXtreme Scale

Muitas vezes os servidores precisam ser seguros para seu ambiente de implementação, o que requer configuração específica para iniciar e parar.

Inicialização de um Servidor Seguro em um Ambiente Java SE

É possível iniciar um serviço de catálogo ou servidores de contêineres da forma a seguir.

Iniciando um serviço de catálogo seguro do eXtreme Scale

A inicialização de um processo do serviço de catálogo seguro do eXtreme Scale exige mais dois arquivos de configuração de segurança:

Arquivo XML descritor de segurança: O arquivo XML descritor de segurança descreve as propriedades de segurança comuns para todos os servidores (incluindo servidores de catálogo e servidores de contêiner). Um exemplo de propriedade é a configuração do autenticador que representa o registro do usuário e o mecanismo de autenticação.

Arquivo de propriedades do servidor. O arquivo de propriedades do servidor configura as propriedades específicas de segurança para o servidor.

Quando você usa o comando startOgServer.sh ou startOgServer.cat para iniciar um processo seguro do serviço de catálogo do eXtreme Scale, é possível usar o -clusterSecurityFile ou -clusterSecurityUrl para configurar seguramente o arquivo XML do descritor de segurança como um tipo file ou URL, e pode usar –serverProps para configurar o arquivo de propriedades do servidor.

Início de um servidor de contêineres seguros do eXtreme Scale

A inicialização de um servidor de contêineres seguro do eXtreme Scale exige um arquivo de configuração de segurança:

- **Arquivo de propriedades do servidor:** O arquivo de propriedades do servidor configura as propriedades de segurança específicas para o servidor. Consulte o “Arquivo de Propriedades do Servidor” na página 183 para obter mais detalhes.

Quando você usa o comando startOgServer.sh ou startOgServer.cat para iniciar um servidor de contêineres seguro do eXtreme Scale, é possível usar –serverProps para configurar o arquivo de propriedades do servidor. Há mais formas para configurar o arquivo de propriedades de servidor, consulte o arquivo de propriedades do servidor para obter mais detalhes.

Para obter mais detalhes sobre como usar o comando startOgServer.sh ou startOgServer.bat e suas opções, consulte “Script startOgServer” na página 334.

Encerramento de um Servidor Seguro do eXtreme Scale

O encerramento de um processo seguro do serviço de catálogo do eXtreme Scale ou servidor de contêineres exige um arquivo de configuração de segurança:

- **Arquivo de propriedades do cliente:** O arquivo de propriedades do cliente pode ser usado para configurar as propriedades de segurança do cliente. As

propriedades de segurança do cliente são necessárias para um cliente se conectar a um servidor seguro. Consulte o “Arquivo de Propriedades do Cliente” na página 202 para obter mais detalhes.

Quando você usa o comando stopOgServer.sh ou stopOgServer.cat para encerrar um processo seguro do serviço de catálogo do eXtreme Scale ou servidor de contêineres, é possível usar -clientSecurityFile para configurar as propriedades de segurança do cliente.

Para obter mais detalhes sobre como usar o comando stopOgServer.sh ou stopOgServer.cat e suas opções, consulte “Script stopOgServer” na página 338.

Iniciando um Servidor Seguro no WebSphere Application Server

A inicialização de um servidor seguro ObjectGrid em WebSphere Application Server é semelhante à inicialização de um servidor não seguro ObjectGrids, exceto que é necessário passar os arquivos de configuração de segurança. Em vez de usar o -[PROPERTY_FILE] (por exemplo -serverProps) no comando como no ambiente Java SE, use -D[PROPERTY_FILE] nos argumentos genéricos da Java Virtual Machine (JVM).

Iniciando um serviço de catálogo seguro no Websphere Application Server

Um servidor de catálogos contém dois níveis diferente de informações de segurança:

- `-Dobjectgrid.cluster.security.xml.url`: Especifica o arquivo objectGridSecurity.xml que descreve as propriedades de segurança comuns a todos os servidores (incluindo servidores de catálogos e servidores de contêineres). Um exemplo é a configuração do autenticador que representa o mecanismo de registro e autenticação do usuário. O nome do arquivo especificado para esta propriedade deve estar em um formato de URL, como "file:///tmp/og/objectGridSecurity.xml".
- `-Dobjectgrid.server.props`: Especifica o arquivo de propriedades do servidor que contém as propriedades de segurança específicas do servidor. O nome do arquivo especificado para esta propriedade está apenas no formato simples de caminho de arquivo, como "c:/tmp/og/catalogserver.props". Observe que o uso de -Dobjectgrid.security.server.props está reprovado, mas ele pode continuar sendo usado para compatibilidade com versões anteriores.

Para iniciar um serviço de catálogo seguro em WebSphere Application Server, siga o "Incorporado no WebSphere Application Server" no “Segurança de Grade” na página 356.

Em seguida, a propriedade de segurança no argumento da JVM genérico do processo.

```
-Dobjectgrid.cluster.security.xml.url=file:///tmp/og/
objectGridSecurity.xml-Dobjectgrid.server.props=/tmp/og/
catalog.server.props
```

As etapas para incluir os argumentos da JVM genéricos são as seguintes:

- Expanda "Administração do sistema" na visualização de tarefa do lado esquerdo.
- Clique no processo WebSphere Application Server no qual o serviço de catálogo está implementado como, por exemplo, "Gerenciador de Implementação".

- Na página à direita, expanda "Java e Gerenciamento de Processo" sob "Infraestrutura do Servidor".
- Clique em "Definição de Processo".
- Clique em "Java Virtual Machine" sob "Propriedades Adicionais".
- Digite as propriedades na caixa de texto Argumentos JVM Genéricos.

Iniciando um servidor de contêineres seguro no WebSphere Application Server

Um servidor de contêineres, ao se conectar ao servidor de catálogos, obterá todas as configurações de segurança definidas no objectGridSecurity.xml, como a configuração do autenticador ou do tempo limite da sessão de login. Além disso, um servidor de contêineres precisa configurar suas próprias propriedades de segurança específicas do servidor na propriedade -Dobjectgrid.server.props.

É necessário usar a propriedade -Dobjectgrid.server.props em vez da propriedade -Dobjectgrid.security.server.props porque também colocamos outras propriedades não relacionadas à segurança neste arquivo de propriedades. O nome do arquivo especificado para esta propriedade está apenas no formato de caminho de arquivo simples, como `c:/tmp/og/server.props`.

Siga as mesmas etapas acima para incluir a propriedade de segurança para os argumentos genéricos da JVM.

Administrando o WebSphere eXtreme Scale com o WebSphere Application Server

É possível executar um serviço de catálogo e processos de servidor de contêiner no WebSphere Application Server. O processo para configurar esses servidores é diferente de uma configuração independente. O serviço de catálogo pode ser iniciado automaticamente nos servidores ou gerenciadores de implementação do WebSphere Application Server. O processo do contêiner é iniciado quando um aplicativo eXtreme Scale for implementado e iniciado no ambiente do WebSphere Application Server.

Iniciando o Processo do Serviço de Catálogo em um Ambiente WebSphere Application Server

Os servidores de catálogos do WebSphere eXtreme Scale podem ser iniciados automaticamente em um ambiente do WebSphere Application Server ou WebSphere Application Server Network Deployment. É possível configurar o serviço de catálogo para execução em qualquer processo na célula do WebSphere. Um serviço de catálogo não armazenado em cluster e de servidor único é aceitável para ambientes de desenvolvimento. Para um ambiente de produção, você deve usar um domínio do serviço de catálogo com vários servidores de catálogos.

Antes de Iniciar

- É necessário instalar o WebSphere Application Server e o WebSphere eXtreme Scale.

  Se estiver executando uma instalação gráfica de eXtreme Scale, você terá a opção de aumentar os perfis existentes, incluindo o perfil de gerenciador de implementação. Você também pode aumentar perfis após a instalação usando Profile Management Tool (PMT), a versão da GUI ou a partir da linha de comandos (manageprofiles.sh|bat). Os perfis criados após o produto ser instalado podem ser aumentados como parte do processo de criação de perfil ou

posteriormente usando a PMT. Não há GUI da Profile Management Tool para as instalações de 64 bits do WebSphere Application Server. Nesses casos, use o script manageprofiles a partir da linha de comandos.

Consulte Capítulo 3, “Instalando e Implementando WebSphere eXtreme Scale”, na página 17 para obter mais informações.

- 7.1+ Defina um domínio do serviço de catálogo. Consulte “Criando Domínios do Serviço de Catálogo no WebSphere Application Server” na página 344 para obter mais informações.

Sobre Esta Tarefa

O processo do serviço de catálogo pode ser executado em um processo do WebSphere Application Server. Onde e como o serviço de catálogo é executado depende da edição do WebSphere Application Server que você está usando:

WebSphere Application Server base
O serviço de catálogo é executado no processo do servidor de aplicativos. Por padrão, ele *não* é ativado para iniciar automaticamente.

WebSphere Application Server Network Deployment
O serviço de catálogo é executado automaticamente no processo do gerenciador de implementação, mas é possível configurá-lo para que seja executado em um ou mais processos do agente do nó ou do servidor de aplicativos.

Recurso Reprovado: Nos releases anteriores, você definiu a propriedade customizada catalog.services.cluster para definir um grupo de servidores de catálogos em seu ambiente do WebSphere Application Server. Se você tiver configurado esta propriedade customizada com um release anterior, as configurações ainda estarão em vigor. No entanto, se você estiver criando uma nova configuração, poderá atingir a mesma configuração criando um domínio do serviço de catálogo no console administrativo do WebSphere Application Server. Consulte “Criando Domínios do Serviço de Catálogo no WebSphere Application Server” na página 344 para obter mais informações.

Restrição: Não é possível ter servidores em um ambiente do WebSphere Application Server com o mesmo nome quando os servidores estiverem usando o Object Request Broker (ORB) para se comunicarem entre si. É possível resolver essa restrição especificando a propriedade de sistema **-Dcom.ibm.websphere.orb.uniqueServerName=true** para os processos que têm o mesmo nome. A seguir temos exemplos: Quando servidores com o nome server1 em cada nó são utilizados como uma grade de serviços de catálogo, ou quando vários agentes de nó são utilizados para formar uma grade de serviço de catálogo.

Procedimento

- **Iniciando um servidor de catálogos no WebSphere Application Server** base:

  Quando o WebSphere eXtreme Scale é integrado a um perfil do WebSphere Application Server não federado, o serviço de catálogo não é iniciado automaticamente, exceto nas duas circunstâncias a seguir:

  – Um aplicativo que é configurado para iniciar automaticamente um contêiner do eXtreme Scale: O serviço de catálogo e o contêiner serão iniciados automaticamente. Este recurso simplifica o teste de unidade em ambientes de desenvolvimento como o Rational Application Developer, pois você não precisa iniciar explicitamente um serviço de catálogo. Consulte “Iniciando

Contêineres do eXtreme Scale Automaticamente em Aplicativos WebSphere Application Server" na página 351 para obter mais informações.
  - Se um domínio do serviço de catálogo estiver definido.
- **Iniciando um serviço de catálogo no WebSphere Application Server Network Deployment**: O serviço de catálogo é iniciado em uma célula do WebSphere Application Server Network Deployment nos seguintes cenários:
  - Quando o WebSphere eXtreme Scale é instalado no WebSphere Application Server Network Deployment, o serviço de catálogo será automaticamente iniciado no processo do gerenciador de implementação se for aumentado para permitir um desenvolvimento rápido prontamente.
  - Quando você define um domínio do serviço de catálogo. O domínio do serviço de catálogo é uma coleta de servidores de catálogos definidos. Esses servidores de catálogos pode ser iniciados em servidores de aplicativos existentes dentro do ambiente do WebSphere Application Server Network Deployment ou é possível definir servidores remotos. Consulte "Criando Domínios do Serviço de Catálogo no WebSphere Application Server" para obter mais informações.

Atenção: Não coloque os serviços de catálogo com servidores de contêiner do eXtreme Scale em um ambiente de produção. Inclua o serviço de catálogo em vários processos do agente do nó ou em um servidor de aplicativos que não esteja hospedando um aplicativo eXtreme Scale.

Após definir um domínio do serviço de catálogo, você deve reiniciar cada servidor identificado. Quando você reinicia esses servidores, o domínio do serviço de catálogo é iniciado automaticamente.

Criando Domínios do Serviço de Catálogo no WebSphere Application Server

Os domínios do serviço de catálogo definem um grupo de servidores de catálogo que gerenciam o posicionamento de shards e monitoram o funcionamento de servidores de contêiner em sua grade de dados.

Antes de Iniciar

- Instale o WebSphere eXtreme Scale no WebSphere Application Server. Consulte "Integrando o WebSphere eXtreme Scale ou o WebSphere eXtreme Scale Client com o WebSphere Application Server" na página 22 para obter mais informações.

Sobre Esta Tarefa

Ao criar um domínio do serviço de catálogo, você está definindo uma coleta altamente disponível de servidores de catálogo.

Esses servidores de catálogo podem ser executados no WebSphere Application Server dentro de uma única célula ou um grupo principal. O domínio do serviço de catálogo também pode definir um grupo remoto de servidores executados em processos SE Java diferentes ou outras células do WebSphere Application Server. Quando você define um domínio do serviço de catálogo que posiciona servidores de catálogo nos servidores de aplicativos dentro da célula, são usados os mecanismos do grupo principal do WebSphere Application Server. Como resultado, os membros de um único domínio do serviço de catálogo não podem ampliar os limites de um grupo principal e um domínio do serviço de catálogo, portanto, não pode ampliar células. No entanto, o WebSphere eXtreme Scale pode ampliar células

conectando a um servidor de catálogo por meio de limites de células, como um domínio do serviço de catálogo independente ou um domínio do serviço de catálogo integrado em outra célula.

Atenção: Não coloque os serviços de catálogo com servidores de contêiner do WebSphere eXtreme Scale em um ambiente de produção. Inclua o serviço de catálogo em vários processos do agente do nó ou em um servidor de aplicativos que não esteja hospedando um aplicativo WebSphere eXtreme Scale.

Também será possível criar um domínio do serviço de catálogo se estiver executando no modo independente. Consulte "Iniciando o Serviço de Catálogo em um Ambiente Independente" na página 329 para obter mais informações.

Procedimento

1. Crie o domínio do serviço de catálogo.
   a. No console administrativo do WebSphere Application Server, clique em **Administração do sistema → WebSphere eXtreme Scale → Domínios do serviço de catálogo → Novo**.
   b. Defina um nome, valor padrão e credenciais de autenticação JMX para o domínio do serviço de catálogo.
   c. Inclua terminais do servidor de catálogos. É possível selecionar servidores de aplicativos existentes ou incluir servidores remotos que estão executando um serviço de catálogo.
2. Teste a conexão com o domínio do serviço de catálogo.
   a. No console administrativo do WebSphere Application Server, clique em **Administração do sistema → WebSphere eXtreme Scale → Domínios do serviço de catálogo**.
   b. Selecione o domínio do serviço de catálogo que deseja testar e clique em **Testar conexão**. Quando você clica neste botão, todos os terminais do domínio de serviço de catálogo definidos são consultados um a um, se algum terminal estiver disponível, retornará uma mensagem que indica que a conexão com o domínio do serviço de catálogo foi bem-sucedida.
3. Se você estiver criando servidores de catálogo nos servidores de aplicativos existentes, reinicie os servidores selecionados. O serviço de catálogo é automaticamente iniciado nos servidores selecionados.

O que Fazer Depois

Também é possível criar esta configuração usando a ferramenta wsadmin. Consulte "Tarefas Administrativas de Domínio do Serviço de Catálogo" para obter mais informações sobre os comandos.

Tarefas Administrativas de Domínio do Serviço de Catálogo

É possível usar os idiomas de script Jacl ou Jython para gerenciar os domínios de serviço de catálogo em sua configuração do WebSphere Application Server.

Requisitos

É necessário ter o WebSphere eXtreme Scale Client instalado em seu ambiente do WebSphere Application Server.

Lista de Todas as Tarefas Administrativas

Para obter uma lista de todas as tarefas administrativas associadas aos domínios do serviço de catálogo, execute o seguinte comando com wsadmin:

```
wsadmin>$AdminTask help XSDomainManagement
```

Comandos

As tarefas administrativas para os domínios do serviço de catálogo incluem os seguintes comandos:

- "createXSDomain"
- "deleteXSDomain" na página 347
- "getDefaultXSDomain" na página 347
- "listXSDomains" na página 348
- "modifyXSDomain" na página 348
- "testXSDomainConnection" na página 349
- "testXSServerConnection" na página 350

createXSDomain

O comando createXSDomain registra um novo domínio do serviço de catálogo.

Tabela 19. Argumentos de Comando createXSDomain

Argumento	Descrição
-name (obrigatório)	Especifica o nome do domínio do serviço de catálogo que você deseja editar.
-default	Especifica se o domínio do serviço de catálogo é o padrão para a célula. O valor-padrão é `true`. (Booleano: configurar para `true` ou `false`)
-userID	Especifica o ID do usuário para o domínio do serviço de catálogo.
-password	Especifica a senha para o domínio do serviço de catálogo.
-properties	Especifica as propriedades customizadas para o domínio do serviço de catálogo.

Tabela 20. Argumentos da Etapa defineDomainServers

Argumento	Descrição
-name	Especifica o nome do terminal do serviço de catálogo.
-hostName	Especifica o nome do host no qual reside o terminal.
-endPoints	Especifica os números de porta para o terminal do domínio do serviço de catálogo.
-properties	Especifica propriedades customizadas para o terminal do domínio do serviço de catálogo.

Valor de retorno:

Uso de exemplo do modo em lote

- Usando Jacl:

```
$AdminTask createXSDomain {-name TestDomain -default true -userID xsuser
-password ***** -defineDomainServers {{cs1 xhost1.ibm.com "" 5501,2809,1099}
{cs2 xhost2.ibm.com "" 5501,2809,1099}}}
```

- Usando a cadeia Jython:

```
AdminTask.createXSDomain('[-name TestDomain -default true -userID xsuser
-password ***** -defineDomainServers [[cs1 xhost1.ibm.com "" 5501,2809,1099]
[cs2 xhost2.ibm.com "" 5501,2809,1099]]]')
```

Uso de exemplo do modo interativo

- Usando Jacl:

```
$AdminTask createXSDomain {-interactive}
```

- Usando a cadeia Jython:

```
AdminTask.createXSDomain ('[-interactive]')
```

deleteXSDomain

O comando deleteXSDomain exclui um domínio do serviço de catálogo.

Parâmetros obrigatórios:

-name
Especifica o nome do domínio do serviço de catálogo a excluir.

Valor de retorno:

Uso de exemplo do modo em lote

- Usando Jacl:

```
$AdminTask deleteXSDomain {-name TestDomain }
```

- Usando a cadeia Jython:

```
AdminTask.deleteXSDomain('[-name TestDomain ]')
```

Uso de exemplo do modo interativo

- Usando Jacl:

```
$AdminTask deleteXSDomain {-interactive}
```

- Usando a cadeia Jython:

```
AdminTask.deleteXSDomain ('[-interactive]')
```

getDefaultXSDomain

O comando getDefaultXSDomain retorna o domínio do serviço de catálogo padrão para a célula.

Parâmetros obrigatórios: Nenhum

Valor de retorno: O nome do domínio do serviço de catálogo padrão.

Uso de exemplo do modo em lote

- Usando Jacl:

```
$AdminTask getDefaultXSDomain
```

- Usando a cadeia Jython:

```
AdminTask.getDefaultXSDomain
```

Uso de exemplo do modo interativo

- Usando Jacl:

```
$AdminTask getDefaultXSDomain {-interactive}
```

- Usando a cadeia Jython:

```
AdminTask.getDefaultXSDomain ('[-interactive]')
```

listXSDomains

O comando listXSDomains retorna uma lista dos domínios do serviço de catálogo existentes.

Parâmetros obrigatórios: Nenhum

Valor de retorno: Uma lista de todos os domínios do serviço de catálogo na célula.

Uso de exemplo do modo em lote

- Usando Jacl:

```
$AdminTask listXSDomains
```

- Usando a cadeia Jython:

```
AdminTask.listXSDomains
```

Uso de exemplo do modo interativo

- Usando Jacl:

```
$AdminTask listXSDomains {-interactive}
```

- Usando a cadeia Jython:

```
AdminTask.listXSDomains ('[-interactive]')
```

modifyXSDomain

O comando modifyXSDomain modifica um domínio do serviço de catálogo existente.

Tabela 21. Argumentos de Comando modifyXSDomain

Argumento	Descrição
-name (obrigatório)	Especifica o nome do domínio do serviço de catálogo que você deseja editar.
-default	Se configurado para `true`, especifica que o domínio do serviço de catálogo selecionado é o padrão para a célula. (Booleano)
-userID	Especifica o ID do usuário para o domínio do serviço de catálogo.
-password	Especifica a senha para o domínio do serviço de catálogo.
-properties	Especifica as propriedades customizadas para o domínio do serviço de catálogo.

Tabela 22. Argumentos da Etapa modifyEndpoints

Argumento	Descrição
-name	Especifica o nome do terminal do serviço de catálogo.

Tabela 22. Argumentos da Etapa modifyEndpoints (continuação)

Argumento	Descrição
-hostName	Especifica o nome do host no qual reside o terminal.
-endPoints	Especifica os números de porta para o terminal do domínio do serviço de catálogo.

Tabela 23. Argumentos da Etapa addEndpoints

Argumento	Descrição
-name	Especifica o nome do terminal do serviço de catálogo.
-hostName	Especifica o nome do host no qual reside o terminal.
-endPoints	Especifica os números de porta para o terminal do domínio do serviço de catálogo.
-properties	Especifica propriedades customizadas para o terminal do domínio do serviço de catálogo.

Tabela 24. Argumentos da Etapa removeEndpoints

Argumento	Descrição
-name	Especifica o nome do terminal do serviço de catálogo a excluir.

Valor de retorno:

Uso de exemplo do modo em lote

- Usando Jacl:

```
$AdminTask modifyXSDomain {-name TestDomain -userID newuser -password *****
-default false -modifyEndpoints {{cs1 xhost1.ibm.com "" 5502,2809,1099}}
-addEndpoints {{cs3 xhost3.ibm.com "" 5501,2809,1099}}
-removeEndpoints {{cs2}}}
```

- Usando a cadeia Jython:

```
AdminTask.modifyXSDomain('[-name TestDomain -userID newuser -password *****
-default false -modifyEndpoints [[cs1 xhost1.ibm.com "" 5502,2809,1099]]
-addEndpoints [[cs3 xhost3.ibm.com "" 5501,2809,1099]]
-removeEndpoints [[cs2]]]')
```

Uso de exemplo do modo interativo

- Usando Jacl:

```
$AdminTask modifyXSDomain {-interactive}
```

- Usando a cadeia Jython:

```
AdminTask.modifyXSDomain ('[-interactive]')
```

testXSDomainConnection

O comando testXSDomainConnection testa a conexão com um domínio do serviço de catálogo.

Parâmetros obrigatórios:

-name
Especifica o nome do domínio do serviço de catálogo para o qual testar a conexão.

Parâmetros Opcionais

-timeout
Especifica o período máximo de tempo a aguardar a conexão, em segundos.

Valor de retorno: Se uma conexão puder ser feita, retornará `true`, caso contrário, serão retornadas informações de erro de conexão.

Uso de exemplo do modo em lote

- Usando Jacl:
  ```
  $Admintask testXSDomainConnection
  ```
- Usando a cadeia Jython:
  ```
  AdminTask.testXSDomainConnection
  ```

Uso de exemplo do modo interativo

- Usando Jacl:
  ```
  $AdminTask testXSDomainConnection {-interactive}
  ```
- Usando a cadeia Jython:
  ```
  AdminTask.testXSDomainConnection ('[-interactive]')
  ```

testXSServerConnection

O comando testXSServerConnection testa a conexão com um servidor de catálogos. Este comando funciona para servidores independentes e para servidores que fazem parte de um domínio do serviço de catálogo.

Parâmetros obrigatórios:

host
Especifica o host no qual reside o servidor de catálogos.

listenerPort
Especifica a porta listener para o servidor de catálogos.

Parâmetros Opcionais

timeout
Especifica o período máximo de tempo a aguardar uma conexão para o servidor de catálogos, em segundos.

Valor de retorno:

Uso de exemplo do modo em lote

- Usando Jacl:
  ```
  $Admintask testXSServerConnection {-host xhost1.ibm.com -listenerPort 2809}
  ```
- Usando a cadeia Jython:
  ```
  AdminTask.testXSServerConnection('[-host xshost3.ibm.com -listenerPort 2809]')
  ```

Uso de exemplo do modo interativo

- Usando Jacl:
  ```
  $AdminTask testXSServerConnection {-interactive}
  ```

- Usando a cadeia Jython:

```
AdminTask.testXSServerConnection ('[-interactive]')
```

Iniciando Contêineres do eXtreme Scale Automaticamente em Aplicativos WebSphere Application Server

Processos de contêiner em um ambiente WebSphere Application Server iniciam automaticamente quando é iniciado um módulo que possui os arquivos XML do eXtreme Scale incluídos.

Antes de Iniciar

WebSphere Application Server e WebSphere eXtreme Scale devem estar instalados e você deve estar apto a acessar o console administrativo do WebSphere Application Server.

Sobre Esta Tarefa

Aplicativos Java Platform, Enterprise Edition possuem regras de utilitário de classe complexas que complicam muito o carregamento de classes ao utilizar um WebSphere eXtreme Scale compartilhado em um servidor Java EE. Um aplicativo Java EE normalmente é um arquivo Enterprise Archive (EAR) único com um ou mais módulos Enterprise JavaBeans (EJB) ou Web archive (WAR) implementados nele.

O WebSphere eXtreme Scale procura cada início de módulo e consulta arquivos XML do eXtreme Scale. Se WebSphere eXtreme Scale detectar que um módulo inicia com os arquivos XML, ele registra o servidor de aplicativos como uma Java Virtual Machine de contêiner do eXtreme Scale (JVM) com o serviço de catálogo. Ao registrar os contêineres com o serviço de catálogo, o mesmo aplicativo pode ser implementado em grades diferentes e ainda ser tratado como uma grade única pelo serviço de catálogo. O serviço de catálogo não é referente às células, grades ou grades dinâmicas. Tudo é uma JVM de contêiner do eXtreme Scale ou não. Uma única grade lógica pode ampliar várias células do WebSphere Extended Deployment, se necessário.

Procedimento

1. Empacote seu arquivo EAR para ter módulos que incluem os arquivos XML do eXtreme Scale na pasta META-INF. O WebSphere eXtreme Scale detecta a presença dos arquivos `objectGrid.xml` e `objectGridDeployment.xml` na pasta META-INF de módulos EJB e da WEB quando eles iniciam. Se apenas um arquivo `objectGrid.xml` for localizado, então a JVM é assumida como sendo o cliente, caso contrário, assume-se que esta JVM atua como um contêiner para a grade que é definida no arquivo `objectGridDeployment.xml`.

   É necessário utilizar os nomes corretos para estes arquivos XML. Os nomes de arquivos fazem distinção entre maiúsculas e minúsculas. Se os arquivos não estiverem presentes, então o contêiner não será iniciado. É possível verificar no arquivo `systemout.log` se há mensagens que indicam que os shards estão sendo localizados. Um módulo EJB ou módulo WAR utilizando o eXtreme Scale deve ter arquivos XML do eXtreme Scale em seu diretório `META-INF`.

   Os arquivos XML do eXtreme Scale incluem:

   - Um arquivo `objectGrid.xml`, comumente referido como um arquivo XML descritor do eXtreme Scale.
   - Um arquivo `objectGridDeployment.xml`, comumente mencionado com um arquivo XML descritor de implementação.

- Um arquivo `entity.xml`, se forem utilizadas entidades (opcional). O nome do arquivo `entity.xml` deve corresponder ao nome que é especificado no arquivo `objectGrid.xml`.

O tempo de execução do eXtreme Scale detecta estes arquivos e, em seguida, entra em contato com o serviço de catálogo para informá-lo que outro contêiner está disponível para hospedar shards para tal eXtreme Scale.

Dica: Se você planejar testar um aplicativo com entidades usando um servidor eXtreme Scale, configure o valor minSyncReplicas para `0` no arquivo XML do descritor de implementação. Caso contrário, é possível ver uma das mensagens a seguir no arquivo `SystemOut.log`, pois o posicionamento não pode ocorrer até que outro servidor seja iniciado para atender a política minSyncReplica:

```
CWPRJ1005E: Erro ao resolver associação de entidade. Entity=entity_name,
association=association_name.

CWOBJ3013E: O repositório EntityMetadata não está disponível.  Tempo Limite
limite atingido ao tentar registrar a entidade: entity_name.
```

2. Implementar e iniciar seu aplicativo.

   O contêiner inicia automaticamente quando o módulo é iniciado. O serviço de catálogo inicia para colocar primários e réplicas de partição (shards) o quanto antes. Esta disposição ocorre imediatamente, a menos que você defina um atributo numInitialContainers no arquivo `objectGridDeployment.xml`. Se você definir o atributo numInitialContainers, então a disposição inicia quando tal número de contêineres for iniciado.

O que Fazer Depois

Aplicativos na mesma célula que os contêineres podem conectar-se a estas grades utilizando um método ObjectGridManager.connect(null, null) e, então, chama o método getObjectGrid(ccc, "object grid name"). Os métodos connect ou getObjectGrid podem ser bloqueados até que os contêineres tenham colocado os shards, mas este bloqueio é um problema apenas quando a grade está iniciando.

ClassLoaders

Quaisquer plug-ins ou objetos armazenados em um eXtreme Scale são carregados em um determinado utilitário de carga de classes. Dois módulos EJB no mesmo EAR podem incluir esses objetos. Os objetos são o mesmos, mas são carregados utilizando ClassLoaders diferentes. Se o aplicativo A armazena um objeto Person em um Mapa do eXtreme Scale que é local para o servidor, o aplicativo B recebe um ClassCastException se ele tentar ler tal objeto. Esta exceção ocorre porque o aplicativo B carregou o objeto Person em um utilitário de carga de classe diferente.

Uma abordagem para resolver este problema é ter um módulo raiz que contém os plug-ins e objetos necessários que estão armazenados no eXtreme Scale. Cada módulo que utiliza o eXtreme Scale deve referenciar tal módulo para suas classes. Outra resolução é colocar este objetos compartilhados em um jar utilitário que está em um utilitário de carga de classe comum compartilhado por ambos os módulos e aplicativos. Os objetos também podem ser colocados nas classes do WebSphere ou no diretório `lib/ext`, mas isso dificulta a implementação e não é recomendado.

Os módulos EJB em um arquivo EAR normalmente compartilham o mesmo ClassLoader e não são afetados por este problema. Cada módulo WAR possui seu próprio ClassLoader e é afetado por este problema.

Conectando a uma grade como um cliente apenas

Se a propriedade catalog.services.cluster for definida na célula, nas propriedades customizadas de nó ou servidor, qualquer módulo no arquivo EAR poderá chamar o método ObjectGridManager.connect (ServerFactory.getServerProperties().getCatalogServiceBootstrap(), null, null) para obter um ClientClusterContext e chamar o método ObjectGridManager.getObjectGrid(ccc, "grid name") para obter uma referência para o eXtreme Scale. Se qualquer objeto do aplicativo for armazenado em Mapas, verifique se tais objetos estão presentes em um ClassLoader comum.

Os clientes do Java Platform, Standard Edition ou clientes fora da célula podem conectar-se utilizando a porta IIOP de autoinicialização do serviço de catálogo (o padrão no Websphere é o gerenciador de implementação) para obter um ClientClusterContext e, em seguida, obter um eXtreme Scale denominado da maneira usual.

Entity Manager

O Entity Manager ajuda porque as Tuplas são armazenadas no Mapas, não os objetos do aplicativo, de forma que há menos problemas com o utilitário de carga de classe. Entretanto, os plug-ins podem ser um problema. Observe também que uma substituição de cliente do arquivo XML descritor do eXtreme Scale sempre é necessária ao conectar-se a um eXtreme Scale que possui entidades definidas: ObjectGridManager.connect("host:port[,host:port], null, objectGridOverride) ou ObjectGridManager.connect(null, objectGridOverride).

Administrando Programaticamente com Beans Gerenciados (MBeans)

É possível usar vários tipos diferentes de Java Management Extensions (JMX) MBeans para administrar e monitorar as implementações. Cada MBean refere-se a uma entidade específica, como um mapa, grade de dados, servidor ou serviço.

Interfaces MBean JMX e WebSphere eXtreme Scale

Cada MBean possui métodos get que representam valores de atributo. Estes métodos get não podem ser chamados diretamente a partir do seu programa. A especificação JMX trata os atributos de maneira diferente das operações. É possível visualizar atributos com um console JMX do fornecedor e executar operações em seu programa ou com um console JMX do fornecedor.

Pacote com.ibm.websphere.objectgrid.management

Consulte a documentação da API gerada para obter uma visão geral e especificações de programação detalhadas para todos os MBeans disponíveis:Pacote com.ibm.websphere.objectgrid.management

Acessando MBeans Utilizando a Ferramenta wsadmin

É possível usar o utilitário wsadmin fornecido no WebSphere Application Server para acessar informações de MBean.

Execute a ferramenta wsadmin a partir do diretório bin em sua instalação do WebSphere Application Server. O exemplo a seguir recupera uma visualização da disposição do shard atual em um eXtreme Scale dinâmico. O wsadmin pode ser executado a partir de qualquer instalação na qual o eXtreme Scale está em execução. Não é necessário executar o wsadmin no serviço de catálogo.

```
$ wsadmin.sh -lang jython
wsadmin>placementService = AdminControl.queryNames
 ("com.ibm.websphere.objectgrid:*,type=PlacementService")
wsadmin>print AdminControl.invoke(placementService,
 "listObjectGridPlacement","library ms1")

<objectGrid name="library" mapSetName="ms1">
  <container name="container-0" zoneName="DefaultDomain"
  hostName="host1.company.org" serverName="server1">
     <shard type="Primary" partitionName="0"/>
     <shard type="SynchronousReplica" partitionName="1"/>
  </container>
  <container name="container-1" zoneName="DefaultDomain"
  hostName="host2.company.org" serverName="server2">
     <shard type="SynchronousReplica" partitionName="0"/>
     <shard type="Primary" partitionName="1"/>
  </container>
  <container name="UNASSIGNED" zoneName="_ibm_SYSTEM"
  hostName="UNASSIGNED" serverName="UNNAMED">
    <shard type="SynchronousReplica" partitionName="0"/>
    <shard type="AsynchronousReplica" partitionName="0"/>
  </container>
</objectGrid>
```

Capítulo 8. Protegendo o Ambiente de Implementação

O WebSphere® eXtreme Scale pode proteger o acesso a dados, incluindo permissão para integração com provedores de segurança externos. Os aspectos de segurança incluem autenticação, autorização, segurança de transporte, segurança de grade, segurança local e segurança JMX (Mbean).

Ativando a Segurança Local

O WebSphere eXtreme Scale fornece vários terminais de segurança para integrar mecanismos customizados. No modelo de programação local, a principal função de segurança é a autorização e não possui suporte à autenticação. É necessário autenticar fora do WebSphere Application Server já existente. Entretanto, são fornecidos plug-ins para obter e validar objetos Subject.

As seguintes seções descrevem as duas maneiras nas quais é possível ativar a segurança local:

É possível utilizar o arquivo XML do ObjectGrid para definir um ObjectGrid e ativar a segurança para esse ObjectGrid. O arquivo `secure-objectgrid-definition.xml`, que é utilizado na amostra do aplicativo corporativo ObjectGridSample, é mostrado no seguinte exemplo. Configure o atributo securityEnabled como `true` para ativar a segurança.

```
<objectGrids>
    <objectGrid name="secureClusterObjectGrid" securityEnabled="true"
        authorizationMechanism="AUTHORIZATION_MECHANISM_JAAS">
    ...
</objectGrids>
```

Também é possível ativar a segurança programaticamente. Para criar um ObjectGrid usando o método ObjectGrid.setSecurityEnabled, chame o seguinte método na interface ObjectGrid:

```
/**
 * Enable the ObjectGrid security
 */
void setSecurityEnabled();
```

Para obter informações adicionais, consulte na documentação da API.

Autenticação de Grade

É possível utilizar o plug-in do gerenciador de token seguro para ativar a autenticação servidor-para-servidor, que requer que você implemente a interface SecureTokenManager.

O método generateToken(Object) obtém uma proteção de objeto, e depois gera um token que não pode ser compreendido pelos outros. O método verifyTokens(byte[]) faz o processo inverso: converte o token de volta ao objeto original.

Uma implementação SecureTokenManager simples usa um algoritmo de codificação simples, como um algoritmo XOR, para codificar o objeto na forma serializada e depois usa o algoritmo de codificação correspondente para decodificar o token. Esta implementação não é segura e é fácil de ser interrompida.

Implementação padrão do **WebSphere eXtreme Scale**

O WebSphere eXtreme Scale fornece uma implementação imediatamente disponível para esta interface. Esta implementação padrão utiliza um par de chaves para assinar e verificar a assinatura e utiliza uma chave secreta para criptografar o conteúdo. Cada servidor tem um armazenamento de chaves de tipo JCKES para armazenar o par de chaves, uma chave privada e uma chave pública e uma chave secreta. O armazenamento de chaves tem que ser do tipo JCKES para armazenar as chaves secretas. Estas chaves são utilizadas para criptografar e assinar ou verificar a cadeia de segredo na extremidade de envio. Além disso, o token está associado ao tempo de expiração. Na extremidade de recebimento, os dados são verificados, decriptografados e comparados com a cadeia de segredo do receptor. Os protocolos de comunicação Secure Sockets Layer (SSL) não são necessários entre um par de servidores para autenticação, porque as chaves privadas e as chaves públicas servem para a mesma finalidade. No entanto, se a comunicação do servidor não for criptografada, os dados poderão ser roubados por violação na comunicação. Como o token expira em breve, a ameaça de ataque à reprodução é minimizada. Esta possibilidade é significativamente reduzida se todos os servidores forem implementados atrás de um firewall.

A desvantagem desta abordagem é que os administradores do WebSphere eXtreme Scale precisam gerar chaves e transportá-las para todos os servidores, o que pode causar violação de segurança durante o transporte.

Segurança de Grade

A segurança de grade do WebSphere eXtreme Scale garante que um servidor que está se juntando tenha as credenciais certas, assim um servidor doloso não pode juntar-se à grade. A segurança de grade utiliza um mecanismo de cadeia de segredo compartilhado.

Todos os servidores WebSphere eXtreme Scale, incluindo servidores de catálogo, concordam quanto a uma cadeia de segredo compartilhado. Quando um servidor se junta à grade, ele é desafiado a apresentar a cadeia secreta. Se a cadeia secreta do servidor que está se juntando corresponder à cadeia no servidor presidente ou servidor de catálogo, o servidor que está se juntando é aceito. Se a cadeia não corresponder, o pedido de junção é rejeitado.

Não é seguro enviar um segredo em texto não-criptografado. A infraestrutura de segurança do WebSphere eXtreme Scale fornece um plug-in do gerenciador de token seguro para permitir que o servidor proteja este segredo antes de enviar. Você deve decidir como implementar a operação de proteção. O WebSphere eXtreme Scale fornece uma implementação pronta para usar, na qual a operação segura é implementada para criptografar e assinar o segredo.

A cadeia do segredo é configurada no arquivo `server.properties`. Consulte o “Arquivo de Propriedades do Servidor” na página 183 para obter mais informações sobre a propriedade authenticationSecret.

Plug-in SecureTokenManager

Um plug-in do gerenciador de token de segurança é representado pela interface com.ibm.websphere.objectgrid.security.plugins.SecureTokenManager.

Para obter mais informações sobre o plug-in SecureTokenManager, consulte aDocumentação da API SecureTokenManager.

O método generateToken(Object) obtém um objeto, e depois gera um token que não pode ser compreendido pelos outros. O método verifyTokens(byte[]) faz o processo inverso: o método converte o token de volta ao objeto original.

Uma implementação SecureTokenManager simples usa um algoritmo de codificação simples, como um algoritmo exclusivo ou (XOR), para codificar o objeto na forma serializada e depois usa o algoritmo de codificação correspondente para codificar o token. Essa implementação não é segura.

O WebSphere eXtreme Scale fornece uma implementação imediatamente disponível para esta interface.

A implementação padrão utiliza um par de chaves para assinar e verificar a assinatura e utiliza uma chave secreta para criptografar o conteúdo. Cada servidor tem um armazenamento de chaves de tipo JCKES para armazenar o par de chaves, uma chave privada e uma chave pública e uma chave secreta. O armazenamento de chaves tem que ser do tipo JCKES para armazenar as chaves secretas.

Estas chaves são utilizadas para criptografar e assinar ou verificar a cadeia de segredo na extremidade de envio. Além disso, o token está associado ao tempo de expiração. Na extremidade de recebimento, os dados são verificados, decriptografados e comparados com a cadeia de segredo do receptor. Os protocolos de comunicação Secure Sockets Layer (SSL) não são necessários entre um par de servidores para autenticação, porque as chaves privadas e as chaves públicas servem para a mesma finalidade. No entanto, se a comunicação do servidor não for criptografada, os dados poderão ser roubados por violação na comunicação. Como o token expira em breve, a ameaça de ataque à reprodução é minimizada. Esta possibilidade é significativamente reduzida se todos os servidores forem implementados atrás de um firewall.

A desvantagem desta abordagem é que os administradores do WebSphere eXtreme Scale precisam gerar chaves e transportá-las para todos os servidores, o que pode causar violação de segurança durante o transporte.

Scripts de Amostra para Criar Propriedades do Gerenciador de Token Seguro

Conforme observado na seção anterior, é possível criar um keystore contendo um par de chaves para assinar e verificar a assinatura e uma chave secreta para criptografar conteúdo.

Por exemplo, é possível utilizar o comando JDK 6 keytool para criar a chave da seguinte forma:

```
keytool -genkeypair -alias keypair1 -keystore key1.jck -storetype
JCEKS -keyalg
rsa -dname "CN=sample.ibm.com, OU=WebSphere eXtreme Scale" -storepass
key111 -keypass
keypair1 -validity 10000

keytool -genseckey -alias seckey1 -keystore key1.jck -storetype JCEKS
-keyalg
DES  -storepass key111 -keypass seckey1 -validity 1000
```

Esses dois comandos criam um par de chaves "keypair1" e uma chave secreta "seckey1". É possível então configurar o seguinte no arquivo de propriedades do servidor:

```
secureTokenKeyStore=key1.jck
secureTokenKeyStorePassword=key111
secureTokenKeyStoreType=JCEKS
```

```
secureTokenKeyPairAlias=keypair1
secureTokenKeyPairPassword=keypair1
secureTokenSecretKeyAlias=seckey1
secureTokenSecretKeyPassword=seckey1
secureTokenCipherAlgorithm=DES
secureTokenSignAlgorithm=RSA
```

Configuração

Consulte Propriedades do servidor para obter mais informações sobre as propriedades que você usa para configurar o gerenciador de token seguro.

Autenticação de Cliente do Aplicativo

A autenticação de cliente do aplicativo consiste em ativar a autenticação de credencial e de segurança cliente/servidor e de configurar um autenticador e um gerador de credencial de sistema.

Ativação da Segurança do Cliente-Servidor

A segurança deve ser ativada no cliente e no servidor para poder autenticar-se com êxito com o ObjectGrid.

Ativar a Segurança do Cliente

O WebSphere eXtreme Scale fornece um arquivo de amostra de propriedade do cliente, o arquivo `sampleClient.properties` no diretório *WAS_HOME*`/optionalLibraries/ObjectGrid/properties` para uma instalação do WebSphere Extended Deployment ou no diretório `/ObjectGrid/properties` em uma instalação do servidor combinada. É possível modificar este arquivo de gabarito com valores apropriados. Configure a propriedade securityEnabled no arquivo `objectgridClient.properties` para `true`. A propriedade securityEnabled indica se a segurança está ativada. Quando um cliente se conecta a um servidor, os valores no lado do cliente e do servidor devem ser ambos `true` ou ambos `false`. Por exemplo, se a segurança do servidor conectado estiver ativada, o valor da propriedade deverá ser configurado como `true` na lado do cliente para que o cliente se conecte ao servidor.

A interface com.ibm.websphere.objectgrid.security.config.ClientSecurityConfiguration representa o arquivo `security.ogclient.props`. É possível usar a API pública com.ibm.websphere.objectgrid.security.config.ClientSecurityConfigurationFactory para criar uma instância desta interface com valores padrão ou criar uma instância passando o perfil de propriedade de segurança do cliente do ObjectGrid. O arquivo `security.ogclient.props` contém outras propriedades. Consulte a Documentação da API ClientSecurityConfiguration e a Documentação da API ClientSecurityConfigurationFactory para obter mais detalhes.

Ativação da Segurança do Servidor

Para ativar a segurança no lado do servidor, é possível configurar a propriedade **securityEnabled** no arquivo `security.xml` para `true`. Use um arquivo XML do descritor de segurança para especificar a configuração de segurança da grade de dados para isolar a configuração de segurança de toda a grade da configuração sem segurança.

Ativando a Autenticação de Credencial

Após o cliente do eXtreme Scale recuperar o objeto Credential usando o objeto CredentialGenerator, o objeto Credential é enviado juntamente com o pedido do cliente para o servidor eXtreme Scale. O servidor autentica o objeto Credential antes de processar o pedido. Se o objeto Credential for autenticado com êxito, um objeto Subject será retornado para representar este objeto Credential. Este objeto Subject é, então, usado para autorizar o pedido.

Configure a propriedade **credentialAuthentication** nos arquivos de propriedades do cliente e do servidor para ativar a autenticação de credencial. Para obter mais informações, consulte "Arquivo de Propriedades do Cliente" na página 202 e "Arquivo de Propriedades do Servidor" na página 183.

As tabelas a seguir exibem qual mecanismo de autenticação será utilizado em diferentes configurações.

Tabela 25. Autenticação de credencial nas configurações do cliente e do servidor

Autenticação de Credencial do Cliente	Autenticação de Credencial do Servidor	Resultado
Não	Nunca	Desativada
Não	Suportado	Desativada
Não	Requerido	Caso de erro
Suportado	Nunca	Desativada
Suportado	Suportado	Ativado
Suportado	Requerido	Ativado
Requerido	Nunca	Caso de erro
Requerido	Suportado	Ativado
Requerido	Requerido	Ativado

Configuração de um autenticador

O servidor eXtreme Scale usa o plug-in do Autenticador para autenticar o objeto Credential. Uma implementação da interface do Autenticador obtém o objeto Credential e, em seguida, autentica-o para um registro do usuário, por exemplo, um servidor Lightweight Directory Access Protocol (LDAP) e assim por diante. O eXtreme Scale não fornece uma configuração de registro. A conexão com um registro do usuário e a autenticação nele devem ser implementadas neste plug-in.

Por exemplo, uma implementação do Autenticador extrai o ID do usuário e a senha da credencial, usa-os para conectar e validar um servidor LDAP e cria um objeto Subject como resultado da autenticação. A implementação pode utilizar os módulos de login do Java Authentication and Authorization Service (JAAS). Um objeto Subject é retornado como resultado da autenticação.

É possível configurar o autenticador no arquivo XML descritor de segurança, como mostra o seguinte exemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<securityConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://ibm.com/ws/objectgrid/config/security ../objectGridSecurity.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config/security">

 <security securityEnabled="true" loginSessionExpirationTime="300" >
```

```
  <authenticator className ="com.ibm.websphere.objectgrid.security.plugins.
builtins.KeyStoreLoginAuthenticator">
        </authenticator>

 </security>

</securityConfig>
```

Use a opção **-clusterSecurityFile** ao iniciar um servidor seguro para configurar o arquivo XML de segurança. Consulte o tutorial de segurança SE Java no *Visão Geral do Produto* para obter mais informações.

Configuração de um Gerador de Credencial do Sistema

O gerador de credenciais do sistema é utilizado para representar um factory para a credencial do sistema. Uma credencial de sistema é semelhante para uma credencial de administrador. É possível configurar o elemento SystemCredentialGenerator no XML de segurança de catálogo, conforme mostrado no seguinte exemplo:

```
<systemCredentialGenerator className ="com.ibm.websphere.objectgrid.security.plugins.
 builtins.UserPasswordCredentialGenerator">
          <property name="properties" type="java.lang.String" value="manager manager1"
      description="username password" />
 </systemCredentialGenerator>
```

Para demonstração, o nome de usuário e senha são armazenados em texto limpo. Não armazene o nome de usuário e senha em texto limpo em um ambiente de produção.

O WebSphere eXtreme Scale fornece um gerador de credencial de sistema padrão, que usa as credenciais do servidor. Se você não especificar explicitamente o gerador de credencial de sistema, este gerador de credencial de sistema padrão é usado.

Autorização do Aplicativo Cliente

A autorização do cliente do aplicativo consiste de classes de permissão do ObjectGrid, de mecanismos de autorização, de um período de verificação de permissão e de autorização de acesso apenas pelo criador.

Para o eXtreme Scale, a autorização se baseia no objeto Subject e nas permissões. O produto suporta dois tipos de mecanismos de autorização: Java Authentication and Authorization Service (JAAS) e autorização customizada.

Classes de permissão do ObjectGrid

A autorização é baseada em permissões. A seguir há quatro tipos diferentes de classes de permissão.

- A classe MapPermission representa as permissões para acessar os dados nos mapas do ObjectGrid.
- A classe ObjectGridPermission representa as permissões para acessar o ObjectGrid.
- A classe ServerMapPermission representa as permissões para acessar os mapas do ObjectGrid no lado do servidor a partir de um cliente.
- A classe AgentPermission representa as permissões para iniciar um agente no lado do servidor.

Para obter mais informações sobre as APIs e permissões associadas, consulte o tópico sobre a programação de autorização do cliente no *Guia de Programação*.

Período de verificação de permissão

O eXtreme Scale suporta armazenamento em cache dos resultados da verificação de permissão de mapa por motivo de desempenho. Sem este mecanismo, quando um método listado na Lista de Métodos com suas permissões necessárias é chamado, o tempo de execução chama o mecanismo de autorização configurado para autorizar acesso. Com este período de verificação de permissão configurado, o mecanismo de autorização é chamado periodicamente com base no período de verificação de permissão.

As informações de autorização da permissão são baseadas no objeto Subject. Quando um cliente tenta acessar os métodos, o tempo de execução do eXtreme Scale consulta o cache com base no objeto Subject. Se o objeto não puder ser localizado no cache, o tempo de execução verifica as permissões concedidas para este objeto Subject, e, então, armazena as permissões em um cache.

O período de verificação de permissão deve ser definido antes da inicialização do ObjectGrid. O período de verificação de permissão pode ser configurado de duas maneiras:

É possível utilizar o arquivo XML do ObjectGrid para definir um ObjectGrid e configurar o período de verificação de permissão. No exemplo a seguir, o período de verificação de permissão é configurado para 45 segundos:

```
<objectGrids>
 <objectGrid name="secureClusterObjectGrid" securityEnabled="true"
 authorizationMechanism="AUTHORIZATION_MECHANISM_JAAS"
 permissionCheckPeriod="45">
  <bean id="bean id="TransactionCallback"
className="com.ibm.websphere.samples.objectgrid.HeapTransactionCallback" />
...
</objectGrids>
```

Se desejar criar um ObjectGrid com APIs, chame o seguinte método para configurar o período de verificação de permissão. Este método pode ser chamado apenas antes da inicialização da instância do ObjectGrid. Este método se aplica apenas ao modelo de programação do eXtreme Scale local quando você utiliza a instância do ObjectGrid diretamente.

```
/**
 * This method takes a single parameter indicating how often you
 * want to check the permission used to allow a client access. If the
 * parameter is 0 then every single get/put/update/remove/evict call
 * asks the authorization mechanism, either JAAS authorization or custom
 * authorization, to check if the current subject has permission. This might be
 * prohibitively expensive from a performance point of view depending on
 * the authorization implementation, but if you need to have ever call check the
 * authorization mechanism, then set the parameter to 0.
 * Alternatively, if the parameter is > 0 then it indicates the number
 * of seconds to cache a set of permissions before returning to
 * the authorization mechanism to refresh them. This value provides much
 * better performance, but if the back-end
 * permissions are changed during this time then the ObjectGrid can
 * allow or prevent access even though the back-end security
 * provider was modified.
 *
 * @param period the permission check period in seconds.
 */
void setPermissionCheckPeriod(int period);
```

Autorização de acesso apenas pelo criador

A autorização de acesso apenas pelo criador garante que apenas o usuário (representado pelos objetos Principal associados a ele) que insere a entidade no mapa ObjectGrid possa acessar (ler, atualizar, invalidar e remover tal entrada.

O modelo existente de autorização do mapa do ObjectGrid é baseado no tipo de acesso mas não em entradas de dados. Em outras palavras, um usuário possui um tipo de acesso específico, tal como read, write, insert, delete ou invalidate, para todo os dados no mapa ou em nenhum dos dados. Entretanto, o eXtreme Scale não autoriza usuários para a entrada de dados individual. Este recurso oferece uma nova maneira de autorizar usuários para entradas de dados.

Em um cenário onde usuários diferentes acessam diferentes conjuntos de dados, este modelo pode ser útil. Quando o usuário carrega dados do armazenamento persistente nos mapas do ObjectGrid, o acesso pode ser autorizado pelo armazenamento persistente. Neste caso, não há necessidade de outra autorização na camada do mapa do ObjectGrid. É necessário apenas garantir que a pessoa que carrega os dados no mapa possa acessá-lo, permitindo o recurso de acesso apenas pelo criador.

Valores de atributos do modo apenas do criador:

desativada
O recurso de acesso apenas pelo criador é desativado.

complement
O recurso de acesso apenas pelo criador é ativado para complementar a autorização do mapa. Em outras palavras, a autorização de mapa e o recurso de acesso apenas pelo criador serão efetivados. Portanto, é possível limitar ainda mais as operações nos dados. Por exemplo, o criador não pode invalidar os dados.

supersede
O recurso de acesso apenas pelo criador é ativado para substituir a autorização do mapa. Em outras palavras, o recurso de acesso apenas pelo criador substituirá a autorização do mapa e nenhuma autorização de mapa será feita.

É possível configurar o modo de acesso apenas pelo criador de duas maneiras:

Pelo arquivo XML:

É possível utilizar o arquivo XML do ObjectGrid para definir um ObjectGrid e configurar o acesso de modo apenas pelo criador para `disabled`, `complement` ou `supersede`, como mostra o seguinte exemplo:

```
<objectGrids>
    <objectGrid name="secureClusterObjectGrid" securityEnabled="true"
         accessByCreatorOnlyMode="supersede"
        <bean id="TransactionCallback"
              classname="com.ibm.websphere.samples.objectgrid.HeapTransactionCallback" />
    ...
</objectGrids>
```

Programaticamente:

Se desejar criar um ObjectGrid programaticamente, é possível chamar o método a seguir para configurar o modo de acesso apenas pelo criador. A chamada deste método aplica-se somente ao modelo de programação local do eXtreme Scale quando você instancia diretamente a instância do ObjectGrid:

```
/**
 * Set the "access by creator only" mode.
 * Enabling "access by creator only" mode ensures that only the user (represented
 * by the Principals associated with it), who inserts the record into the map,
 * can access (read, update, invalidate, and remove) the record.
 * The "access by creator only" mode can be disabled, or can complement the
 * ObjectGrid authorization model, or it can supersede the ObjectGrid
 * authorization model. The default value is disabled:
 * {@link SecurityConstants#ACCESS_BY_CREATOR_ONLY_DISABLED}.
```

```
 * @see SecurityConstants#ACCESS_BY_CREATOR_ONLY_DISABLED
 * @see SecurityConstants#ACCESS_BY_CREATOR_ONLY_COMPLEMENT
 * @see SecurityConstants#ACCESS_BY_CREATOR_ONLY_SUPERSEDE
 *
 * @param accessByCreatorOnlyMode the access by creator mode.
 *
 * @since WAS XD 6.1 FIX3
*/
void setAccessByCreatorOnlyMode(int accessByCreatorOnlyMode);
```

Para ilustrar ainda mais, considere um cenário no qual uma conta de mapa do ObjectGrid está em uma grade financeira e Manager1 e o Employee1 são os dois usuários. A política de autorização do eXtreme Scale concede todas as permissões de acesso para o Manager1 e apenas a permissão de acesso de leitura para o Employee1. A política do JAAS para a autorização do mapa do ObjectGrid é mostrada no seguinte exemplo:

```
grant codebase "http://www.ibm.com/com/ibm/ws/objectgrid/security/PrivilegedAction"
    Principal com.acme.PrincipalImpl "Manager1" {
    permission com.ibm.websphere.objectgrid.security.MapPermission
        "banking.account", "all"
};
grant codebase "http://www.ibm.com/com/ibm/ws/objectgrid/security/PrivilegedAction"
    Principal com.acme.PrincipalImpl "Employee1" {
    permission com.ibm.websphere.objectgrid.security.MapPermission
        "banking.account", "read, insert"
};
```

Considere como o recurso de acesso apenas pelo criador afeta a autorização:

- **disabled** Se o recurso de acesso apenas pelo criador estiver desativado, a autorização de mapa não é diferente. O usuário "Manager1" pode acessar todos os dados no mapa "account". O usuário "Employee1" pode ler e inserir todos os dados no mapa, mas não pode atualizar, invalidar e remover nenhum dado no mapa.
- **complement** Se o recurso de acesso apenas pelo criador estiver ativado com a opção "complement", a autorização de mapa e a autorização de acesso apenas pelo criador estarão em vigor. O usuário "Manager1" pode acessar os dados no mapa "account", mas apenas se o usuário sozinho os carregou no mapa. O usuário "Employee1" pode ler os dados no mapa "account", mas apenas se tal usuário sozinho os carregou no mapa. (Entretanto, este usuário não pode atualizar, invalidar ou remover dados no mapa).
- **supersede** Se o recurso de acesso apenas pelo criador estiver ativado com a opção "supersede", a autorização do mapa não será imposta. A autorização de acesso apenas pelo criador será a política apenas de autorização. O usuário "Manager1" possui o mesmo privilégio que no modo "complement": este usuário pode acessar os dados no mapa "account" apenas se o mesmo usuário carregou os dados no mapa. Entretanto, o usuário "Employee1" agora possui acesso total aos dados no mapa "account" se este usuários os carregou no mapa. Em outras palavras, a política de autorização definida na política do Java Authentication and Authorization Service (JAAS) não será imposta.

Transport Layer Security e Secure Sockets Layer

O WebSphere eXtreme Scale suporta TCP/IP e Transport Layer Security/Secure Sockets Layer (TLS/SSL) para a comunicação segura entre clientes e servidores.

O TLS/SSL fornece comunicação segura entre o cliente e o servidor. O mecanismo de comunicação que é usado depende do valor do parâmetro transportType que está especificado nos arquivos de configuração do cliente e servidor.

É possível configurar a propriedade transportType nos seguintes arquivos de configuração de cliente e servidor:

- Para configurar a propriedade na configuração de segurança do cliente, consulte "Arquivo de Propriedades do Cliente" na página 202.
- Para configurar a propriedade na configuração de segurança do servidor de contêineres, consulte "Arquivo de Propriedades do Servidor" na página 183.
- Para configurar a propriedade na configuração de segurança do servidor de catálogos, consulte "Arquivo de Propriedades do Servidor" na página 183.

Tabela 26. Protocolo de Transporte a Ser Utilizado nas Configurações de Transporte do Cliente e Transporte do Servidor

Propriedade transportType do cliente	Propriedade transportType do servidor	Protocolo resultante
TCP/IP	TCP/IP	TCP/IP
TCP/IP	Suportado pelo SSL	TCP/IP
TCP/IP	Requerido pelo SSL	Erro
Suportado pelo SSL	TCP/IP	TCP/IP
Suportado pelo SSL	Suportado pelo SSL	SSL (se o SSL falhar, TCP/IP)
Suportado pelo SSL	Requerido pelo SSL	SSL
Requerido pelo SSL	TCP/IP	Erro
Requerido pelo SSL	Suportado pelo SSL	SSL
Requerido pelo SSL	Requerido pelo SSL	SSL

Quando é usado SSL, os parâmetros de configuração SSL devem ser fornecidos tanto do lado do cliente quanto do servidor. Em um ambiente Java SE, a configuração SSL é configurada nos arquivos de propriedades do cliente ou servidor. Se o cliente ou o servidor estiver em um WebSphere Application Server, será possível usar o suporte de segurança de transporte no WebSphere Application Server para configurar parâmetros SSL.

Configuração do arquivo `orb.properties` para Suporte de Segurança de Transporte

Use o TLS/SSL quando a propriedade transportType possuir um valor "SSL-Suportado".

Para suportar o transporte seguro em um ambiente Java Platform, Standard Edition, é necessário modificar o arquivo "Arquivo de Propriedades ORB" na página 192 para incluir as seguintes propriedades:

```
# IBM JDK properties
org.omg.CORBA.ORBClass=com.ibm.CORBA.iiop.ORB
org.omg.CORBA.ORBSingletonClass=com.ibm.rmi.corba.ORBSingleton
javax.rmi.CORBA.StubClass=com.ibm.rmi.javax.rmi.CORBA.StubDelegateImpl
javax.rmi.CORBA.PortableRemoteObjectClass=com.ibm.rmi.javax.rmi.PortableRemoteObject
javax.rmi.CORBA.UtilClass=com.ibm.ws.orb.WSUtilDelegateImpl

# WS Plugins
com.ibm.CORBA.ORBPluginClass.com.ibm.ws.orbimpl.transport.WSTransport
com.ibm.CORBA.ORBPluginClass.com.ibm.ws.orbimpl.WSORBPropertyManager
com.ibm.CORBA.ORBPluginClass.com.ibm.ISecurityUtilityImpl.SecurityPropertyManager

# WS Interceptors
org.omg.PortableInterceptor.ORBInitializerClass.com.ibm.ws.objectgrid.corba.ObjectGridInitializer
org.omg.PortableInterceptor.ORBInitializerClass.com.ibm.ISecurityLocalObjectBaseL13Impl.SecurityComponentFactory

# WS ORB & Plugins properties
com.ibm.ws.orb.transport.ConnectionInterceptorName=com.ibm.ISecurityLocalObjectBaseL13Impl.SecurityConnectionInterceptor
com.ibm.ws.orb.transport.WSSSLClientSocketFactoryName=com.ibm.ws.security.orbssl.WSSSLClientSocketFactoryImpl

com.ibm.CORBA.TransportMode=Conectável
com.ibm.CORBA.ServerName=ogserver
```

Configuração de Parâmetros SSL para Clientes do eXtreme Scale

É possível configurar os parâmetros SSL para os clientes da seguinte forma:

1. Crie um objeto com.ibm.websphere.objectgrid.security.config.SSLConfiguration usando o com.ibm.websphere.objectgrid.security.config. Classe de factory ClientSecurityConfigurationFactory. Para obter mais detalhes, consulte a Documentação da API do ClientSecurityConfigurationFactory.
2. Configure os parâmetros no arquivo `client.propeties` e, em seguida, use o método ClientSecurityConfigurationFactory.getClientSecurityConfiguration(String) para preencher a instância do objeto.

Consulte a seção nas propriedades do cliente de segurança no "Arquivo de Propriedades do Cliente" na página 202 para obter exemplos de propriedades que podem ser configuradas em um cliente.

Configuração de Parâmetros SSL para Servidores do eXtreme Scale

Os parâmetros SSL são configurados para servidores usando um arquivo de propriedades do servidor, como por exemplo o arquivo `server.properties` referido acima. Este arquivo de propriedades pode ser transmitido como um parâmetro ao iniciar um servidor do eXtreme Scale. Para obter informações adicionais sobre os parâmetros SSL, é possível configurar servidores do eXtreme Scale, consulte "Arquivo de Propriedades do Servidor" na página 183.

Suporte a Segurança de Transporte no WebSphere Application Server

Quando um cliente, servidor de contêiner ou servidor de catálogos do eXtreme Scale está em execução em um processo do WebSphere Application Server, a segurança do transporte do eXtreme Scale é gerenciada pelas configurações de transporte do Application Server CSIV2. Para o cliente ou servidor de contêiner do eXtreme Scale, você não deve utilizar as propriedades de servidor ou cliente do eXtreme Scale para configurar definições de SSL. Todas as configurações de SSL devem ser especificadas na configuração do WebSphere Application Server.

Entretanto, o servidor de catálogos é um pouco diferente. O servidor de catálogos tem seus próprios caminhos de transporte proprietários que não podem ser gerenciados pelas configurações de transporte do Application Server CSIV2. Portanto, as propriedades de SSL ainda precisam ser configuradas no arquivo de propriedades do servidor para o servidor de catálogos.

Ativar Segurança de Transporte para Sun JDK

O WebSphere eXtreme Scale requer um IBM Java Secure Sockets Extension (IBMJSSE) ou IBM Java Secure Sockets Extension 2 (IBMJSSE2). Os provedores IBMJSSE e IBMJSSE2 contêm uma implementação de referência que suporta os protocolos SSL e TLS (Transport Layer Security) e uma estrutura de API (Application Programming Interface).

O Sun JDK puro não fornece os provedores IBM JSSE and IBM JSSE2, portanto, a segurança do transporte não pode ser ativada com um Sun JDK. Para realizar esse

trabalho, um Sun JDK fornecido com WebSphere Application Server é necessário. O Sun JDK fornecido pelo WebSphere Application Server contém os provedores IBM JSSE e IBM JSSE2.

Leia configurando um Object Request Broker para poder utilizar um não IBM JDK para WebSphere eXtreme Scale. Se `-Djava.endorsed.dirs` estiver configurado, ele apontará para os diretórios `objectgridRoot/lib/endorsed` e `JRE/lib/endorsed`. O diretório `objectgridRoot/lib/endorsed` é necessário de modo que o IBM ORB seja utilizado e o diretório `JRE/lib/endorsed` é necessário para carregar os provedores IBM JSSE e IBM JSSE2.

Trabalhe com a etapa 4 do tutorial de segurança em *Visão Geral do Produto* para obter informações sobre como configurar suas propriedades de SSL necessárias, para criar keystores e truststores e para iniciar servidores seguros no WebSphere eXtreme Scale.

Segurança do Java Management Extensions (JMX)

É possível proteger chamadas de beans gerenciados (MBean) em um ambiente distribuído.

Para obter mais informações sobre MBeans disponíveis, consulte "Administrando Programaticamente com Beans Gerenciados (MBeans)" na página 353.

Na topologia de implementação distribuída, MBeans são hospedados diretamente nos servidores de catálogos e nos servidores de contêiner. Em geral, a segurança de JMX em uma topologia distribuída segue a especificação de segurança de JMX conforme especificado na Especificação JavaTM Management Extensions (JMX). Ela consiste nas três partes a seguir:

1. Autenticação - O cliente remoto precisa ser autenticado no servidor do conector.
2. Controle de acesso - O controle de acesso de MBean limita quem pode acessar as informações de MBean e quem pode executar as operações de MBean.
3. Transporte seguro - O transporte entre o cliente JMX e o servidor pode ser protegido utilizando TLS/SSL.

Autenticação

O JMX fornece métodos para os servidores conectores para autenticar os clientes remotos. Para o conector RMI, a autenticação é concluída fornecendo um objeto que implementa a interface JMXAuthenticator quando o servidor conector é criado. Assim, o eXtreme Scale implementa esta interface JMXAuthenticator para utilizar o plug-in do ObjectGrid Authenticator para autenticar os clientes remotos. Consulte o tutorial de segurança no *Visão Geral do Produto* para obter detalhes sobre como o eXtreme Scale autentica um cliente.

O cliente JMX segue as APIs do JMX para fornecer credenciais para conectar-se com o servidor conector. A estrutura JMX passa a credencial para o servidor conector e, então, chama a implementação do JMXAuthenticator para autenticação. Conforme descrito anteriormente, a implementação do JMXAuthenticator então delega a autenticação para a implementação do Autenticador doObjectGrid.

Revise o exemplo a seguir que descreve como conectar-se a um servidor conector com uma credencial:

```
javax.management.remote.JMXServiceURL jmxUrl = new JMXServiceURL(
        "service:jmx:rmi:///jndi/rmi://localhost:1099/objectgrid/MBeanServer");
```

```
    environment.put(JMXConnector.CREDENTIALS, new UserPasswordCredential("admin", "xxxxxx"));

    // Create the JMXCconnectorServer
    JMXConnector cntor = JMXConnectorFactory.newJMXConnector(jmxUrl, null);

    // Connect and invoke an operation on the remote MBeanServer
    cntor.connect(environment);
```

No exemplo anterior, um UserPasswordCredential é fornecido com o ID do usuário configurado como admin e a senha configurada como xxxxx. Este objeto UserPasswordCredential é configurado no mapa do ambiente, que é utilizado no método JMXConnector.connect(Map). Este objeto UserPasswordCredential é então passado para o servidor pela estrutura JMX, e finalmente passado para a estrutura de autenticação do ObjectGrid para a autenticação.

O modelo de programação do cliente segue estritamente a especificação JMX.

Controle de Acesso

Um servidor MBean JMX pode ter acesso a informações sensitivas e pode executar operações sensitivas. O JMX fornece o controle de acesso necessário que identifica quais clientes podem acessar tais informações e quem pode executar tais operações. o controle de acesso é integrado no modelo de segurança Java padrão por meio da definição de permissões que controlam o acesso ao servidor MBean e às suas operações.

Para controle de acesso de operação ou autorização do JMX, oeXtreme Scale conta com o suporte de JAAS fornecido pela implementação JMX. Em qualquer ponto determinado na execução de um programa, há um conjunto atual de permissões que um encadeamento de execuções contém. Quando um destes encadeamentos chama uma operação de especificação JMX, estes são conhecidos como as permissões contida. Quando uma operação JMX é executada, uma verificação de segurança é feita para verificar se a permissão necessária é incluída pela permissão contida.

A definição de política do MBean segue o formato da política Java. Por exemplo, a seguinte política concede a todos os assinantes e todas as bases de código o direito de recuperar o endereço JMX do servidor para o PlacementServiceMBean, mas com restrição ao domínio com.ibm.websphere.objectgrid.

```
grant {
    permission javax.management.MBeanPermission
        "com.ibm.websphere.objectgrid.management.PlacementServiceMBean#retrieveServerJMXAddress
[com.ibm.websphere.objectgrid:*,type=PlacementService]",
        "invoke";
}
```

O exemplo de política a seguir pode ser utilizado para concluir a autorização baseada na identidade do cliente remoto. A política concede a mesma permissão MBean, conforme mostrado no exemplo anterior, exceto apenas para usuários com o nome X500Principal como CN=Administrator,OU=software,O=IBM,L=Rochester,ST=MN,C=US.

```
grant principal javax.security.auth.x500.X500Principal
"CN=Administrator,OU=software,O=IBM,
    L=Rochester,ST=MN,C=US" {permission
javax.management.MBeanPermission
        "com.ibm.websphere.objectgrid.management.PlacementServiceMBean#retrieveServerJMXAddress
[com.ibm.websphere.objectgrid:*,type=PlacementService]",
        "invoke";
}
```

As políticas do Java são verificadas somente se o gerenciador de segurança estiver ativado. Inicie os servidores de catálogos e servidores de contêineres com o argumento JVM -Djava.security.manager para forçar o controle de acesso da operação MBean.

Transporte Seguro

O transporte entre o cliente e o servidor JMX pode ser protegido utilizando TLS/SSL. Se o transportType do servidor de catálogos ou servidor de contêineres for configurado como SSL_Required ou SSL_Supported, então, é necessário utilizar o SSL para conectar-se ao servidor JMX.

Para utilizar SSL, é necessário configurar o trust store, tipo de trust store e a senha trust store no cliente MBean utilizando propriedades do sistema -D:

1. -Djavax.net.ssl.trustStore=TRUST_STORE_LOCATION
2. -Djavax.net.ssl.trustStorePassword=TRUST_STORE_PASSWORD
3. -Djavax.net.ssl.trustStoreType=TRUST_STORE_TYPE

Se você utilizar com.ibm.websphere.ssl.protocol.SSLSocketFactory como seu socket factory SSL em seu arquivo JAVA_HOME/jre/lib/security/java.security, então, utilize as seguintes propriedades:

1. -Dcom.ibm.ssl.trustStore=TRUST_STORE_LOCATION
2. -Dcom.ibm.ssl.trustStorePassword=TRUST_STORE_PASSWORD
3. -Dcom.ibm.ssl.trustStoreType=TRUST_STORE_TYPE

Integração de Segurança com Provedores Externos

Para proteger seus dados do WebSphere eXtreme Scale, o eXtreme Scale pode se integrar com vários provedores de segurança.

O WebSphere eXtreme Scale pode integrar-se a uma implementação de segurança externa. Esta implementação externa deve fornecer serviços de autenticação e autorização para o eXtreme Scale. O eXtreme Scale possui pontos de plug-in para integrar-se com uma implementação de segurança. WebSphere eXtreme Scale foi integrado com sucesso com os seguintes componentes:

- LDAP (Lightweight Directory Access Protocol)
- Kerberos
- Segurança do ObjectGrid
- Tivoli Access Manager
- JAAS (Java Authentication and Authorization Service)

O eXtreme Scale utiliza o provedor de segurança para as seguintes tarefas:

- Autenticar clientes para servidores.
- Autorizar clientes para acessar determinados artefatos do eXtreme Scale ou para especificar o que pode ser feito com os artefatos do eXtreme Scale.

O eXtreme Scale possui os seguintes tipos de autorizações:

Autorização de mapa
Clientes ou grupos podem ser autorizados a executar operações de inserção, leitura, atualização, despejo ou exclusão nos mapas.

Autorização do ObjectGrid
Os clientes ou grupos podem ser autorizados a executar consultas de objetos ou entidades nas grades de objetos.

Autorização do agente do DataGrid
Clientes ou grupos podem ser autorizados a permitir que os agentes DataGrid sejam implementados a um ObjectGrid.

Autorização de Mapa do Lado do Servidor
Clientes ou grupos podem ser autorizados a replicar um mapa do servidor no lado do cliente ou criar um índice dinâmico para o mapa do servidor.

Autorização de administração
Clientes ou grupos podem ser autorizados a executar tarefas de administração.

Nota: Se você já tiver a segurança ativada para seu backend, lembre-se de que estas configurações de segurança não são mais suficientes para proteger seus dados. As configurações de segurança do seu banco de dados ou outro datastore não são, de forma alguma, transferidas para o seu cache. É necessário proteger separadamente os dados que agora são armazenados em cache utilizando o mecanismo de segurança do eXtreme Scale, incluindo segurança no nível de autenticação, autorização e transporte.

Restrição: Não use JDK/jre 1.6 e superior ao usar a segurança SSL Transport Layer com o WeSbphere eXtreme Scale 7.1 (ou 7.0) independente. O JDK/jre 1.6 e versões superiores não suportam as interfaces de programação de aplicativos WXS 7.1. Use o JDK/jre 1.5 ou inferior para configurações que requerem a segurança SSL Transport para instalações do eXtreme Scale independentes. Isso é aplicável apenas ao usar a segurança SSL em configurações do eXtreme Scale independentes. O JDK/jre é suportado para configurações de transporte não SSL.

Integração de Segurança com o WebSphere Application Server

O WebSphere eXtreme Scale fornece diversos recursos de segurança para integrar com a infraestrutura de segurança do WebSphere Application Server.

Integração de Autenticação

Quando os clientes e servidores do eXtreme Scale estão em execução no WebSphere Application Server e no mesmo domínio de segurança, é possível usar a infraestrutura de segurança do WebSphere Application Server para propagar as credenciais de autenticação do cliente para o servidor do eXtreme Scale. Por exemplo, se um servlet atuar como um cliente do eXtreme Scale para se conectar a um servidor do eXtreme Scale no mesmo domínio de segurança, e o servlet já estiver autenticado, é possível propagar o token de autenticação do cliente (servlet) para o servidor e, em seguida, usar a infraestrutura de segurança do WebSphere Application Server para converter o token de autenticação de volta para as credenciais do cliente.

Integração de segurança distribuída com o WebSphere Application Server

Para o modelo de ObjectGrid distribuído, a integração de segurança pode ser feita utilizando as seguintes classes:

```
com.ibm.websphere.objectgrid.security.plugins.builtins.WSTokenCredentialGenerator
com.ibm.websphere.objectgrid.security.plugins.builtins.WSTokenAuthenticator
```

```
com.ibm.websphere.objectgrid.security.
plugins.builtins.WSTokenCredential
```

Consulte o "Autenticação de Cliente do Aplicativo" na página 358 para obter informações adicionais. O exemplo a seguir ilustra como utilizar a classe WSTokenCredentialGenerator:

```
/**
 * connect to the ObjectGrid Server.
 */
protected ClientClusterContext connect() throws ConnectException {
  ClientSecurityConfiguration csConfig = ClientSecurityConfigurationFactory
  .getClientSecurityConfiguration(proFile);

  CredentialGenerator gen = getWSCredGen();

  csConfig.setCredentialGenerator(gen);

  return objectGridManager.connect(csConfig, null);
}

/**
 * Get a WSTokenCredentialGenerator
 *
private CredentialGenerator getWSCredGen() {
    WSTokenCredentialGenerator gen = new WSTokenCredentialGenerator(
        WSTokenCredentialGenerator.RUN_AS_SUBJECT);
    return gen;
}
```

No lado do servidor, o WSTokenAuthentication pode ser utilizado como o autenticador para autenticar o objeto WSTokenCredential.

Integração de segurança local com o WebSphere Application Server

Para o modelo de ObjectGrid local, a integração de segurança pode ser feita utilizando as duas classes a seguir:

- com.ibm.websphere.objectgrid.security.plugins.builtins.WSSubjectSourceImpl
- com.ibm.websphere.objectgrid.security.plugins.builtins.WSSubjectValidationImpl

Para obter informações adicionais sobre estas classes, consulte as informações sobre a segurança local no *Guia de Programação*. É possível configurar a classe WSSubjectSourceImpl como o plug-in SubjectSource e a classe WSSubjectValidationImpl como o plug-in SubjectValidation.

Início e Encerramento de Servidores Seguros do eXtreme Scale

Muitas vezes os servidores precisam ser seguros para seu ambiente de implementação, o que requer configuração específica para iniciar e parar.

Inicialização de um Servidor Seguro em um Ambiente Java SE

É possível iniciar um serviço de catálogo ou servidores de contêineres da forma a seguir.

Iniciando um serviço de catálogo seguro do eXtreme Scale

A inicialização de um processo do serviço de catálogo seguro do eXtreme Scale exige mais dois arquivos de configuração de segurança:

Arquivo XML descritor de segurança: O arquivo XML descritor de segurança descreve as propriedades de segurança comuns para todos os servidores (incluindo servidores de catálogo e servidores de contêiner). Um exemplo de propriedade é a configuração do autenticador que representa o registro do usuário e o mecanismo de autenticação.

Arquivo de propriedades do servidor. O arquivo de propriedades do servidor configura as propriedades específicas de segurança para o servidor.

Quando você usa o comando startOgServer.sh ou startOgServer.cat para iniciar um processo seguro do serviço de catálogo do eXtreme Scale, é possível usar o -clusterSecurityFile ou -clusterSecurityUrl para configurar seguramente o arquivo XML do descritor de segurança como um tipo file ou URL, e pode usar –serverProps para configurar o arquivo de propriedades do servidor.

Início de um servidor de contêineres seguros do eXtreme Scale

A inicialização de um servidor de contêineres seguro do eXtreme Scale exige um arquivo de configuração de segurança:

- **Arquivo de propriedades do servidor:** O arquivo de propriedades do servidor configura as propriedades de segurança específicas para o servidor. Consulte o "Arquivo de Propriedades do Servidor" na página 183 para obter mais detalhes.

Quando você usa o comando startOgServer.sh ou startOgServer.cat para iniciar um servidor de contêineres seguro do eXtreme Scale, é possível usar –serverProps para configurar o arquivo de propriedades do servidor. Há mais formas para configurar o arquivo de propriedades de servidor, consulte o arquivo de propriedades do servidor para obter mais detalhes.

Para obter mais detalhes sobre como usar o comando startOgServer.sh ou startOgServer.bat e suas opções, consulte "Script startOgServer" na página 334.

Encerramento de um Servidor Seguro do eXtreme Scale

O encerramento de um processo seguro do serviço de catálogo do eXtreme Scale ou servidor de contêineres exige um arquivo de configuração de segurança:

- **Arquivo de propriedades do cliente:** O arquivo de propriedades do cliente pode ser usado para configurar as propriedades de segurança do cliente. As propriedades de segurança do cliente são necessárias para um cliente se conectar a um servidor seguro. Consulte o "Arquivo de Propriedades do Cliente" na página 202 para obter mais detalhes.

Quando você usa o comando stopOgServer.sh ou stopOgServer.cat para encerrar um processo seguro do serviço de catálogo do eXtreme Scale ou servidor de contêineres, é possível usar -clientSecurityFile para configurar as propriedades de segurança do cliente.

Para obter mais detalhes sobre como usar o comando stopOgServer.sh ou stopOgServer.cat e suas opções, consulte "Script stopOgServer" na página 338.

Iniciando um Servidor Seguro no WebSphere Application Server

A inicialização de um servidor seguro ObjectGrid em WebSphere Application Server é semelhante à inicialização de um servidor não seguro ObjectGrids, exceto que é necessário passar os arquivos de configuração de segurança. Em vez de usar o -[PROPERTY_FILE] (por exemplo -serverProps) no comando como no ambiente Java SE, use -D[PROPERTY_FILE] nos argumentos genéricos da Java Virtual Machine (JVM).

Iniciando um serviço de catálogo seguro no Websphere Application Server

Um servidor de catálogos contém dois níveis diferente de informações de segurança:

- `-Dobjectgrid.cluster.security.xml.url`: Especifica o arquivo objectGridSecurity.xml que descreve as propriedades de segurança comuns a todos os servidores (incluindo servidores de catálogos e servidores de contêineres). Um exemplo é a configuração do autenticador que representa o mecanismo de registro e autenticação do usuário. O nome do arquivo especificado para esta propriedade deve estar em um formato de URL, como "file:///tmp/og/objectGridSecurity.xml".
- `-Dobjectgrid.server.props`: Especifica o arquivo de propriedades do servidor que contém as propriedades de segurança específicas do servidor. O nome do arquivo especificado para esta propriedade está apenas no formato simples de caminho de arquivo, como "c:/tmp/og/catalogserver.props". Observe que o uso de -Dobjectgrid.security.server.props está reprovado, mas ele pode continuar sendo usado para compatibilidade com versões anteriores.

Para iniciar um serviço de catálogo seguro em WebSphere Application Server, siga o "Incorporado no WebSphere Application Server" no “Segurança de Grade” na página 356.

Em seguida, a propriedade de segurança no argumento da JVM genérico do processo.

```
-Dobjectgrid.cluster.security.xml.url=file:///tmp/og/
objectGridSecurity.xml-Dobjectgrid.server.props=/tmp/og/
catalog.server.props
```

As etapas para incluir os argumentos da JVM genéricos são as seguintes:

- Expanda "Administração do sistema" na visualização de tarefa do lado esquerdo.
- Clique no processo WebSphere Application Server no qual o serviço de catálogo está implementado como, por exemplo, "Gerenciador de Implementação".
- Na página à direita, expanda "Java e Gerenciamento de Processo" sob "Infraestrutura do Servidor".
- Clique em "Definição de Processo".
- Clique em "Java Virtual Machine" sob "Propriedades Adicionais".
- Digite as propriedades na caixa de texto Argumentos JVM Genéricos.

Iniciando um servidor de contêineres seguro no WebSphere Application Server

Um servidor de contêineres, ao se conectar ao servidor de catálogos, obterá todas as configurações de segurança definidas no objectGridSecurity.xml, como a configuração do autenticador ou do tempo limite da sessão de login. Além disso, um servidor de contêineres precisa configurar suas próprias propriedades de segurança específicas do servidor na propriedade -Dobjectgrid.server.props.

É necessário usar a propriedade -Dobjectgrid.server.props em vez da propriedade -Dobjectgrid.security.server.props porque também colocamos outras propriedades não relacionadas à segurança neste arquivo de propriedades. O nome do arquivo especificado para esta propriedade está apenas no formato de caminho de arquivo simples, como `c:/tmp/og/server.props`.

Siga as mesmas etapas acima para incluir a propriedade de segurança para os argumentos genéricos da JVM.

Arquivo XML Descritor de Segurança

Use um arquivo descritor de segurança do ObjectGrid para configurar uma topologia de implementação do eXtreme Scale com segurança ativada. Os arquivos XML de amostra a seguir descrevem várias configurações.

Cada elemento e atributo do arquivo XML do cluster está descrito na lista a seguir. Utilize os exemplos para aprender como utilizar esses elementos e atributos para configurar o ambiente.

Elemento securityConfig

O elemento securityConfig é o elemento de nível superior do arquivo XML de segurança do ObjectGrid. Este elemento configura o espaço de nomes do arquivo e o local do esquema. O esquema está definido no arquivo `objectGridSecurity.xsd`.

- Número de ocorrências: Uma
- Elementos filho : segurança

Elemento security

Utilize o elemento security para definir uma segurança do ObjectGrid.

- Número de ocorrências: Uma
- Elementos-filho: authenticator, adminAuthorization e systemCredentialGenerator

Atributos

securityEnabled
Ativa a segurança para a grade quando configurado para true. O valor padrão é false Se o valor for configurado como false, a segurança no escopo da grade será desativada. Para obter informações adicionais, consulte "Segurança de Grade" na página 356. (Opcional)

singleSignOnEnabled
Permite que o cliente se conecte a qualquer servidor depois que ele for autenticado com um dos servidores, se o valor for configurado para true. Caso contrário, um cliente deverá se autenticar com cada servidor antes de poder se conectar. O valor padrão é false. (Opcional)

loginSessionExpirationTime
Especifica a quantidade de tempo, em segundos, antes de a sessão de login expirar. Se a sessão de login expirar, o cliente precisa autenticar-se novamente. (Opcional)

adminAuthorizationEnabled
Ativa a autorização administrativa. Se o valor for configurado para true, todas as tarefas administrativas precisarão de autorização. O mecanismo de autorização utilizado é especificado pelo valor do atributo adminAuthorizationMechanism. O valor padrão é false. (Opcional)

adminAuthorizationMechanism
Indica qual mecanismo de autorização deve ser usado. O WebSphere eXtreme Scale suporta dois mecanismos de autorização, JAAS (Java Authentication and Authorization Service) e autorização personalizada. O mecanismo de autorização JAAS utiliza a abordagem baseada em política JAAS padrão. Para especificar JAAS como o mecanismo de autorização, configure o valor como AUTHORIZATION_MECHANISM_JAAS. O mecanismo de autorização customizado utiliza uma implementação de AdminAuthorization conectada

pelo usuário. Para especificar um mecanismo de autorização customizado, configure o valor como AUTHORIZATION_MECHANISM_CUSTOM. Para obter informações adicionais sobre como estes dois mecanismos são utilizados, consulte “Autorização do Aplicativo Cliente” na página 360. (Opcional)

O arquivo `security.xml` a seguir é uma configurações de amostra para ativar a segurança da grade do eXtreme Scale.

security.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<securityConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://ibm.com/ws/objectgrid/config/security ../objectGridSecurity.xsd"
 xmlns="http://ibm.com/ws/objectgrid/config/security">

    <security securityEnabled="true" singleSignOnEnabled="true"
        loginSessionExpirationTime="20"
        adminAuthorizationEnabled="true"
        adminAuthorizationMechanism="AUTHORIZATION_MECHANISM_JAAS" >

        <authenticator className ="com.ibm.websphere.objectgrid.security.plugins.builtins.WSTokenAuthenticator">
        </authenticator>

        <systemCredentialGenerator className ="com.ibm.websphere.objectgrid.security.plugins.builtins.WSTokenCredential(
            <property name="properties" type="java.lang.String" value="runAs" description="Using runAs subject" />
        </systemCredentialGenerator>

    </security>
</securityConfig>
```

Elemento authenticator

Autentica clientes para servidores eXtreme Scale na grade. A classe especificada pelo atributo className deve implementar a interface com.ibm.websphere.objectgrid.security.plugins.Authenticator. O autenticador pode utilizar propriedades para chamar métodos na classe que é especificada pelo atributo className. Consulte o elemento de propriedade para obter mais informações sobre como utilizar as propriedades.

No exemplo do arquivo `security.xml` anterior, a classe com.ibm.websphere.objectgrid.security.plugins.builtins.WSTokenAuthenticator é especificada como o autenticador. Esta classe implementa a interface com.ibm.websphere.objectgrid.security.plugins.Authenticator.

- Número de ocorrências: zero ou uma
- Elemento filho: propriedade

Atributos

className
Especifica a classe que implementa a interface com.ibm.websphere.objectgrid.security.plugins.Authenticator. Use esta classe para autenticar clientes para os servidores na grade do eXtreme Scale. (Obrigatório)

Elemento adminAuthorization

Use o elemento adminAuthorization para configurar o acesso administrativo para a grade.

- Número de ocorrências: zero ou uma
- Elemento filho: propriedade

Atributos

className
Especifica a classe que implementa a interface com.ibm.websphere.objectgrid.security.plugins.AdminAuthorization. (Obrigatório)

Elemento systemCredentialGenerator

Utilize um elemento systemCredentialGenerator para configurar um gerador de credenciais do sistema. Este elemento é aplicável apenas a um ambiente dinâmico. No modelo de configuração dinâmica, o servidor de contêineres dinâmicos se conecta ao servidor de catálogos como um cliente do eXtreme Scale e o servidor de catálogos pode se conectar ao servidor de contêineres do eXtreme Scale como um cliente também. O gerador de credenciais do sistema é utilizado para representar um depósito de informações para a credencial do sistema.

- Número de ocorrências: zero ou uma
- Elemento filho: propriedade

Atributos

className
Especifica a classe que implementa a interface com.ibm.websphere.objectgrid.security.plugins.CredentialGenerator. (Obrigatório)

Consulte o arquivo `security.xml` anterior para saber como utilizar um systemCredentialGenerator. Neste exemplo, o gerador de credenciais do sistema é um com.ibm.websphere.objectgrid.security.plugins.builtins. WSTokenCredentialGenerator, que recupera o objeto RunAs Subject a partir do encadeamento.

Elemento property

Chama o método set nas classes authenticator e adminAuthorization. O nome da propriedade corresponde a um método configurado no atributo className do elemento authenticator ou adminAuthorization.

- Número de ocorrências: zero ou mais
- Elemento filho: propriedade

Atributos

name
Especifica o nome da propriedade. O valor atribuído a esse atributo deve corresponder a um método set na classe que é fornecida como atributo className no bean que contém o atributo. Por exemplo, se o atributo className do bean for configurado como com.ibm.MyPlugin e o nome da propriedade que é fornecido for size, então, a classe com.ibm.MyPlugin deve ter um método setSize. (Obrigatório)

type
Especifica o tipo da propriedade. O tipo do parâmetro é transmitido ao método set identificado pelo atributo name. Os valores válidos são as primitivas Java, suas partes java.lang e java.lang.String. Os atributos name e type devem corresponder a uma assinatura de método no atributo className do bean. Por exemplo, se o nome for size e o tipo for int, então, deve existir um método setSize(int) na classe que é especificada como o atributo className para o bean. (Obrigatório)

value
Especifica o valor da propriedade. Este valor é convertido no tipo especificado pelo atributo type e, em seguida, é utilizado como um parâmetro na chamada para o método set identificado pelos atributos name e type. O valor deste atributo não é validado de nenhuma maneira. O implementador do plug-in deve verificar se o valor transmitido é válido. (Obrigatório)

description
Fornece uma descrição da propriedade (Opcional)

Consulte "Arquivo `objectGridSecurity.xsd`" para obter mais informações.

Arquivo `objectGridSecurity.xsd`

Use o seguinte esquema XML de segurança do ObjectGrid para ativar a segurança para uma implementação do eXtreme Scale.

Consulte o "Arquivo XML Descritor de Segurança" na página 373 para obter descrições dos elementos e atributos definidos no arquivo `objectGridSecurity.xsd`.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:cc="http://ibm.com/ws/objectgrid/config/security"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 targetNamespace="http://ibm.com/ws/objectgrid/config/security"
 elementFormDefault="qualified">

 <xsd:element name="securityConfig">
  <xsd:complexType>
   <xsd:sequence>
    <xsd:element name="security" type="cc:security" />
   </xsd:sequence>
  </xsd:complexType>
 </xsd:element>

 <xsd:complexType name="security">
  <xsd:sequence>
   <xsd:element name="authenticator" type="cc:bean" minOccurs="0"
    maxOccurs="1" />
   <xsd:element name="adminAuthorization" type="cc:bean" minOccurs="0"
    maxOccurs="1" />
   <xsd:element name="systemCredentialGenerator" type="cc:bean" minOccurs="0"
    maxOccurs="1" />
  </xsd:sequence>
  <xsd:attribute name="securityEnabled" type="xsd:boolean" use="optional" />
  <xsd:attribute name="singleSignOnEnabled" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="loginSessionExpirationTime" type="xsd:int" use="optional"/>
  <xsd:attribute name="adminAuthorizationMechanism"
type="cc:adminAuthorizationMechanism"
   use="optional" />
  <xsd:attribute name="adminAuthorizationEnabled" type="xsd:boolean" use="optional" />
 </xsd:complexType>

 <xsd:complexType name="bean">
  <xsd:sequence>
   <xsd:element name="property" type="cc:property" maxOccurs="unbounded" minOccurs="0" />
  </xsd:sequence>
  <xsd:attribute name="className" type="xsd:string" use="required" />
 </xsd:complexType>

 <xsd:complexType name="property">
  <xsd:attribute name="name" type="xsd:string" use="required" />
  <xsd:attribute name="value" type="xsd:string" use="required" />
  <xsd:attribute name="type" type="cc:propertyType" use="required" />
  <xsd:attribute name="description" type="xsd:string" use="optional" />
 </xsd:complexType>

 <xsd:simpleType name="propertyType">
  <xsd:restriction base="xsd:string">
   <xsd:enumeration value="java.lang.Boolean"/>
   <xsd:enumeration value="boolean"/>
   <xsd:enumeration value="java.lang.String"/>
   <xsd:enumeration value="java.lang.Integer"/>
```

```
   <xsd:enumeration value="int"/>
   <xsd:enumeration value="java.lang.Double"/>
   <xsd:enumeration value="double"/>
   <xsd:enumeration value="java.lang.Byte"/>
   <xsd:enumeration value="byte"/>
   <xsd:enumeration value="java.lang.Short"/>
   <xsd:enumeration value="short"/>
   <xsd:enumeration value="java.lang.Long"/>
   <xsd:enumeration value="long"/>
   <xsd:enumeration value="java.lang.Float"/>
   <xsd:enumeration value="float"/>
   <xsd:enumeration value="java.lang.Character"/>
   <xsd:enumeration value="char"/>
  </xsd:restriction>
 </xsd:simpleType>

 <xsd:simpleType name="adminAuthorizationMechanism">
  <xsd:restriction base="xsd:string">
   <xsd:enumeration value="AUTHORIZATION_MECHANISM_JAAS" />
   <xsd:enumeration value="AUTHORIZATION_MECHANISM_CUSTOM" />
  </xsd:restriction>
 </xsd:simpleType>

</xsd:schema>
```

Capítulo 9. Monitorando o Ambiente de Implementação

É possível utilizar APIs, MBeans, logs e utilitários para monitorar o desempenho do seu ambiente de aplicativo.

Visão Geral de Estatísticas

As estatísticas no WebSphere eXtreme Scale são construídas a partir de uma árvore de estatísticas internas. A API StatsAccessor, módulos da Performance Monitoring Infrastructure (PMI) e a API MBean são construídos a partir da árvore interna.

A seguinte figura mostra a configuração geral das estatísticas para o eXtreme Scale.

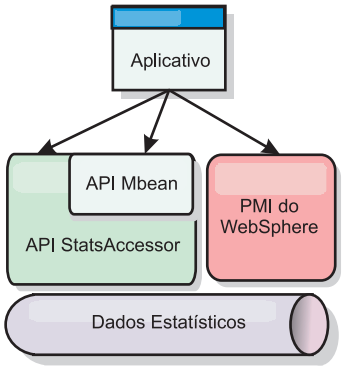


Figura 26. Visão Geral de Estatísticas

Cada uma das APIs oferece uma visualização da árvore de estatísticas, mas são utilizadas por diferentes motivos:

- **API de Estatísticas**: A API de Estatísticas permite que os desenvolvedores tenham acesso direto às estatísticas, o que permite soluções de integração de estatísticas flexíveis e customizáveis, como MBeans customizados ou a criação de log.
- **API do MBean**: A API do MBean é um mecanismo baseado em especificação para monitoramento. A API MBean usa a API Statistics e executa localmente para o servidor do JVM (Java Virtual Machine). As estruturas de API e MBean são projetadas para integração imediata com utilitários de outros fornecedores. Use a API MBean quando estiver executando uma grade de objeto distribuído.
- **WebSphere Application Server Módulos Performance Monitoring Infrastructure (PMI)** : Utilize a PMI se estiver executando o WebSphere eXtreme Scale no WebSphere Application Server. Estes módulos fornecem uma visualização da árvore de estatísticas internas.

API de Estatísticas

Muito semelhante a um mapa de árvore, há um caminho e uma chave correspondentes para recuperar um módulo específico, ou neste caso, nível de granularidade ou agregação. Por exemplo, assume que sempre há um nó-raiz arbitrário na árvore e que as estatísticas estão sendo reunidas para um mapa nomeado como "payroll", pertencendo a um ObjectGrid nomeado como "accounting". Por exemplo, para acessar o módulo para um nível de agregação ou

granularidade do mapa, você poderia passar uma String[] dos caminhos. Neste caso, isto seria igual a String[] {root, "accounting", "payroll"}, já que cada String representaria o caminho do nó. A vantagem desta estrutura é que um usuário pode especificar a matriz para qualquer nó no caminho e obter o nível de agregação para tal nó. Portando, passar String[] {root, "accounting"} forneceria a você as estatísticas do mapa, mas para a grade inteira de "accounting." Isto deixa o usuário tanto com a habilidade de especificar tipos de estatísticas a serem monitorados quanto em que nível de agregação é necessário para o aplicativo.

Módulos PMI do WebSphere Application Server

O WebSphere eXtreme Scale inclui módulos de estatísticas para uso com a PMI do WebSphere Application Server. Quando um perfil do WebSphere Application Server é aumentado com o WebSphere eXtreme Scale, os scripts de aumento integram-se automaticamente aos módulos do WebSphere eXtreme Scale nos arquivos de configuração do WebSphere Application Server. Com a PMI, é possível ativar e desativar módulos de estatísticas, automaticamente agregar estatísticas em várias granularidades e, até mesmo, criar gráficos dos dados utilizando o Tivoli Performance Viewer integrado. Consulte “Monitorando com a PMI do WebSphere Application Server” na página 388 para obter mais informações.

Integração de Produtos de Fornecedores com Beans Gerenciados (MBean)

As APIs do eXtreme Scale e os Beans Gerenciados são projetados para permitir a fácil integração com aplicativos de monitoramento de terceiros. JConsole ou MC4J são alguns exemplos de consoles Java Management Extensions (JMX) leves que podem ser utilizados para analisar informações sobre uma topologia do eXtreme Scale. Também é possível utilizar as APIs programáticas para criar implementações do adaptador para realizar uma captura instantânea ou controlar o desempenho do eXtreme Scale. O WebSphere eXtreme Scale inclui um aplicativo de monitoramento de amostra que permite recursos de monitoramento prontos para utilização e que pode ser utilizado como um modelo para criação de utilitário de monitoramento customizados mais avançados.

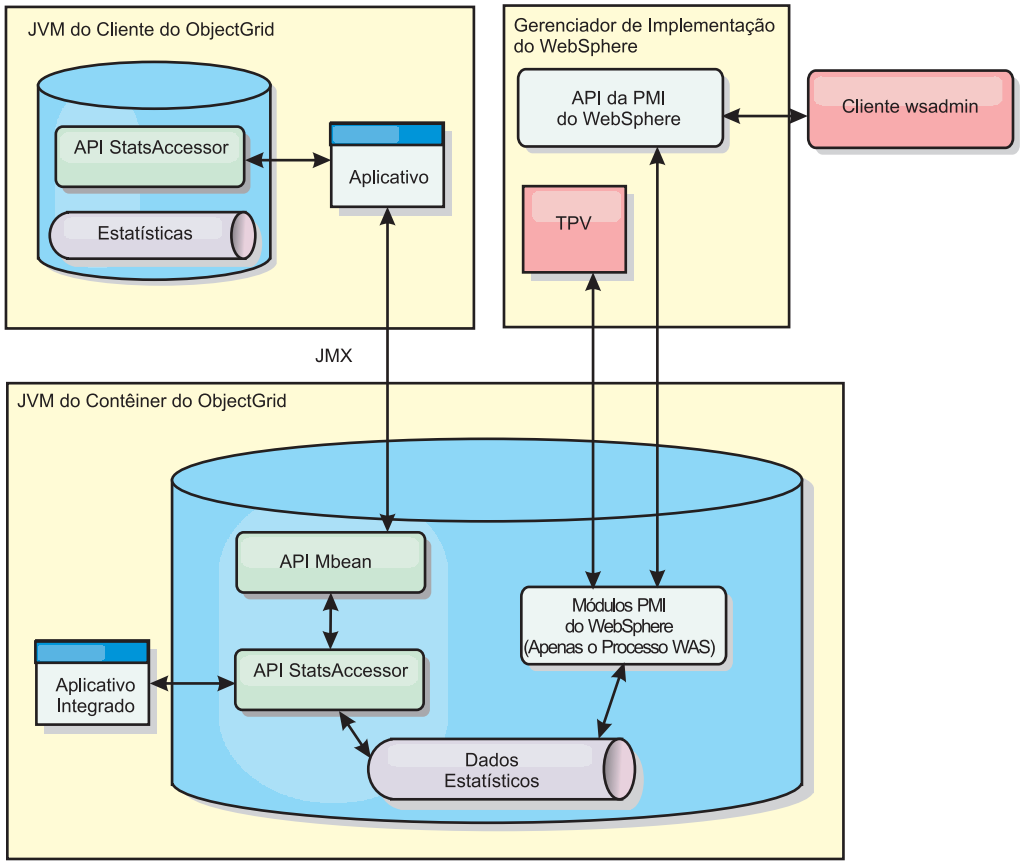


Figura 27. Visão Geral do MBean

Consulte "Monitorando com o Utilitário de Amostra xsAdmin" na página 385 para obter mais informações. Para obter informações adicionais sobre a integração com aplicativos de um fornecedor específico, consulte os tópicos a seguir:

- Monitoramento do eXtreme Scale com o IBM Tivoli Monitoring Agent
- "Monitorando o eXtreme Scale com o Hyperic HQ" na página 418
- "Monitorando Aplicativos eXtreme Scale com o CA Wily Introscope" na página 414

Monitorando com a API de Estatísticas

A API de Estatísticas é a interface direta para a árvore de estatísticas interna. As estatísticas são desativadas por padrão, mas podem ser ativadas configurando uma interface StatsSpec. Uma interface StatsSpec define como o WebSphere eXtreme Scale deve monitorar estatísticas.

Sobre Esta Tarefa

É possível usar a API StatsAccessor local para consultar dados e acessar estatísticas em qualquer instância do ObjectGrid que esteja no mesmo Java Virtual Machine (JVM) que o código de execução. Para obter mais informações sobre as interfaces específicas, consulte a documentação da API. Conclua as seguintes etapas para ativar o monitoramento da árvore de estatísticas internas.

Procedimento

1. Recupere o objeto StatsAccessor. A interface StatsAccessor segue o padrão do singleton. Assim, além dos problemas relacionados ao carregador de classes, uma instância do StatsAccessor deverá existir para cada JVM. Esta classe atua como a interface principal para todas as operações locais de estatísticas. O seguinte código é um exemplo de como recuperar a classe do acessador. Chame essa operação antes de qualquer outra chamada ObjectGrid.

```
public class LocalClient {

   public static void main(String[] args) {

      // retrieve a handle to the StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

   }

}
```

2. Configure a interface StatsSpec da grade. Configure esse JVM para coletar todas as estatísticas apenas no nível do ObjectGrid. É necessário garantir que um aplicativo ative todas as estatísticas que podem ser necessárias antes de iniciar quaisquer transações. O exemplo a seguir configura a interface StatsSpec utilizando um campo constante estático e utilizando uma spec String. Usar um campo de constante estático é mais simples porque o campo já definiu a especificação. Entretanto, ao utilizar uma spec String, é possível ativar qualquer combinação de estatísticas que são necessárias.

```
public static void main(String[] args) {

      // retrieve a handle to the StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

      // Set the spec via the static field
      StatsSpec spec = new StatsSpec(StatsSpec.OG_ALL);
      accessor.setStatsSpec(spec);

      // Set the spec via the spec String
      StatsSpec spec = new StatsSpec("og.all=enabled");
      accessor.setStatsSpec(spec);

}
```

3. Envie as transações para a grade para forçar a coleta dos dados para monitoramento. Para coletar dados úteis para as estatísticas, é necessário enviar as transações para a grade. O seguinte extrato de código insere um registro no MapA, que é um ObjectGridA. Como as estatísticas estão no nível do ObjectGrid, qualquer mapa dentro do ObjectGrid produz os mesmos resultados.

```
public static void main(String[] args) {

      // retrieve a handle to the StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

      // Set the spec via the static field
      StatsSpec spec = new StatsSpec(StatsSpec.OG_ALL);
      accessor.setStatsSpec(spec);

      ObjectGridManager manager =
    ObjectGridmanagerFactory.getObjectGridManager();
      ObjectGrid grid = manager.getObjectGrid("ObjectGridA");
      Session session = grid.getSession();
      Map map = session.getMap("MapA");

      // Drive insert
```

```
      session.begin();
      map.insert("SomeKey", "SomeValue");
      session.commit();
}
```

4. Consulte um StatsFact utilizando a API do StatsAccessor. Cada caminho de estatísticas é associado com uma interface StatsFact. A interface StatsFact é um marcador genérico que é utilizado para organizar e conter um objeto StatsModule. Antes de poder acessar o módulo de estatísticas real, o objeto StatsFact deve ser recuperado.

```
public static void main(String[] args)
{

      // retrieve a handle to the StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

      // Set the spec via the static field
      StatsSpec spec = new StatsSpec(StatsSpec.OG_ALL);
      accessor.setStatsSpec(spec);

      ObjectGridManager manager =
    ObjectGridManagerFactory.getObjectGridManager();
      ObjectGrid grid = manager.getObjectGrid("ObjectGridA");
      Session session = grid.getSession();
      Map map = session.getMap("MapA");

      // Drive insert
      session.begin();
      map.insert("SomeKey", "SomeValue");
      session.commit();

      // Retrieve StatsFact

      StatsFact fact = accessor.getStatsFact(new String[] {"EmployeeGrid"},
    StatsModule.MODULE_TYPE_OBJECT_GRID);

}
```

5. Interaja com o objeto StatsModule. O objeto StatsModule está contido na interface StatsFact. É possível obter uma referência para o módulo utilizando a interface StatsFact. Como a interface StatsFact é uma interface genérica, é necessário converter o módulo retornado para o tipo StatsModule esperado. Como esta tarefa coleta estatísticas do eXtreme Scale, o objeto StatsModule retornado é convertido para um tipo OGStatsModule. Após o módulo ser convertido, é possível acessar todas as estatísticas disponíveis.

```
public static void main(String[] args) {

      // retrieve a handle to the StatsAccessor
      StatsAccessor accessor = StatsAccessorFactory.getStatsAccessor();

      // Set the spec via the static field
      StatsSpec spec = new StatsSpec(StatsSpec.OG_ALL);
      accessor.setStatsSpec(spec);

      ObjectGridManager manager =
    ObjectGridmanagerFactory.getObjectGridManager();
      ObjectGrid grid = manager.getObjectGrid("ObjectGridA");
      Session session = grid.getSession();
      Map map = session.getMap("MapA");

      // Drive insert
      session.begin();
      map.insert("SomeKey", "SomeValue");
      session.commit();
```

```
        // Retrieve StatsFact
        StatsFact fact = accessor.getStatsFact(new String[] {"EmployeeGrid"},
      StatsModule.MODULE_TYPE_OBJECT_GRID);

        // Retrieve module and time
        OGStatsModule module = (OGStatsModule)fact.getStatsModule();
        ActiveTimeStatistic timeStat =
      module.getTransactionTime("Default", true);
        double time = timeStat.getMeanTime();

  }
```

Módulos Estatísticos

O WebSphere eXtreme Scale usa um modelo de estatísticas interno para controlar e filtrar dados, que é a estrutura subjacente usada por todas as visualizações de dados para reunir capturas instantâneas das estatísticas. É possível utilizar diversos métodos para recuperar as informações dos módulos estatísticos.

Visão Geral

As estatísticas no WebSphere eXtreme Scale são controladas e contidas nos componentes StatsModules. No modelo estatístico, existem diversos tipos de módulos estatísticos:

OGStatsModule
: Fornece estatísticas para uma instância do ObjectGrid, incluindo tempos de resposta da transação.

MapStatsModule
: Fornece estatísticas para um único mapa, incluindo o número de entradas e a taxa de ocorrências.

QueryStatsModule
: Fornece estatísticas para consultas, incluindo criação de planos e tempo de execução.

AgentStatsModule
: Fornece estatísticas para agentes de API do DataGrid API, incluindo tempos de de serialização e tempos de execução.

HashIndexStatsModule
: Fornece estatísticas para consulta HashIndex e tempos de execução de manutenção.

SessionStatsModule
: Fornece estatísticas ao plug-in do gerenciador de sessões HTTP.

Para obter detalhes dobre os módulos estatísticos, consulte o pacote com.ibm.websphere.objectgrid.stats na Documentação da API .

Estatísticas em um Ambiente Local

O modelo é organizado como uma árvore n-ária (uma estrutura em árvore com o mesmo grau para todos os nós) que é composta de todos os tipos de StatsModule mencionados na lista anterior. Devido a esta estrutura de organização, cada nó na árvore é representado pela interface StatsFact. A interface StatsFact pode representar um módulo individual ou um grupo de módulos para propósitos de agregação. Por exemplo, se vários nós-folhas na árvore representarem objetos MapStatsModule específicos, o nó-pai StatsFact para estes nós conterá estatísticas

agregadas para todos os módulos-filhos. Após buscar um objeto StatsFact, é possível utilizar a interface para recuperar o StatsModule correspondente.

Muito semelhante a um mapa de árvore, é utilizado um caminho ou chave correspondente para recuperar um StatsFact específico. O caminho é um valor String[] que consiste em cada nó que está ao long do caminho para o fato solicitado. Por exemplo, você criou um ObjectGrid denominado ObjectGridA, que contém dois Mapas: MapA e MapB. O caminho para o StatsModule para MapA seria semelhante a `[ObjectGridA, MapA]`. O caminho para as estatísticas agregadas para ambos os mapas seria: `[ObjectGridA]`.

Estatísticas em um Ambiente Distribuído

Em um ambiente distribuído, os módulos de estatísticas são recuperados utilizando um caminho diferente. Como um servidor pode conter várias partições, a árvore de estatísticas precisa controlar a partição a qual cada módulo pertence. Como resultado, o caminho para consultar um objeto StatsFact específico é diferente. Utilizando o exemplo anterior, mas incluindo nele os mapas existentes na partição 1, o caminho é `[1, ObjectGridA, MapA]` para recuperação de tal objeto StatsFact para MapA.

Monitorando com o Utilitário de Amostra xsAdmin

Com o utilitário de amostra xsAdmin, é possível formar e exibir informações contextuais sobre sua topologia do WebSphere eXtreme Scale. O utilitário de amostra fornece um método para analisar e descobrir dados de implementação atuais e pode ser utilizado com uma base para criação de utilitários customizados.

Antes de Iniciar

É necessário ter o WebSphere eXtreme Scale instalado.

Sobre Esta Tarefa

É possível utilizar o utilitário de amostra xsAdmin para fornecer feedback sobre o layout atual e o estado específico da grade, tal como o conteúdo de mapa. Neste exemplo, o layout da grade nesta tarefa consiste em uma grade única, denominada *ObjectGridA* com um mapa definido, denominado *MapA*, pertencendo ao mapset, intitulado *MapSetA*. Este exemplo demonstra como é possível exibir todos os contêineres ativos em uma grade e imprimir métricas filtradas relativa ao tamanho do mapa do *MapA*. Para visualizar todas as opções de comando possíveis, execute o utilitário xsAdmin sem nenhum argumento ou com a opção **-help**.

Procedimento

1. Na linha de comandos, configure a variável de ambiente JAVA_HOME.
   - UNIX `export JAVA_HOME=javaHome`
   - Windows `set JAVA_HOME=javaHome`
2. Navegue até o diretório bin.
   ```
   cd objectGridRoot/bin
   ```
3. Ative o utilitário xsAdmin.
   - **Para exibir a ajuda on-line, execute o comando a seguir:**

     UNIX

     ```
     xsadmin.sh
     ```

Windows

```
xsadmin.bat
```

Preste atenção à seção de argumentos necessários da mensagem de ajuda, porque é necessário passar apenas uma das opções listadas para o utilitário trabalhar. Se nenhuma opção **-g** ou **-m** for especificada, o utilitário xsAdmin imprime informações para cada grade na topologia.

- **Para ativar as estatísticas para todos os servidores, execute o seguinte comando:**

UNIX

```
xsadmin.sh -g ObjectGridA -setstatsspec ALL=enabled
```

Windows

```
xsadmin.bat -g ObjectGridA -setstatsspec ALL=enabled
```

- **Para exibir todos os contêineres on-line para uma grade, execute o seguinte comando:**

UNIX

```
xsadmin.sh -g ObjectGridA -m MapSetA -containers
```

Windows

```
xsadmin.bat -g ObjectGridA -m MapSetA -containers
```

Todas as informações do contêiner são exibidas. A seguir está um exemplo de saída:

```
This administrative utility is provided as a sample only and is not to be
considered a fully supported component of the WebSphere eXtreme Scale product

Connecting to Catalog service at localhost:1099


*** Show all online containers for grid - ObjectGridA & mapset - MapSetA

Host: 192.168.0.186
Container: server1_C-0, Server:server1, Zone:DefaultZone
Partition Shard Type
        0 Primary

Num containers matching = 1
Total known containers = 1
Total known hosts = 1
```

- **Para conectar ao serviço de catálogo e exibir informações sobre MapA, execute o seguinte comando:**

UNIX

```
xsadmin.sh -g ObjectGridA -m MapSetA -mapsizes -fm MapA
```

Windows

```
xsadmin.bat -g ObjectGridA -m MapSetA -mapsizes -fm MapA
```

O tamanho do mapa especificado é exibido. A seguir está um exemplo de saída:

```
This administrative utility is provided as a sample only and is not to be
considered a fully supported component of the WebSphere eXtreme Scale product

Connecting to Catalog service at localhost:1099

****Displaying Results for Grid - ObjectGridA, MapSet - MapSetA*****
```

```
*** Listing Maps for server1 ***
 Nome do Mapa Partição Tamanho do Mapa Bytes Usados (B) Tipo de Shard
 MapA     0         0        0              Primário
```

- **Para conectar ao serviço de catálogo usando uma porta JMX específica e exibir informações sobre MapA, execute o seguinte comando:** UNIX

```
xsadmin.sh -g ObjectGridA -m MapSetA -mapsizes -fm MapA
  -ch CatalogMachine -p 6645
```

Windows

```
xsadmin.bat -g ObjectGridA -m MapSetA -mapsizes -fm MapA
  -ch CatalogMachine -p 6645
```

O utilitário de amostra xsAdmin conecta ao servidor MBean que está em execução em um servidor de catálogos. Um servidor de catálogos pode ser executado em um processo independente, processo do WebSphere Application Server ou integrado em um processo aplicativo customizado. Utilize a opção **-ch** para especificar o nome do host do serviço de catálogo e a opção **-p** para especificar a porta de nomenclatura do serviço de catálogo.

O tamanho do mapa especificado é exibido. A seguir está um exemplo de saída:

```
This administrative utility is provided as a sample only and is not to be
considered a fully supported component of the WebSphere eXtreme Scale product

Connecting to Catalog service at CatalogMachine:6645

*****Displaying Results for Grid - ObjectGridA, MapSet - MapSetA*****


*** Listing Maps for server1 ***
Map Name: MapA  Partition #: 0  Map Size: 0  Shard Type: Primary
Server Total: 0
```

- **Para conectar-se a um serviço de catálogo hospedado em um processo do WebSphere Application Server, execute as seguintes etapas:**

  A opção **-dmgr** é necessária ao conectar-se a um serviço de catálogo hospedado por qualquer processo do WebSphere Application Server ou cluster de processos. Utilize a opção **-ch** para especificar o nome do host se não `localhost` e a opção **-p** para substituir a porta de autoinicialização do serviço de catálogo, que utiliza o processo BOOTSTRAP_ADDRESS. A opção **-p** é necessária apenas se o BOOTSTRAP_ADDRESS não estiver configurado com o padrão de 9809.

  Nota: A versão independente do WebSphere eXtreme Scale não pode ser utilizada para conectar-se a um serviço de catálogo hospedado por um processo do WebSphere Application Server. Utilize o script xsAdmin incluído no diretório *was_root*`/bin`, que está disponível ao instalar o WebSphere eXtreme Scale no WebSphere Application Server ou WebSphere Application Server Network Deployment.

  a. Navegue até o diretório bin do WebSphere Application Server:

     ```
     cd wasRoot/bin
     ```

  b. Ative o utilitário xsAdmin usando o seguinte comando:

     UNIX

     ```
     xsadmin.sh -g ObjectGridA -m MapSetA -mapsizes -fm MapA -dmgr
     ```

     Windows

     ```
     xsadmin.bat -g ObjectGridA -m MapSetA -mapsizes -fm MapA -dmgr
     ```

O tamanho do mapa especificado é exibido.

```
This administrative utility is provided as a sample only and is not to be
considered a fully supported component of the WebSphere eXtreme Scale product

Connecting to Catalog service at localhost:9809

****Displaying Results for Grid - ObjectGridA, MapSet - MapSetA*****


*** Listing Maps for server1 ***
Map Name: MapA  Partition #: 0  Map Size: 0  Shard Type: Primary
Server Total: 0
```

Monitorando com a PMI do WebSphere Application Server

O WebSphere eXtreme Scale suporta Performance Monitoring Infrastructure (PMI) ao executar em um WebSphere Application Server ou servidor de aplicativos WebSphere Extended Deployment. A PMI coleta dados de desempenho em aplicativos de tempo de execução e fornece interfaces que suportam aplicativos externos para monitorar dados de desempenho. É possível utilizar o console administrativo ou a ferramenta wsadmin para acessar dados de monitoramento.

Antes de Iniciar

É possível utiliza a PMI para monitorar seu ambiente quando você está utilizando oWebSphere eXtreme Scale combinado com o WebSphere Application Server.

Sobre Esta Tarefa

O WebSphere eXtreme Scale utiliza o recurso PMI customizado do WebSphere Application Server para incluir sua própria instrumentação PMI. Com esta abordagem, é possível ativar e desativar a PMI do WebSphere eXtreme Scale com o console administrativo ou com as interfaces Java Management Extensions (JMX) na ferramenta wsadmin. Além disso, é possível acessar as estatísticas do WebSphere eXtreme Scale com a PMI padrão e as interfaces JMX que são utilizadas pela ferramentas de monitoramento, incluindo o Tivoli Performance Viewer.

Procedimento

1. Ative a PMI do eXtreme Scale. É necessário ativar a PMI para visualizar as estatísticas de PMI. Consulte “Ativando a PMI” para obter mais informações.
2. Recupere as estatísticas de PMI do eXtreme Scale. Visualize o desempenho dos seus aplicativos eXtreme Scale com o Tivoli Performance Viewer. Consulte “Recuperando Estatísticas PMI” na página 391 para obter mais informações.

O que Fazer Depois

Para obter mais informações sobre a ferramenta wsadmin, consulte “Acessando MBeans Utilizando a Ferramenta wsadmin” na página 353.

Ativando a PMI

É possível utilizar a WebSphere Application Server Performance Monitoring Infrastructure (PMI) para ativar ou desativar estatísticas em qualquer nível. Por exemplo, é possível optar por ativar as estatísticas de taxa de acesso do mapa para um mapa específico, mas não o número de estatísticas de entrada ou as estatísticas de tempo de atualização de lote do utilitário de carga. É possível ativar a PMI no console administrativo ou com scripts.

Antes de Iniciar

Seu servidor de aplicativos deve ser iniciado e ter um aplicativo ativado para o eXtreme Scale instalado. Para ativar a PMI com scripts, você também deve estar apto a efetuar login e utilizar a ferramenta wsadmin. Para obter informações adicionais sobre a ferramenta wsadmin, consulte o tópico ferramenta wsadmin no centro de informações do WebSphere Application Server.

Sobre Esta Tarefa

Utilize a PMI do WebSphere Application Server para fornecer um mecanismo granular com o qual é possível ativar ou desativar estatísticas em qualquer nível. Por exemplo, é possível optar por ativar as estatísticas de taxa de acesso do mapa para um mapa específico, mas não o número de estatísticas de entrada ou as estatísticas de tempo de atualização de lote do utilitário de carga. Esta seção mostra como utilizar o console administrativo e os scripts wsadmin para ativar a PMI do ObjectGrid.

Procedimento

- **Ativar a PMI no console administrativo.**
  1. No console administrativo, clique em **Monitoramento e Ajuste** → **Performance Monitoring Infrastructure** → *server_name*.
  2. Verifique se opção Ativar PMI (Performance Monitoring Infrastructure) está selecionada. Essa definição é ativada por padrão. Se a configuração não estiver ativada, selecione a caixa de opções e reinicie o servidor.
  3. Clique em **Customizar**. Na árvore de configuração, selecione o ObjectGrid e o módulo Mapas do ObjectGrid. Ative as estatísticas para cada módulo.

  A categoria de tipo de transação para estatísticas do ObjectGrid é criada no tempo de execução. É possível visualizar apenas as subcategorias das estatísticas do ObjectGrid e do Mapa na guia **Tempo de Execução**.
- **Ativar a PMI com scripts.**
  1. Abra um prompt de linha de comandos. Navegue para o diretório install_root/bin. Digite wsadmin para iniciar a ferramenta de linha de comandos wsadmin.
  2. Modifique a configuração do tempo de execução do PMI do eXtreme Scale. Verifique se a PMI está ativada para o servidor, por meio dos seguintes comandos:

     ```
     wsadmin>set s1 [$AdminConfig getid
     /Cell:CELL_NAME/Node:NODE_NAME/
         Server:APPLICATION_SERVER_NAME/]
     wsadmin>set pmi [$AdminConfig list PMIService $s1]
     wsadmin>$AdminConfig show $pmi.
     ```

     Se a PMI não estiver ativada, execute os seguintes comandos para ativá-la:

     ```
     wsadmin>$AdminConfig modify $pmi {{enable true}}
     wsadmin> $AdminConfig save
     ```

     Se precisar ativar a PMI, reinicie o servidor.
  3. Configure as variáveis para customizar o conjunto de estatísticas utilizando os seguintes comandos:

     ```
     wsadmin>set perfName [$AdminControl completeObjectName type=Perf,
     process=APPLICATION_SERVER_NAME,*]
     wsadmin>set perfOName [$AdminControl makeObjectName $perfName]
     ```

```
wsadmin>set params [java::new {java.lang.Object[]} 1]
wsadmin>$params set 0 [java::new java.lang.String custom]
wsadmin>set sigs [java::new {java.lang.String[]} 1]
wsadmin>$sigs set 0 java.lang.String
```

4. Configure o conjunto de estatísticas para customizar o seguinte comando:

```
wsadmin>$AdminControl invoke_jmx $perfOName setStatisticSet $params $sigs
```

5. Utilize os comandos a seguir para configurar as variáveis de modo a ativar a estatística da PMI objectGridModule:

```
wsadmin>set params [java::new {java.lang.Object[]} 2]
wsadmin>$params set 0 [java::new java.lang.String objectGridModule=1]
wsadmin>$params set 1 [java::new java.lang.Boolean false]
wsadmin>set sigs [java::new {java.lang.String[]} 2]
wsadmin>$sigs set 0 java.lang.String
wsadmin>$sigs set 1 java.lang.Boolean
```

6. Configure a cadeia de estatísticas por meio deste comando:

```
wsadmin>set params2 [java::new {java.lang.Object[]} 2]
wsadmin>$params2 set 0 [java::new java.lang.String mapModule=*]
wsadmin>$params2 set 1 [java::new java.lang.Boolean false]
wsadmin>set sigs2 [java::new {java.lang.String[]} 2]
wsadmin>$sigs2 set 0 java.lang.String
wsadmin>$sigs2 set 1 java.lang.Boolean
```

7. Configure a cadeia de estatísticas por meio deste comando:

```
wsadmin>$AdminControl invoke_jmx $perfOName setCustomSetString $params2 $sigs2
```

Essas etapas ativam o PMI do tempo de execução do eXtreme Scale, mas não modificam a configuração do PMI. Se você reiniciar o servidor de aplicativos, as configurações da PMI serão perdidas, exceto para a ativação da PMI principal.

Exemplo

É possível executar as seguintes etapas para ativar as estatísticas da PMI para o aplicativo de amostra:

1. Ative o aplicativo utilizando o endereço da Web `http://host:port/ObjectGridSample`, em que host e porta são o nome do host e número de porta HTTP do servidor no qual a amostra está instalada.
2. No aplicativo de amostra, clique consecutivamente em ObjectGridCreationServlet e nos botões de ação 1, 2, 3, 4 e 5 para gerar ações para o ObjectGrid e os mapas. Não feche esta página do servlet neste momento.
3. No console administrativo, clique em **Monitoramento e Ajuste** → **Performance Monitoring Infrastructure** → *server_name* Clique na guia **Tempo de Execução**.
4. Clique no botão de rádio **Customizado**.
5. Expanda o módulo Mapas do ObjectGrid na árvore de tempo de execução e clique no link clusterObjectGrid. No grupo Mapas do ObjectGrid, há uma instância do ObjectGrid denominada clusterObjectGrid; há quatro mapas debaixo do grupo clusterObjectGrid: contadores, funcionários, escritórios e sites. Na instância do ObjectGrids, existe uma instância do clusterObjectGrid e sob esta há um tipo de transação denominado DEFAULT.
6. É possível ativar as estatísticas de seu interesse. Por exemplo, é possível ativar o número de entradas para o mapa funcionários e o tipo de resposta de transação para o tipo de transação DEFAULT.

O que Fazer Depois

Quando ativar a PMI, será possível visualizar estatísticas de PMI com o console administrativo ou por meio de scripts.

Recuperando Estatísticas PMI

Ao recuperar estatísticas PMI, é possível visualizar o desempenho dos aplicativos eXtreme Scale.

Antes de Iniciar

- Ative o controle de estatísticas PMI para o seu ambiente. Consulte "Ativando a PMI" na página 388 para obter mais informações.
- Os caminhos nesta tarefa assumem que você está recuperando estatísticas para o aplicativo de amostra, mas é possível utilizar estas estatísticas para qualquer outro aplicativo com etapas semelhantes.
- Se estiver utilizando o console administrativo para recuperar estatísticas, você deve estar apto a efetuar login no console administrativo. Se estiver utilizando scripts, você deve estar apto a efetuar login no wsadmin.

Sobre Esta Tarefa

É possível recuperar estatísticas PMI para visualização no Tivoli Performance Viewer concluindo etapas no console administrativo ou com scripts.

- Etapas do console administrativo
- Etapas de scripts

Para obter mais informações sobre as estatísticas que podem ser recuperadas, consulte "Módulos PMI" na página 392.

Procedimento

- Recupere estatísticas PMI no console administrativo.
  1. No console administrativo, clique em **Monitoramento e Ajuste** → **Performance Viewer** → **Atividade Atual**
  2. Selecione o servidor que deseja monitorar utilizando o Tivoli Performance Viewer, em seguida, ative o monitoramento.
  3. Clique no servidor para visualizar a página do Performance Viewer.
  4. Expanda a árvore de configuração. Clique em **Mapas do ObjectGrid** → **clusterObjectGrid** e selecione **employees**. Expanda **ObjectGrids** → **clusterObjectGrid** e selecione **DEFAULT**.
  5. No aplicativo de amostra do ObjectGrid, retorne ao servlet ObjectGridCreationServlet, clique no botão 1, em seguida, ocupar mapas. É possível visualizar a estatística no visualizador.
- Recupere estatísticas PMI com scripts.
  1. Em um prompt de linha de comandos, navegue até o diretório `install_root/bin`. Digite `wsadmin` para iniciar a ferramenta wsadmin.
  2. Configure as variáveis para o ambiente utilizando os seguintes comandos:

     ```
     wsadmin>set perfName [$AdminControl completeObjectName type=Perf,*]
     wsadmin>set perfOName [$AdminControl makeObjectName $perfName]
     wsadmin>set mySrvName [$AdminControl completeObjectName type=Server,
      name=APPLICATION_SERVER_NAME,*]
     ```

  3. Configure as variáveis para obter o ambiente mapModule utilizando os seguintes comandos:

     ```
     wsadmin>set params [java::new {java.lang.Object[]} 3]
     wsadmin>$params set 0 [$AdminControl makeObjectName $mySrvName]
     wsadmin>$params set 1 [java::new java.lang.String mapModule]
     wsadmin>$params set 2 [java::new java.lang.Boolean true]
     ```

```
wsadmin>set sigs [java::new {java.lang.String[]} 3]
wsadmin>$sigs set 0 javax.management.ObjectName
wsadmin>$sigs set 1 java.lang.String
wsadmin>$sigs set 2 java.lang.Boolean
```

4. Obtenha a estatística mapModule utilizando os seguintes comandos:

```
wsadmin>$AdminControl invoke_jmx $perfOName getStatsString $params $sigs
```

5. Configure as variáveis para obter a estatística objectGridModule utilizando os seguintes comandos:

```
wsadmin>set params2 [java::new {java.lang.Object[]} 3]
wsadmin>$params2 set 0 [$AdminControl makeObjectName $mySrvName]
wsadmin>$params2 set 1 [java::new java.lang.String objectGridModule]
wsadmin>$params2 set 2 [java::new java.lang.Boolean true]
wsadmin>set sigs2 [java::new {java.lang.String[]} 3]
wsadmin>$sigs2 set 0 javax.management.ObjectName
wsadmin>$sigs2 set 1 java.lang.String
wsadmin>$sigs2 set 2 java.lang.Boolean
```

6. Obtenha a estatística objectGridModule utilizando os seguintes comandos:

```
wsadmin>$AdminControl invoke_jmx $perfOName getStatsString $params2 $sigs2
```

Resultados

É possível visualizar estatísticas no Tivoli Performance Viewer.

Módulos PMI

É possível monitorar o desempenho dos seus aplicativos com os módulos Performance Monitoring Infrastructure (PMI).

objectGridModule

O objectGridModule contém uma estatística de tempo: tempo de resposta de transação. Uma transação é definida como uma duração entre a chamada de método Session.begin e a chamada de método Session.commit. Esta duração é rastreada como o tempo de resposta da transação. O elemento-raiz do objectGridModule, "root", serve como ponto de entrada para as estatísticas do WebSphere eXtreme Scale. Este elemento-raiz possui ObjectGrids como seus elementos-filhos, que possuem tipos de transações como seus elementos-filhos. A estatística de tempo de resposta está associada a cada tipo de transação.

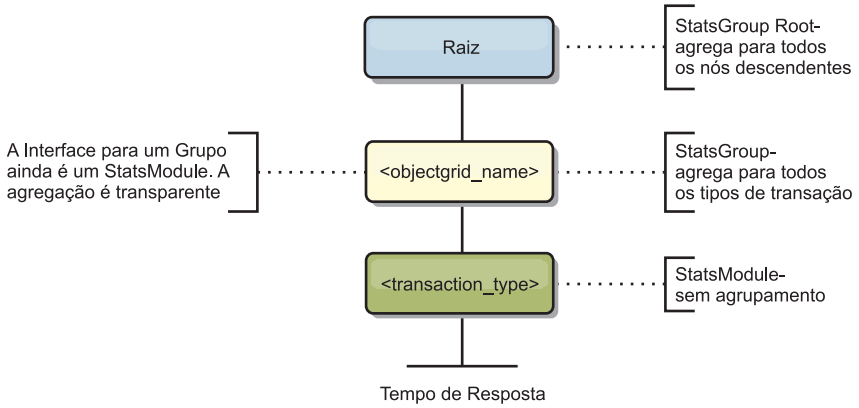


Figura 28. Estrutura do Módulo ObjectGridModule

O seguinte diagrama mostra um exemplo da estruturaObjectGridModule. Neste exemplo, duas instâncias ObjectGrid existem no sistema: ObjectGrid A e ObjectGrid

B. A instância ObjectGrid A possui dois tipos de transações: A e padrão. A instância ObjectGrid B possui apenas o tipo de transação padrão.

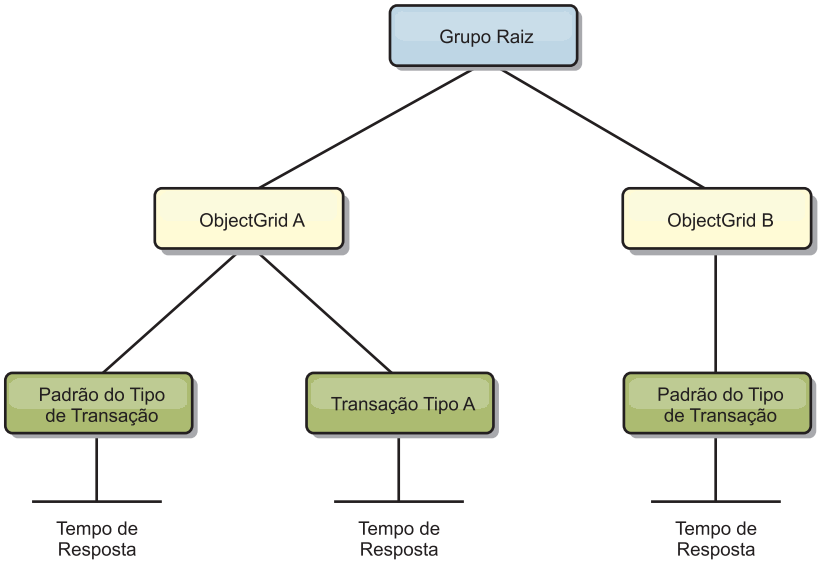


Figura 29. Exemplo da Estrutura do Módulo ObjectGridModule

Os tipos de transações são definidos por desenvolvedores de aplicativos, porque eles sabem quais tipos de transações seus aplicativos utilizam. O tipo de transação é configurando utilizando o método Session.setTransactionType(String) a seguir:

```
/**
* Configura o tipo de transação para futuras transações.
*
* Quando este método é chamado, todas as futuras transações terão o mesmo tipo
* até que outro tipo de transação seja configurado. Se nenhum tipo de transação
* estiver configurado, será utilizado o tipo de transação TRANSACTION_TYPE_DEFAULT
* padrão.
*
* Os tipos de transações são utilizados principalmente para finalidade de
rastreamento de dados estatísticos.
* Os usuários podem predefinir tipos de transações que são executadas em um
* application. A idéia é categorizar transações com as mesmas características
* para uma categoria (tipo), portanto, uma estatística de tempo de resposta de
 transação pode ser
* utilizada para rastrear cada tipo de transação.
*
* Este rastreamento é útil quando seu aplicativo tem diferentes tipos de
* transações.
* Entre eles, alguns tipos de transações, como transações de atualização,
têm um processamento
* mais longo que outras transações, como transações de leitura. Utilizando o tipo de
* transação, diferentes transações são rastreadas por diferentes estatísticas, portanto,
* as estatísticas podem ser mais úteis.
*
* @param tranType o tipo de transação para futuras transações.
*/
void setTransactionType(String tranType);
```

O exemplo a seguir configura o tipo de transação como `updatePrice`:

```
// Set the transaction type to updatePrice
// The time between session.begin() and session.commit() will be
// tracked in the time statistic for "updatePrice".
```

```
session.setTransactionType("updatePrice");
session.begin();
map.update(stockId, new Integer(100));
session.commit();
```

A primeira linha indica que o tipo de transação subseqüente é updatePrice. Uma estatística updatePrice existe na instância ObjectGrid que corresponde à sessão no exemplo. Utilizando interfaces Java Management Extensions (JMX), é possível obter o tempo de resposta da transação para transações updatePrice. Também é possível obter a estatística agregada para todos os tipos de transações na instância ObjectGrid especificada.

mapModule

O mapModule contém três estatísticas relacionadas aos mapas eXtreme Scale:

- **Taxa de ocorrências do mapa** - *BoundedRangeStatistic*: Controla a taxa de ocorrências de um mapa. A taxa de ocorrência é um valor flutuante entre 0 e 100 inclusivamente, que representa a porcentagem de ocorrências do mapa em relação a operações get do mapa.
- **Número de entradas**-*CountStatistic*: Controla o número de entradas no mapa.
- **Tempo de resposta de atualização de lote do utilitário de carga**-*TimeStatistic*: Controla o tempo de resposta que é utilizada para a operação de atualização de lote do utilitário de carga.

O elemento-raiz do mapModule, "root", serve como ponto de entrada para as estatísticas do Mapa ObjectGrid. Este elemento-raiz possui ObjectGrids como seus elementos-filhos, que possuem mapas como seus elementos-filhos. Cada instância do mapa possui três estatísticas listadas. A estrutura mapModule é mostrada no diagrama a seguir:

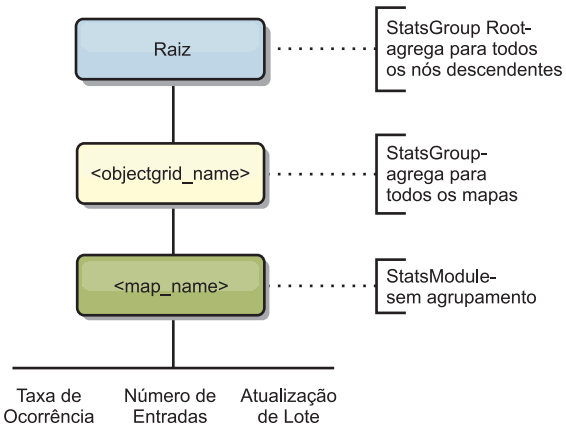
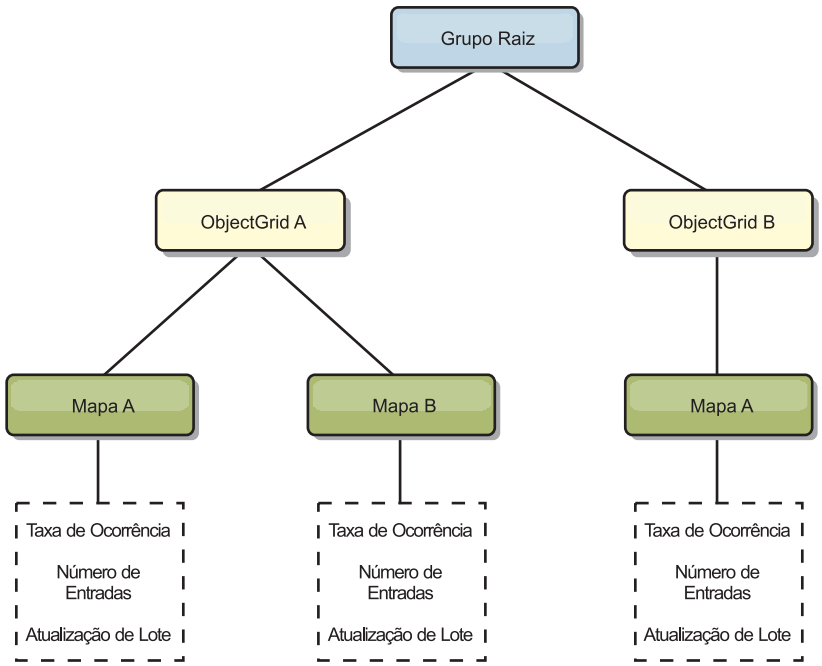


Figura 30. Estrutura do mapModule

O diagrama a seguir mostra um exemplo da estrutura mapModule:

Figura 31. Exemplo de Estrutura do Módulo mapModule



hashIndexModule

O hashIndexModule contém as seguintes estatísticas que são relacionadas aos índices de nível de Mapa:

- **Contagem de Localizações**-*CountStatistic*: O número de chamadas para a operação find do índice.
- **Contagem de Colisões**-*CountStatistic*: O número de colisões para a operação find.
- **Contagem de Falhas**-*CountStatistic*: O número de falhas para a operação find.
- **Contagem de Resultados**-*CountStatistic*: O número de chaves retornadas da operação find.
- **Contagem de BatchUpdate**-*CountStatistic*: O número de atualizações de lote junto a este índice. Quando o mapa correspondente é alterado de qualquer maneira, o método doBatchUpdate() do índice será chamado. Esta estatística informará com que frequência seu índice está sendo alterado ou atualizado.
- **Tempo de Duração da Operação Find**-*TimeStatistic*: A quantidade de tempo que a operação find leva para ser concluída

O elemento-raiz do hashIndexModule, "root", serve como ponto de entrada para as estatísticas do HashIndex. Este elemento-raiz possui ObjectGrids como seus elementos-filhos, os ObjectGrids possuem mapas como seus elementos-filhos, que, finalmente, possuem HashIndexes como seus elementos-filhos e nós-folhas da árvore. Cada instância do HashIndex possui três estatísticas listadas. A estrutura hashIndexModule é mostrada no diagrama a seguir:

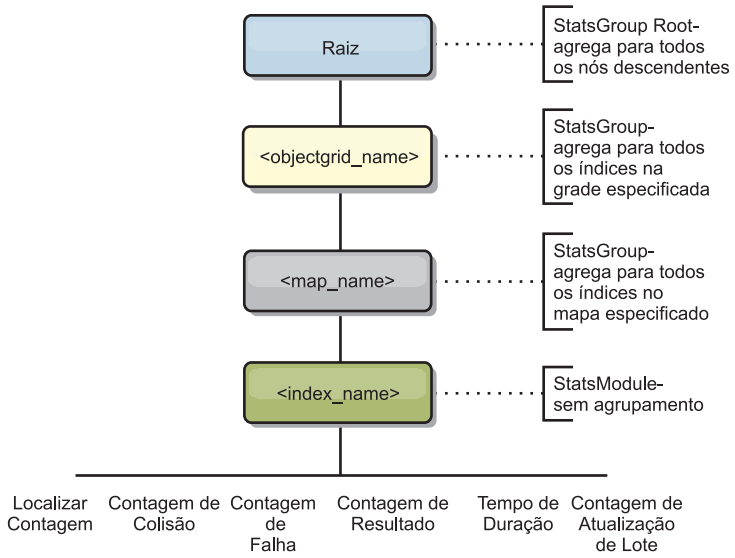


Figura 32. Estrutura do Módulo hashIndexModule

O diagrama a seguir mostra um exemplo da estrutura hashIndexModule:

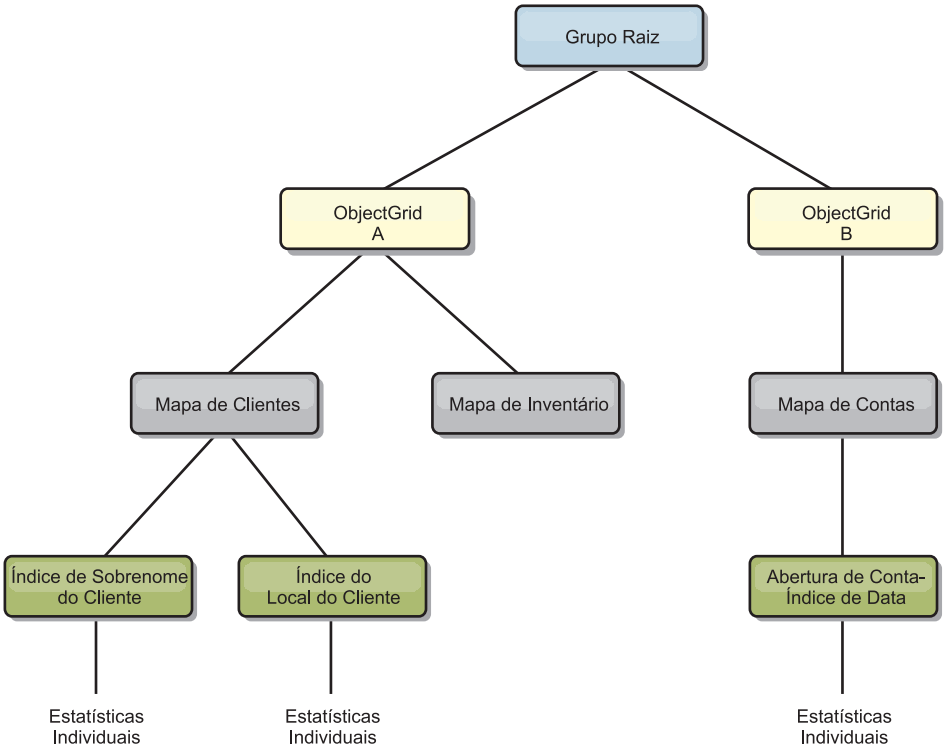


Figura 33. Exemplo da Estrutura do Módulo hashIndexModule

agentManagerModule

O agentManagerModule contém as estatísticas que são relacionadas aos agentes de nível de mapa:

- **Tempo de Redução**: *TimeStatistic* - A quantidade de tempo para o agente concluir a operação reduce.

- **Tempo de Duração Total**: *TimeStatistic* - A quantidade total de tempo para o agente concluir todas as operações.
- **Tempo de Serialização do Agente**: *TimeStatistic* - A quantidade de tempo para serialização do agente.
- **Tempo de Aumento do Agente**: *TimeStatistic* - A quantidade de tempo para aumentar o agente no servidor.
- **Tempo de Serialização do Resultado**: *TimeStatistic* - A quantidade de tempo para serializar os resultados do agente.
- **Tempo de Aumento do Resultado**: *TimeStatistic* - A quantidade de tempo para aumentar os resultados do agente.
- **Contagem de Falhas**: *CountStatistic* - O número de vezes que o agente falhou.
- **Contagem de Chamada**: *CountStatistic* - O número de vezes que o AgentManager foi chamado.
- **Contagem de Partições**: *CountStatistic* - O número de partições para as quais o agente é enviado.

O elemento-raiz do agentManagerModule, "root", serve como ponto de entrada para as estatísticas do AgentManager. O elemento-raiz possui ObjectGrids como seus elementos-filhos, os ObjectGrids possuem seus elementos-filhos, que, finalmente, possuem instâncias do AgentManager como seus elementos-filhos e nós-folhas da árvore. Cada instância do AgentManager possui três estatísticas listadas.

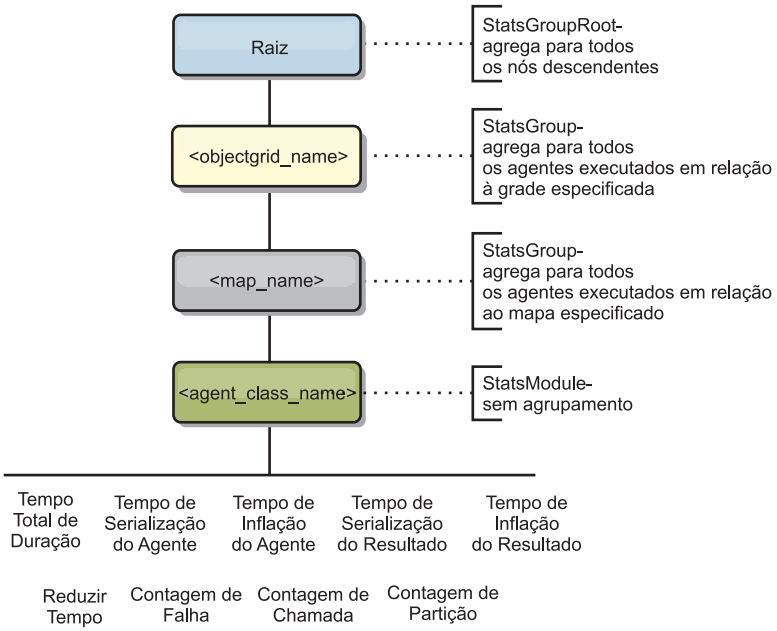


Figura 34. Estrutura agentManagerModule

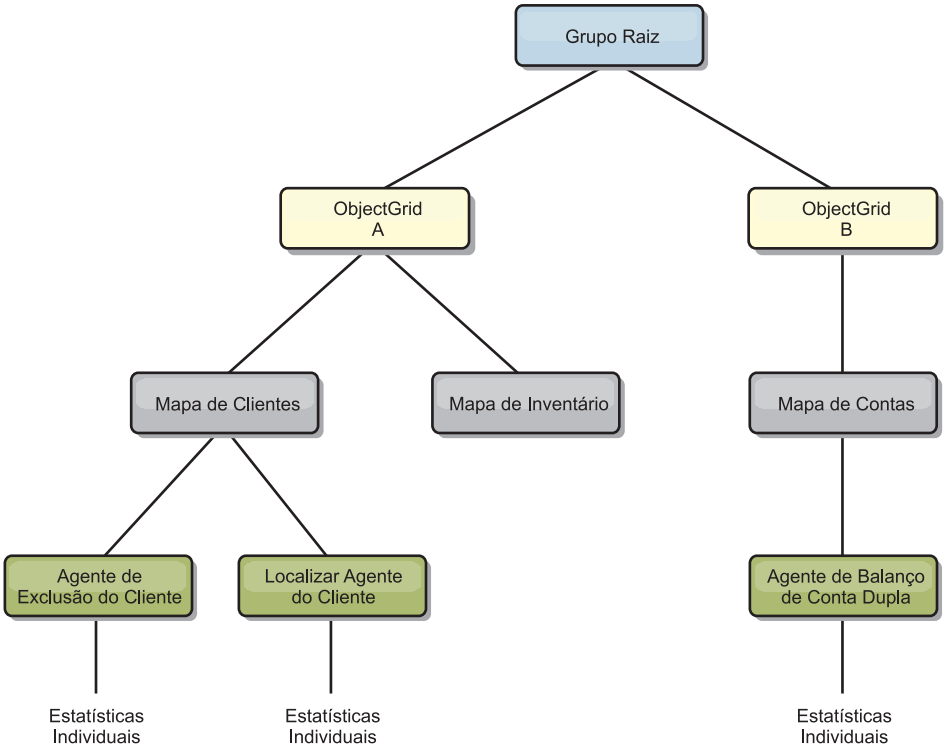


Figura 35. Exemplo da Estrutura agentManagerModule

queryModule

O queryModule contém estatísticas relacionadas às consultas do eXtreme Scale:

- **Tempo de Criação do Plano**: *TimeStatistic* - A quantidade de tempo para criar o plano de consulta.
- **Tempo de Execução**: *TimeStatistic* - A quantidade de tempo para executar a consulta.
- **Contagem de Execução**: *CountStatistic* - O número de vezes que a consulta foi executada.
- **Contagem de Resultados**: *CountStatistic* - A contagem para cada conjunto de resultados de cada execução de consulta.
- **FailureCount**: *CountStatistic* - O número de vezes que a consulta falhou.

O elemento-raiz do queryModule, "root", serve como ponto de entrada para as estatísticas do Query. Este elemento-raiz possui ObjectGrids como seus elementos-filhos, que possuem objetos Query como seus elementos-filhos e nós-folhas da árvore. Cada instância do Query possui as três estatísticas listadas.

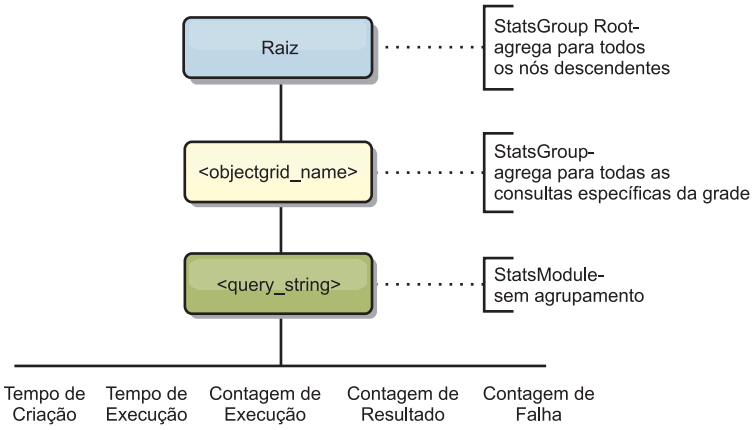


Figura 36. Estrutura queryModule

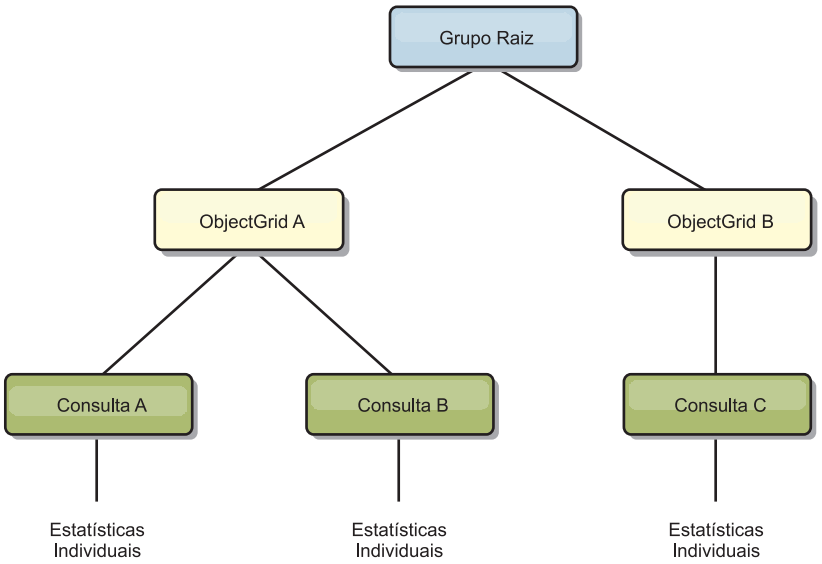


Figura 37. Exemplo da Estrutura queryModule QueryStats.jpg

Acessando MBeans Utilizando a Ferramenta wsadmin

É possível usar o utilitário wsadmin fornecido no WebSphere Application Server para acessar informações de MBean.

Execute a ferramenta wsadmin a partir do diretório bin em sua instalação do WebSphere Application Server. O exemplo a seguir recupera uma visualização da disposição do shard atual em um eXtreme Scale dinâmico. O wsadmin pode ser executado a partir de qualquer instalação na qual o eXtreme Scale está em execução. Não é necessário executar o wsadmin no serviço de catálogo.

```
$ wsadmin.sh -lang jython
wsadmin>placementService = AdminControl.queryNames
 ("com.ibm.websphere.objectgrid:*,type=PlacementService")
wsadmin>print AdminControl.invoke(placementService,
 "listObjectGridPlacement","library ms1")

<objectGrid name="library" mapSetName="ms1">
  <container name="container-0" zoneName="DefaultDomain"
  hostName="host1.company.org" serverName="server1">
     <shard type="Primary" partitionName="0"/>
```

```
    <shard type="SynchronousReplica" partitionName="1"/>
  </container>
  <container name="container-1" zoneName="DefaultDomain"
  hostName="host2.company.org" serverName="server2">
    <shard type="SynchronousReplica" partitionName="0"/>
    <shard type="Primary" partitionName="1"/>
  </container>
  <container name="UNASSIGNED" zoneName="_ibm_SYSTEM"
  hostName="UNASSIGNED" serverName="UNNAMED">
    <shard type="SynchronousReplica" partitionName="0"/>
    <shard type="AsynchronousReplica" partitionName="0"/>
  </container>
</objectGrid>
```

Monitoramento com Beans Gerenciados (MBeans)

É possível usar os beans gerenciados (MBeans) para controlar as estatísticas no seu ambiente.

Antes de Iniciar

Para que os atributos sejam registrados, é necessário ativar as estatísticas. É possível ativar as estatísticas de uma das seguintes formas:

- **Com o arquivo de propriedades do servidor:**

  É possível ativar as estatísticas no arquivo de propriedades do servidor com uma entrada de valor de chave `statsSpec=<StatsSpec>`. Alguns exemplos das possíveis configurações são:

  – Para ativar todas as estatísticas, use `statsSpec=all=enabled`

  – Para ativar apenas as estatísticas do ObjectGrid, use `statsSpec=og.all=enabled`. Para visualizar uma descrição de todas as especificações de estatísticas possíveis, consulte o StatsSpec API na documentação da API.

  Para obter mais informações sobre o arquivo de propriedades do servidor, consulte “Arquivo de Propriedades do Servidor” na página 183.

- **Com um bean gerenciado:**

  As estatísticas podem ser ativadas usando o atributo StatsSpec no MBean do ObjectGrid. Para obter detalhes adicionais, consulte a API StatsSpec

- **Programaticamente:**

  Também é possível ativar as estatísticas programaticamente com a interface StatsAccessor, que é recuperada com a classe StatsAccessorFactory. Utilize esta interface em um ambiente do cliente ou quando precisar monitorar um eXtreme Scale que está em execução em outro processo.

Exemplo

Para obter um exemplo de como usar os beans gerenciados, consulte “Monitorando com o Utilitário de Amostra xsAdmin” na página 385.

Monitorando com o Console da Web

Um dos novos recursos no release do eXtreme Scale 7.1 é um console da Web que permite a você representar graficamente as estatísticas atuais e históricas. Este console fornece alguns gráficos configurados para visões gerais resumidas e tem uma página de relatórios customizada que pode ser usada para construir gráficos a partir das estatísticas disponíveis. É possível usar os recursos gráficos no console

de monitoramento do WebSphere eXtreme Scale para visualizar o desempenho geral das grades de dados em seu ambiente.

Antes de Iniciar

O servidor do console é executado nos sistemas AIX, Linux ou Windows. O sistema do servidor do console deve ser capaz de conectar ao serviço de catálogo e o serviço de catálogo também deve ser capaz de conectar de volta ao servidor do console. É necessário de uma instalação do WebSphere eXtreme Scale independente no sistema que hospedará o servidor do console. O script startConsoleServer.bat|sh para iniciar o servidor do console está localizado no diretório XS_ROOT/ObjectGrid/bin de sua instalação.

Depois de criar suas grades de dados e configurar seus aplicativos para usarem as grades de dados, permita que passe algum tempo para que as estatísticas sejam disponibilizadas. Por exemplo, com uma grade de dados de cache dinâmico, as estatísticas não estarão disponíveis até que um WebSphere Application Server que está executando um cache dinâmico conecte à grade de dados de cache dinâmico. Se você estiver usando um coletivo, a inicialização coletiva deverá ser concluída antes que as estatísticas estejam disponíveis. No geral, aguarde um minuto após uma mudança de configuração maior para ver as mudanças em suas estatísticas.

Dica: Para visualizar informações mais específicas sobre qualquer ponto de dados em um gráfico, é possível mover o ponteiro do mouse sobre o ponto de dados.

Procedimento

- Inicie o servidor do console. O script startConsoleServer.bat|sh para iniciar o servidor do console está localizado no diretório XS_ROOT/ObjectGrid/bin de sua instalação.
- Efetue login no console.
  1. Do seu navegador da Web, navegue para `https://your.console.host:7443`, substituindo `your.console.host` pelo nome do host da máquina na qual você instalou o console.
  2. Efetue login no console.
     - **ID do usuário:** `admin`
     - **Senha:** `admin`

     A página de boas-vindas do console será exibida.
  3. Clique em **Definições > Configuração** para revisar a configuração do console. A configuração do console inclui informações como:
     - Cadeia de rastreio para o cliente do WebSphere eXtreme Scale, como `*=all=disabled`
     - O nome e a senha do Administrador
     - O endereço de e-mail do Administrador
- Visualize o status da conexão.
  1. Navegue para a página de boas-vindas clicando no link **Home** na barra de ferramentas.
  2. No lado direito da barra de ferramentas, localize a lista suspensa que mostra o domínio ao qual o servidor do console está conectado. No lado direito da lista suspensa, há um indicador do status da conexão.
- Estabeleça e mantenha conexões com servidores de catálogos que você deseja monitorar.

1. Navegue para a página **Configurações > Servidores de Catálogos do eXtreme Scale**.
2. Inclua um novo servidor de catálogos.
   a. Clique no sinal de mais para exibir um diálogo para registrar um servidor de catálogos existente.
   b. Forneça informações, como o nome do host, a porta JMX e a porta listener.

      Nome do host

      Exibe o nome do host da estação de trabalho na qual o serviço de catálogo está em execução.

      Porta JMX

      Exibe o número da porta por meio de conexões JMX/RMI ativadas. O JMX ativa o monitoramento e o gerenciamento para os sistemas remotos.

      Porta Listener

      Exibe a porta listener para comunicação com o Internet Inter-ORB Protocol (IIOP).
   c. Clique em **OK**.
   d. Verifique se o servidor de catálogos foi incluído na árvore de navegação.

   Para visualizar informações sobre um servidor de catálogos existente, clique no nome do servidor de catálogos na árvore de navegação na página **Configurações > Servidores de Catálogos do eXtreme Scale**.

- Estabeleça e mantenha conexões com domínios que você deseja monitorar.
  1. Navegue para a página **Configurações > Domínios do eXtreme Scale**.
  2. Inclua um novo domínio.
     a. Clique no sinal de mais para exibir um diálogo para registrar um domínio do serviço de catálogo.
     b. Forneça informações.

        Name Exibe o nome do host do domínio, conforme designado pelo administrador.

        Usuário JMX
        Exibe o nome de usuário do Java Management Extensions (JMX) em uso (se a segurança estiver ativada).

        Senha JMX
        Exibe a senha JMX, se a segurança estiver ativada.

        Servidores de catálogos
        Lista um ou mais servidores de catálogos que pertencem ao domínio selecionado.

        Classe de gerador
        Exibe o nome da classe que implementa a interface CredentialGenerator. Essa classe é usada para obter credenciais para os clientes.

        Propriedades do gerador
        Exibe as propriedades para a classe de implementação CredentialGenerator. As propriedades são configuradas para o objeto com o método setProperties(String). O valor

credentialGeneratorprops é usado apenas se o valor da propriedade credentialGeneratorClass não for nulo.

c. Clique em OK.

d. Verifique se o domínio foi incluído na árvore de navegação.

3. Especifique as informações de segurança requeridas nesta página.

4. Associe o domínio a servidores de catálogos incluídos anteriormente.

Para visualizar informações sobre um domínio existente, clique no nome do servidor de catálogos na árvore de navegação na página **Configurações > Domínios do eXtreme Scale**.

- Para visualizar as estatísticas atuais do servidor, clique em **Monitorar > Visão Geral do Servidor**.

  Estatísticas do servidor

  Memória Usada
  Exibe a quantia atual de memória usada (real) no tempo de execução do servidor. Apenas os 25 principais servidores que estão usando a maior parte da memória são exibidos.

  Total de Memória com o Passar do Tempo
  Exibe o uso de memória real no tempo de execução do servidor.

  Memória Usada com o Passar do Tempo
  Exibe a quantia de memória usada no tempo de execução do servidor.

- Para visualizar o desempenho de todas as suas grades de dados, clique em **Monitorar > Visão geral do domínio da grade de dados**.

  Estatísticas da visão geral do domínio da grade de dados

  Distribuição da Capacidade Usada da Grade de Dados Atual
  Este gráfico contém uma figura do conjunto total e os principais consumidores da capacidade usada. Apenas as 25 principais grades de dados são exibidas.

  Cinco Principais Grades de Dados por Tempo Médio de Transação em Milissegundos
  Este gráfico contém uma lista dos cinco principais caches de dados, organizados pelo tempo médio de transação.

  Cinco Principais Grades de Dados por Rendimento Médio em Transações/Segundo
  Este gráfico contém uma lista das cinco principais grades de dados, organizadas pelo rendimento médio, organizado por transações/segundo

- Para visualizar as grades de dados individuais, clique em **Monitorar > Visão geral da grade de dados > *data_grid_name*.** Esta página mostra um resumo que inclui o número de entradas de cache, o tempo médio de transação e o rendimento médio. É possível também visualizar os seguintes gráficos:

  Estatísticas da visão geral da grade de dados

  Resumo Atual
  Exibe estatísticas tais como o número atual de objetos em cache nesta grade (entradas de cache), o tempo médio de transação e o rendimento médio de transação.

  Capacidade usada vs. Número de entradas de cache
  Este gráfico mostra a capacidade usada do cache versus o número de entradas no cache. É possível editar o tempo médio que é exibido:

última hora, último dia, última semana, último mês. O nível de detalhe que é mostrado no gráfico varia dependendo do intervalo de tempo que você selecionar.

Total de pedidos get de cache vs. Pedidos bem-sucedidos get de cache
Este gráfico ajuda a visualizar o número de consultas bem-sucedidas para o cache.

- Para visualizar detalhes adicionais sobre uma grade de dados específica, clique em **Monitorar > Detalhes da grade de dados**. Uma árvore é exibida com todas as grades de dados em sua configuração. É possível fazer uma pesquisa detalhada em uma grade de dados específica para visualizar os mapas que fazem parte dessa grade de dados. É possível clicar no nome de uma grade de dados ou em um mapa para obter informações adicionais.

  Estatísticas da grade de dados

  Resumo Atual
  Visualize a capacidade usada atual e uma lista de zonas às quais pertence a grade de dados.

  Distribuição da Capacidade Usada do Mapa da Grade de Objeto do eXtreme Scale Atual
  Visualize um conjunto total, que inclui a capacidade por zona e a capacidade total em cada zona. Apenas os 25 principais mapas de ObjectGrid são exibidos.

  Distribuição da Capacidade Usada da Zona Atual
  Visualize uma zona, que inclui o conjunto total e os principais consumidores da capacidade usada. Apenas as 25 principais zonas são exibidas.

  Estatísticas do mapa

  Resumo Atual
  Exibe estatísticas como:

  Capacidade Usada
  Visualize a capacidade usada e uma lista das zonas às quais pertence o mapa.

  Número de Entradas de Cache
  Exibe o número de objetos em cache no mapa.

  Tempo Médio da Transação (ms)
  Exibe o tempo médio de conclusão para transações envolvendo este mapa.

  Rendimento Médio da Transação (trans/seg)
  Exibe o número médio de transações por segundo, envolvendo este mapa.

  Distribuição da Capacidade Usada da Partição Atual
  Este gráfico contém uma figura do conjunto total e os principais consumidores da capacidade usada. Apenas as 25 principais partições são exibidas.

- Para escolher quais estatísticas você gostaria que seu relatório customizado contivesse, clique em **Monitorar > Relatórios Customizados**.

  Use esta visualização para construir gráficos de dados detalhados das várias estatísticas. Use a árvore para explorar os servidores e as grades de dados disponíveis e suas estatísticas associadas. Um menu é aberto quando você clica ou pressiona Enter em um nó que faz referência a dados que podem ser

representados graficamente. Crie um novo gráfico contendo as estatísticas ou inclua as estatísticas em um gráfico existente com estatísticas compatíveis.

Estatísticas do domínio

Tempo Médio da Transação (ms)
Exibe o tempo médio necessário para concluir uma transação neste domínio.

Rendimento Médio da Transação (trans/seg)
Exibe o número médio de transações por segundo neste domínio.

Tempo Máximo da Transação (ms)
Exibe o tempo gasto pela transação que *mais* consome tempo neste domínio.

Tempo Mínimo da Transação (ms)
Exibe o tempo gasto pela transação que *menos* consome tempo neste domínio.

Tempo Total da Transação (ms)
Exibe o tempo total gasto em transações neste domínio, desde o momento em que o domínio foi inicializado.

Estatísticas do contêiner do eXtreme Scale

Tempo Médio da Transação (ms)
Exibe o tempo médio necessário para concluir uma transação para este servidor de catálogos.

Rendimento Médio da Transação (trans/seg)
Exibe o número médio de transações por segundo para este servidor de catálogos.

Tempo Máximo da Transação (ms)
Exibe o tempo gasto pela transação que *mais* consome tempo para este servidor de catálogos.

Tempo Mínimo da Transação (ms)
Exibe o tempo gasto pela transação que *menos* consome tempo para este servidor de catálogos.

Tempo Total da Transação (ms)
Exibe o tempo total gasto em transações para este servidor de catálogos, desde o momento em que este servidor de catálogos foi inicializado.

Total de Entradas em Cache
Exibe o número atual de objetos em cache nas grades examinadas por este servidor de catálogos.

Máximo de Entradas em Cache
Exibe o número máximo de objetos em cache nas grades examinadas por este servidor de catálogos.

Mínimo de Entradas em Cache
Exibe o número mínimo de objetos em cache nas grades examinadas por este servidor de catálogos.

Taxa de ocorrência (porcentagem)
Exibe a taxa de ocorrência (taxa de acertos) para a grade de dados selecionada. Uma taxa alta de ocorrência é desejável. A taxa de ocorrência indica como a grade está ajudando a evitar o acesso ao armazenamento persistente.

Bytes Usados
Exibe o consumo de memória por este mapa.

Mínimo de Bytes Usados
Exibe o ponto baixo no consumo de memória por este serviço de catálogo e seus mapas.

Máximo de Bytes Usados
Exibe o ponto alto no consumo de memória por este serviço de catálogo e seus mapas.

Número Total de Ocorrências
Exibe o número total de vezes que os dados solicitados foram localizados no mapa, evitando a necessidade de acessar o armazenamento persistente.

Número Total de Gets
Exibe o número total de vezes que o mapa teve que acessar o armazenamento persistente para obter dados.

Heap Grátis (MB)
Exibe a quantia real de heap disponível para a JVM que está sendo usada pelo servidor de catálogos.

Total de Heap
Exibe a quantia de heap disponível para a JVM que está sendo usada por este servidor de catálogos.

Memória Usada
Exibe a memória usada na JVM que está sendo usada por este servidor de catálogos.

Número de Processadores Disponíveis
Exibe o número de CPUs disponíveis para este serviço de catálogo e seus mapas. Para obter maior estabilidade, execute os seus servidores a um carregamento de processador de 60% e o JVM heaps a um carregamento de heap de 60%. Os picos podem, então, conduzir o uso do processador para 80–90%, mas não execute seus servidores regularmente em níveis superiores a estes

Tamanho Máximo de Heap (MB)
Exibe a quantia máxima de heap disponível para a JVM que está sendo usada por este servidor de catálogos.

Estatísticas da grade

Tempo Médio da Transação (ms)
Exibe o tempo médio necessário para concluir transações envolvendo esta grade.

Rendimento Médio da Transação (trans/seg)
Exibe o número médio de transações por segundo concluídas por esta grade.

Tempo Máximo da Transação (ms)
Exibe o tempo gasto pela transação que *mais* consome tempo concluída por esta grade.

Tempo Mínimo da Transação (ms)
Exibe o tempo gasto pela transação que *menos* consome tempo concluída por esta grade.

Tempo Total da Transação (ms)
Exibe o período total de tempo de processamento de transações para esta grade.

Estatísticas do mapa

Total de Entradas em Cache
Exibe o número atual de objetos em cache neste mapa.

Máximo de Entradas em Cache
Exibe o número máximo de objetos em cache neste mapa desde o momento em que o mapa foi inicializado.

Mínimo de Entradas em Cache
Exibe o número mínimo de objetos em cache neste mapa desde o momento em que o mapa foi inicializado.

Taxa de Ocorrência (porcentagem)
Exibe a taxa de ocorrência (taxa de acertos) para o mapa selecionado. Uma taxa alta de ocorrência é desejável. A taxa de ocorrência indica como o mapa está ajudando a evitar o acesso ao armazenamento persistente.

Bytes Usados
Exibe o consumo de memória por este mapa.

Mínimo de Bytes Usados
Exibe o consumo mínimo (em Bytes) para este mapa.

Máximo de Bytes Usados
Exibe o consumo máximo (em Bytes) para este mapa.

Número Total de Ocorrências
Exibe o número total de vezes que os dados solicitados foram localizados no mapa, evitando a necessidade de acessar o armazenamento persistente.

Número Total de Gets
Exibe o número total de vezes que o mapa teve que acessar o armazenamento persistente para obter dados.

Heap Grátis (MB)
Exibe a quantia atual de heap disponível para este mapa, na JVM que está sendo usada pelo servidor de catálogos.

Total de Heap (MB)
Exibe a quantia total de heap disponível para este mapa, na JVM que está sendo usada pelo servidor de catálogos. Para obter maior estabilidade, execute os seus servidores a um carregamento de processador de 60% e o JVM heaps a um carregamento de heap de 60%. Os picos podem, então, conduzir o uso do processador para 80–90%, mas não execute seus servidores regularmente em níveis superiores a estes

Memória Usada (MB)
Exibe a quantia usada de memória neste mapa.

Número de Processadores Disponíveis
Exibe o número de CPUs disponíveis para este mapa. Para obter maior estabilidade, execute os seus servidores a um carregamento de processador de 60% e o JVM heaps a um carregamento de heap de 60%.

Os picos podem, então, conduzir o uso do processador para 80–90%, mas não execute seus servidores regularmente em níveis superiores a estes

Tamanho Máximo de Heap (MB)
Exibe a quantia máxima de heap disponível para este mapa, na JVM que está sendo usada pelo servidor de catálogos.

Monitorando com Ferramentas do Fornecedor

O WebSphere eXtreme Scale pode ser monitorado usando diversas soluções populares de monitoramento corporativo. Os agentes do plug-in estão incluídos no IBM Tivoli Monitoring and Hyperic HQ, os quais monitoram o WebSphere eXtreme Scale usando os beans de gerenciamento acessíveis. O CA Wily Introscope usa a instrumentação do método Java para capturar estatísticas.

Monitorando com o IBM Tivoli Enterprise Monitoring Agent para WebSphere eXtreme Scale

O IBM Tivoli Enterprise Monitoring Agent é uma solução de monitoramento cheia de recursos que pode ser usada para monitorar bancos de dados, sistemas operacionais e servidores em ambientes distribuídos e de host. O WebSphere eXtreme Scale inclui um agente customizado que pode ser usado para fazer introspecção dos beans de gerenciamento do eXtreme Scale. Essa solução trabalha eficientemente para ambas as implementações do eXtreme Scale e do WebSphere Application Server independentes.

Antes de Iniciar

- Instale o WebSphere eXtreme Scale Versão 7.0.0 ou superior.

  Além disso, as estatísticas devem ser ativadas para coletar dados estatísticos de servidores WebSphere eXtreme Scale. Várias opções para ativar estatísticas são descritas no “Monitoramento com Beans Gerenciados (MBeans)” na página 400 e no “Monitorando com o Utilitário de Amostra xsAdmin” na página 385
- Instale o IBM Tivoli Monitoring Versão 6.2.1 com o fix pack 2 ou superior.
- Instale o agente de S.O. Tivoli em cada servidor ou host no qual o servidor do eXtreme Scale é executado.
- Instale o agente do WebSphere eXtreme Scale, que pode ser obtido por download gratuitamente a partir do Site do IBM Open Process Automation Library (OPAL).

Conclua as seguintes etapas para instalar e configurar o Tivoli Monitoring Agent:

Procedimento

1. Instale o Tivoli Monitoring Agent para WebSphere eXtreme Scale.

   Faça download da imagem de instalação do Tivoli e extraia os arquivos para um diretório temporário.
2. Instale os arquivos de suporte do aplicativo eXtreme Scale.

   Instale o suporte do aplicativo eXtreme Scale em cada uma das seguintes implementações.
   - Tivoli Enterprise Portal Server (TEPS)
   - Enterprise Desktop client (TEPD)
   - Tivoli Enterprise Monitoring Server (TEMS)

a. No diretório temporário que você criou, inicie uma nova janela de comando e execute o arquivo executável apropriado para sua plataforma. O script de instalação detecta automaticamente o tipo de implementação do Tivoli (TEMS, TEPD ou TEPS). É possível instalar qualquer tipo em um único host ou em vários hosts e todos os três tipos de implementação requerem a instalação dos arquivos de suporte de aplicativo do agente eXtreme Scale.

b. Na janela do **Instalador**, verifique se as seleções para os componentes Tivoli implementados estão corretos. Clique em **Avançar**.

c. Se for solicitado, envie o nome do host e as credenciais administrativas. Clique em **Avançar**.

d. Selecione **Monitoring Agent para WebSphere eXtreme Scale**. Clique em **Avançar**.

e. Você será notificado sobre quais ações de instalação deverão ser executadas. Clique em **Avançar** para visualizar o progresso da instalação até a conclusão.

Depois de concluir o procedimento, todos os arquivos de suporte do aplicativo necessários pelo agente do WebSphere eXtreme Scale serão instalados.

3. Instale o agente em cada um dos nós do eXtreme Scale.

   Instale um agente do S.O. Tivoli em cada um dos computadores. Não é necessário configurar ou iniciar este agente. Use a mesma imagem de instalação da etapa anterior para executar o arquivo executável específico da plataforma.

   Como uma recomendação, você precisa instalar apenas um agente por host. Cada agente é capaz de suportar várias instâncias dos servidores eXtreme Scale. Para obter melhor desempenho, utiliza uma instância do agente para monitorar cerca de 50 servidores eXtreme Scale.

   a. Na tela de boas-vindas do assistente de instalação, clique em **Avançar** para abrir a tela para especificar as informações do caminho de instalação.

   b. Para o campo **Diretório de Instalação do Tivoli Monitoring**, insira ou procure por `C:\IBM\ITM` (ou `/opt/IBM/ITM`). Em seguida para o campo **Local para a Mídia de Instalação**, verifique se o valor exibido está correto e clique em **Avançar**.

   c. Selecione os componentes que deseja incluir, como **Executar uma instalação local da solução** e clique em **Avançar**.

   d. Selecione os aplicativos para os quais o suporte será incluído ao selecionar o aplicativo, como **Monitoring Agent para WebSphere eXtreme Scale** e clique em **Avançar**.

   e. É possível visualizar o progresso até o suporte do aplicativo ser incluído com êxito.

   Nota: Repita essas etapas em cada um dos nós do eXtreme Scale. Também é possível utilizar a instalação silenciosa. Consulte o IBM Tivoli Monitoring Information Center para obter mais informações sobre a instalação silenciosa.

4. Configure o agente do WebSphere eXtreme Scale.

   Cada um dos agentes instalado precisa ser configurado para monitorar qualquer servidor de catálogos, servidor eXtreme Scale ou ambos.

   As etapas para configurar as plataformas Windows eUNIX são diferentes. A configuração para a plataforma Windows é feita com a interface com o usuário **Gerenciar o Tivoli Monitoring Services**. A configuração para as plataformas UNIX baseia-se na linha de comandos.

   Windows Use as seguintes etapas para configurar inicialmente o agente no Windows

a. Na janela **Gerenciar o Tivoli Enterprise Monitoring Services**, clique em **Iniciar → Todos os Programas → IBM Tivoli Monitoring → Gerenciar o Tivoli Monitoring Services**.
b. Clique com o botão direito do mouse em **Monitoring Agent para WebSphere eXtreme Scale** e selecione **Configurar usando padrão**, que abre uma janela para criar uma instância exclusiva do agente.
c. Escolha um nome exclusivo, como por exemplo, `instance1` e clique em **Avançar**.

- Se planejar monitorar servidores do eXtreme Scale independentes, conclua as seguintes etapas:
  a. Atualize os parâmetros Java e certifique-se de que o valor **Java Home** esteja correto. Os argumentos da JVM podem ser deixados vazios. Clique em **Avançar**.
  b. Selecione o **Tipo de Conexão do Servidor MBean** e use `Servidor Compatível com JSR-160` para servidores eXtreme Scale independentes. Clique em **Avançar**.
  c. Se a segurança estiver ativada, atualize os valores de **ID do Usuário** e **Senha**. Deixe o valor **URL de Serviço JMX** como está. Substitua esse valor depois. Deixe o campo **Informações do Caminho da Classe JMX** como está. Clique em **Avançar**.

  Para configurar os servidores para o agente no Windows, conclua as seguintes etapas:
  a. Configure as instâncias de subnó dos servidores do eXtreme Scale na área de janela dos **Servidores de Grade do WebSphere eXtreme Scale**. Se nenhum servidor de contêiner existir no computador, clique em **Avançar** para continuar com a área de janela de serviço de catálogo.
  b. Se vários servidores de contêiner do eXtreme Scale existirem no computador, configure o agente para monitorar cada um dos servidores.
  c. É possível incluir quantos servidores do eXtreme Scale forem necessários se os nomes e portas forem exclusivos ao clicar em **Novo**. (Quando um servidor do eXtreme Scale for iniciado, um valor JMXPort deverá ser especificado).
  d. Depois de configurar os servidores de contêiner, clique em **Avançar**, para levá-lo para a área de janela **Servidores de Catálogo do WebSphere eXtreme Scale**.
  e. Se você não possuir nenhum servidor de catálogo, clique em **OK**. Se você tiver servidores de catálogo, inclua uma nova configuração para cada servidor, conforme foi feito com os servidores de contêiner. Novamente, escolha um nome exclusivo, de preferência o mesmo nome que foi usado ao iniciar o serviço de catálogo. Clique **OK** para concluir.
- Se você planejar monitorar servidores para o agente nos servidores do eXtreme Scale que são integrados em um processo do WebSphere Application Server, conclua as seguintes etapas:
  a. Atualize os parâmetros Java e certifique-se de que o valor **Java Home** esteja correto. Os argumentos da JVM podem ser deixados vazios. Clique em **Avançar**.
  b. Selecione o **Tipo de Conexão do Servidor MBean**. Selecione a versão do WebSphere Application Server que seja apropriada para seu ambiente. Clique em **Avançar**.
  c. Certifique-se de que as informações do WebSphere Application Server no painel estejam corretas. Clique em **Avançar**.

d. Inclua apenas uma definição de subnó. Forneça um nome para a definição de subnó, mas não atualize a definição de porta. No ambiente do WebSphere Application Server, os dados podem ser coletados de todos os servidores de aplicativos que forem gerenciados pelo agente do nó que estiver em execução no computador. Clique em **Avançar**.

e. Se nenhum servidor de catálogo existir no ambiente, clique em **OK**. Se você tiver servidores de catálogo, inclua uma nova configuração para cada servidor, conforme foi feito com os servidores de contêiner. Escolha um nome exclusivo para o serviço de catálogo, de preferência o mesmo nome que foi usado ao iniciar o serviço de catálogo. Clique **OK** para concluir.

Nota: Os servidores de contêineres não precisam ser colocados junto com o serviço de catálogo.

Agora que o agente e os servidores estão configurados e prontos, na próxima janela, clique com o botão direito do mouse em `instance1` para iniciar o agente.

UNIX Para configurar o agente na plataforma UNIX na linha de comandos, conclua as seguintes etapas:

A seguir há um exemplo de servidores independentes que usam um tipo de conexão Compatível com JSR160. O exemplo mostra três contêineres do eXtreme Scale em um único host (rhea00b02) e os endereços de listener JMX são 15000, 15001 e 15002, respectivamente. Não há nenhum servidor de catálogo.

A saída do utilitário de configuração é exibida em *`itálico de espaço simples`*, enquanto que a resposta do usuário está em **`negrito de espaço simples`**. (Se nenhuma resposta do usuário era necessária, o padrão foi selecionado pressionando a tecla enter).

```
rhea00b02 # ./itmcmd config -A xt
Agent configuration started...
Enter instance name (default is: ):  inst1
Edit "Monitoring Agent for
WebSphere eXtreme Scale"
settings? [ 1=Yes, 2=No ] (default is: 1):
Edit 'Java' settings? [ 1=Yes, 2=No ] (default is: 1):
Java home (default is: C:\Program Files\IBM\Java50):  /opt/OG61/java
Java trace level [ 1=Error,
2=Warning, 3=Information, 4=Minimum Debug, 5=Medium Debug, 6=Maximum
Debug,
    7=All ] (default is: 1):
JVM arguments (default is: ):
Edit 'Connection' settings? [ 1=Yes, 2=No ] (default is: 1):
MBean server connection type [ 1=JSR-160-Compliant Server, 2=WebSphere Application Server version 6.0,
3=WebSphere Application Server version 6.1, 4=WebSphere Application Server version 7.0 ] (default is: 1): 1
Edit 'JSR-160-Compliant Server' settings? [ 1=Yes, 2=No ] (default is: 1):
JMX user ID (default is: ):
Enter JMX password (default is: ):
Re-type : JMX password (default is: ):
JMX service URL (default is: service:jmx:rmi:///jndi/rmi://localhost:port/objectgrid/MBeanServer):
----------------------------------------
JMX Class Path Information
JMX base paths (default is: ):
JMX class path (default is: ):
JMX JAR directories (default is: ):
Edit 'WebSphere eXtreme Scale Catalog Service' settings? [ 1=Yes, 2=No ] (default is: 1):  2
Edit 'WebSphere eXtreme Scale Grid Servers' settings? [ 1=Yes, 2=No ] (default is: 1): 1
No 'WebSphere eXtreme Scale Grid Servers' settings available?
Edit 'WebSphere eXtreme Scale Grid Servers' settings, [1=Add, 2=Edit, 3=Del, 4=Next, 5=Exit] (default is: 4): 1
WebSphere eXtreme
Scale Grid Servers (default is: ): rhea00b02_c0
JMX service URL (default is: service:jmx:rmi:///jndi/rmi://localhost:<port>/objectgrid/MBeanServer):
service:jmx:rmi:///jndi/rmi://localhost:15000/objectgrid/MBeanServer

'WebSphere eXtreme Scale Grid Servers' settings:
WebSphere eXtreme Scale
Grid Servers=ogx
```

```
Edit 'WebSphere eXtreme Scale Grid Servers' settings, [1=Add, 2=Edit, 3=Del, 4=Next, 5=Exit] (default is: 4): 1
WebSphere eXtreme Scale Grid Servers (default is: ): rhea00b02_c1
JMX service URL (default is: service:jmx:rmi:///jndi/rmi://localhost:<port>/objectgrid/MBeanServer):
service:jmx:rmi:///jndi/rmi://localhost:15001/objectgrid/MBeanServer

'WebSphere eXtreme Scale Grid Servers' settings:
WebSphere eXtreme Scale
Grid Servers= rhea00b02_c1
Edit 'WebSphere eXtreme Scale Grid Servers' settings, [1=Add, 2=Edit, 3=Del, 4=Next, 5=Exit] (default is: 4): 1
WebSphere eXtreme Scale Grid Servers (default is: ): rhea00b02_c2
JMX service URL (default is: service:jmx:rmi:///jndi/rmi://localhost:<port>/objectgrid/MBeanServer):
service:jmx:rmi:///jndi/rmi://localhost:15002/objectgrid/MBeanServer

'WebSphere eXtreme Scale Grid Servers' settings:
WebSphere eXtreme Scale
Grid Servers= rhea00b02_c2
Edit 'WebSphere eXtreme Scale Grid Servers' settings, [1=Add, 2=Edit, 3=Del, 4=Next, 5=Exit] (default is: 4): 5

Will this agent connect to a TEMS? [1=YES, 2=NO] (Default is: 1):
TEMS Host Name (Default is: rhea00b00):

Network Protocol [ip, sna, ip.pipe or ip.spipe] (Default is: ip.pipe):

     Now choose the next protocol number from one of these:
     - ip
     - sna
     - ip.spipe
     - 0 for none
Network Protocol 2 (Default is: 0):
IP.PIPE Port Number (Default is: 1918):
Enter name of KDC_PARTITION (Default is: null):

Configure connection for a secondary TEMS? [1=YES, 2=NO] (Default is: 2):
Enter Optional Primary Network Name or 0 for "none" (Default is: 0):
Agent configuration completed...
```

O exemplo anterior cria uma instância do agente chamada “inst1” e atualiza as configurações Java Home. Os servidores de contêiner do eXtreme Scale são configurados, mas o serviço de catálogo não é configurado.

Nota: O procedimento anterior cria um arquivo de texto no seguinte formato no diretório: `<ITM_install>/config/<host>_xt_<instance name>.cfg`.

Exemplo: `rhea00b02_xt_inst1.cfg`

É recomendado editar esse arquivo com seu editor de texto simples escolhido. A seguir há um exemplo de conteúdo desse arquivo:

```
INSTANCE=inst2 [        SECTION=KQZ_JAVA [ { JAVA_HOME=/opt/OG61/java }  { JAVA_TRACE_LEVEL=ERROR }  ]
SECTION=KQZ_JMX_CONNECTION_SECTION [ { KQZ_JMX_CONNECTION_PROPERTY=KQZ_JMX_JSR160_JSR160 }  ]
SECTION=KQZ_JMX_JSR160_JSR160 [ { KQZ_JMX_JSR160_JSR160_CLASS_PATH_TITLE= }
{ KQZ_JMX_JSR160_JSR160_SERVICE_URL=service:jmx:rmi:///jndi/rmi://localho
st:port/objectgrid/MBeanServer }  { KQZ_JMX_JSR160_JSR160_CLASS_PATH_SEPARATOR= }  ]
SECTION=OGS:rhea00b02_c1 [ { KQZ_JMX_JSR160_JSR160_SERVICE_URL=service:jmx:
rmi:///jndi/rmi://localhost:15001/objectgrid/MBeanServer }  ]
SECTION=OGS:rhea00b02_c0 [ { KQZ_JMX_JSR160_JSR160_SERVICE_URL=service:jmx:
rmi:///jndi/rmi://localhost:15002/objectgrid/MBeanServer }  ]
SECTION=OGS:rhea00b02_c2 [ { KQZ_JMX_JSR160_JSR160_SERVICE_URL=service:jmx:
rmi:///jndi/rmi://localhost:15002/objectgrid/MBeanServer }  ]]
```

A seguir há um exemplo que mostra uma configuração de uma implementação do WebSphere Application Server:

```
rhea00b02 # ./itmcmd config -A xt
Agent configuration started...
Enter instance name (default is: ):  inst1
Edit "Monitoring Agent for
WebSphere eXtreme Scale"
settings? [ 1=Yes, 2=No ] (default is: 1): 1
Edit 'Java' settings? [ 1=Yes, 2=No ] (default is: 1): 1
Java home (default is:
C:\Program Files\IBM\Java50): /opt/WAS61/java
Java trace level [ 1=Error,
2=Warning, 3=Information, 4=Minimum Debug, 5=Medium Debug, 6=Maximum
Debug,
    7=All ] (default is: 1):
JVM arguments (default is: ):
Edit 'Connection' settings? [ 1=Yes, 2=No ] (default is: 1):
```

```
MBean server connection type [ 1=JSR-160-Compliant Server, 2=WebSphere Application Server version 6.0,
3=WebSphere Application Server version 6.1, 4=WebSphere Application Server version 7.0 ] (default is: 1): 4
Edit 'WebSphere Application Server version 7.0' settings? [ 1=Yes, 2=No ] (default is: 1):WAS user ID (default is: ):
Enter WAS password (default is: ):
Re-type : WAS password (default is: ):
WAS host name (default is: localhost): rhea00b02
WAS port (default is: 2809):
WAS connector protocol [ 1=rmi, 2=soap ] (default is: 1):
WAS profile name (default is: ): default
----------------------------------------
WAS Class Path Information
WAS base paths (default is: C:\Program Files\IBM\WebSphere\AppServer;/opt/IBM/WebSphere/AppServer): /opt/WAS61
WAS class path (default is: runtimes/com.ibm.ws.admin.client_6.1.0.jar;runtimes/com.ibm.ws.ejb.thinclient_7.0.0.jar):
WAS JAR directories (default is: lib;plugins):
Edit 'WebSphere eXtreme Scale Grid Servers' settings? [ 1=Yes, 2=No ] (default is: 1):
No 'WebSphere eXtreme Scale Grid Servers' settings available?
Edit 'WebSphere eXtreme Scale Grid Servers' settings, [1=Add, 2=Edit, 3=Del, 4=Next, 5=Exit] (default is: 4): 1
WebSphere eXtreme Scale Grid Servers (default is: ): rhea00b02
JMX service URL (default is: service:jmx:rmi:///jndi/rmi://localhost:<port>/objectgrid/MBeanServer):

'WebSphere eXtreme Scale Grid Servers' settings:
WebSphere eXtreme Scale
Grid Servers=rhea00b02
Edit 'WebSphere eXtreme Scale Grid Servers' settings, [1=Add, 2=Edit, 3=Del, 4=Next, 5=Exit] (default is: 4): 5
Edit 'WebSphere eXtreme Scale Catalog Service' settings? [ 1=Yes, 2=No ] (default is: 1):  2
Will this agent connect to a TEMS? [1=YES, 2=NO] (Default is: 1):
TEMS Host Name (Default is: rhea00b02):

Network Protocol [ip, sna, ip.pipe or ip.spipe] (Default is: ip.pipe):

     Now choose the next protocol number from one of these:
     - ip
     - sna
     - ip.spipe
     - 0 for none
Network Protocol 2 (Default is: 0):
IP.PIPE Port Number (Default is: 1918):
Enter name of KDC_PARTITION (Default is: null):

Configure connection for a secondary TEMS? [1=YES, 2=NO] (Default is: 2):
Enter Optional Primary Network Name or 0 for "none" (Default is: 0):
Agent configuration completed...
rhea00b02 #
```

Para implementações do WebSphere Application Server, não é necessário criar vários subnós. O agente do eXtreme Scale se conecta como agente de nó para reunir todas as informações a partir dos servidores de aplicativos para os quais ele é responsável.

`SECTION=CAT` significa uma linha de serviço de catálogo, enquanto `SECTION=OGS` significa uma linha de configuração do servidor eXtreme Scale.

5. Configure a porta JMX para todos os servidores de contêiner do eXtreme Scale.

   Quando os servidores de contêiner do eXtreme Scale forem iniciados sem especificar o argumento **-JMXServicePort**, um servidor MBean será designado a uma porta dinâmica. O agente precisa saber antecipadamente com qual porta JMX ele se comunicará. O agente não trabalha com portas dinâmicas.

   Ao iniciar os servidores, é necessário especificar o argumento `-JMXServicePort <port_number>` ao iniciar o servidor eXtreme Scale usando o comando `startOgServer.sh | .bat`. Executar esse comando garante que o servidor JMX dentro do processo atenda em uma porta estática predefinida.

   Para os exemplos anteriores em uma instalação UNIX, dois servidores eXtreme Scale precisarão ser iniciados com as portas configuradas:

   a. "-JMXServicePort" "15000" (para rhea00b02_c0)

   b. "-JMXServicePort" "15001" (para rhea00b02_c1)

   a. Inicie o agente eXtreme Scale.

      Supondo que a instância `inst1` foi criada, como no exemplo anterior, emita os seguintes comandos.

1) `cd <ITM_install>/bin`
2) `itmcmd agent -o inst1 start xt`

b. Pare o agente do eXtreme Scale.

Supondo que a instância "inst1" foi criada, como no exemplo anterior, emita os seguintes comandos.

1) `cd <ITM_install>/bin`
2) `itmcmd agent -o inst1 stop xt`

6. Ative Estatísticas para todos os servidores de contêiner do eXtreme Scale.

O agente usa MBeans de estatísticas do eXtreme Scale para registrar estatísticas. A especificação de estatísticas do eXtreme Scale deve ser ativada usndo um dos métodos a seguir.

- Configure propriedades do servidor para ativar todas as estatísticas quando os servidores de contêiner foram iniciados: all=enabled.
- Use o utilitário de amostra xsadmin para ativar estatísticas para todos os contêineres ativos usando os parâmetros -setstatsspec all=enabled.

Resultados

Após todos os servidores serem configurados e iniciados, os dados do MBeans são exibidos no console do IBM Tivoli Portal. As áreas de trabalho predefinidas mostram gráficos e métricas de dados em cada nível de nó.

As seguintes áreas de trabalho são definidas: nó de **Servidores de Grade do eXtreme Scale** para todos os nós monitorados.

- Visualização Transações do eXtreme Scale
- Visualização Shard Primário do eXtreme Scale
- Visualização Memória do eXtreme Scale
- Visualização ObjectMap do eXtreme Scale

Também é possível configurar suas próprias áreas de trabalho. Para obter mais informações, consulte as informações sobre a customização das áreas de trabalho no IBM Tivoli Monitoring Information Center.

Monitorando Aplicativos eXtreme Scale com o CA Wily Introscope

O CA Wily Introscope é um produto de gerenciamento de terceiro que pode ser usado para detectar e diagnosticar problemas de desempenho em ambientes de aplicativos corporativos. O eXtreme Scale inclui detalhes sobre a configuração do CA Wily Introscope para fazer introspecção de partes selecionadas do tempo de execução do eXtreme Scale para visualizar e validar rapidamente os aplicativos do eXtreme Scale. O CA Wily Introscope funciona eficientemente para as implementações do WebSphere Application Server e independentes.

Visão Geral

Para monitorar aplicativos eXtreme Scale com o CA Wily Introscope, é necessário colocar as configurações nos arquivos ProbeBuilderDirective (PBD) que fornecem acesso às informações de monitoramento para o eXtreme Scale.

Atenção: Os pontos de instrumentação para o Introscope podem ser alterados com cada fix pack ou release. Ao instalar um novo fix pack ou release, consulte a documentação para saber se há alguma mudança nos pontos de instrumentação.

é possível configurar os arquivos ProbeBuilderDirective (PBD) do CA Wily Introscope para monitorar seus aplicativos eXtreme Scale. O CA Wily Introscope é um produto de gerenciamento de aplicativos com o qual é possível detectar, selecionar e diagnosticar proativamente problemas de desempenho nos seus ambientes complexos, compostos e de aplicativos da Web.

Configurações de Arquivos PBD para Monitoramento do Serviço de Catálogo

É possível utilizar uma ou mais das seguintes configurações em seu arquivo PBD para monitorar o serviço de catálogo.

```
TraceOneMethodOfClass: com.ibm.ws.objectgrid.hamanager.HAControllerImpl
  changeDefinedCompleted
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.hamanager.HAControllerImpl
  viewChangeCompleted
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.hamanager.HAControllerImpl
  viewAboutToChange
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeat
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatCluster
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatCurrentLeader
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatDeadServer
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatNewLeader
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatNewServer
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.catalog.placement.
  PlacementServiceImpl
importRouteInfo BlamePointTracerDifferentMethods
  "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.catalog.placement.
  PlacementServiceImpl heartbeat
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.catalog.placement.
  PlacementServiceImpl joinPlacementGroup
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
  TraceOneMethodOfClass:
com.ibm.ws.objectgrid.catalog.placement.PlacementServiceImpl
  classifyServer
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.catalog.placement.
  BalanceGridEventListener shardActivated
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.catalog.placement.
  BalanceGridEventListener shardDeactivate
BlamePointTracerDifferentMethods "OGcatalog|{classname}|{method}"
```

Classes para monitoramento do serviço de catálogo

HAControllerImpl

A classe HAControllerImpl manipula eventos de ciclo de vida e feedback

do grupo principal. É possível monitorar esta classe para obter uma indicação da estrutura e das alterações do grupo principal.

ServerAgent
A classe ServerAgent é responsável pela comunicação de eventos do grupo principal com o serviço de catálogo. É possível monitorar as diversas chamadas de pulsação para marcar eventos graves.

PlacementServiceImpl
A classe PlacementServiceImpl coordena os contêineres. É possível utilizar os métodos nesta classe para monitorar eventos de junção e disposição do servidor.

BalanceGridEventListener
A classe BalanceGridEventListener controla a liderança de catálogo. É possível monitorar esta classe para obter uma indicação de qual serviço de catálogo está atualmente atuando como o líder.

Configurações de Arquivos PBD para Monitoramento de Contêineres

É possível utilizar uma ou mais das seguintes configurações em seu arquivo PBD para monitorar os contêineres.

```
TraceOneMethodOfClass: com.ibm.ws.objectgrid.ShardImpl processMessage
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.plugins.
  CommittedLogSequenceListenerProxy applyCommitted
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.plugins.
  CommittedLogSequenceListenerProxy sendApplyCommitted
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.map.BaseMap evictMapEntries
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.checkpoint.
  CheckpointMapImpl$CheckpointIterator activateListener
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.hamanager.HAControllerImpl
  changeDefinedCompleted
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.hamanager.HAControllerImpl
  viewChangeCompleted
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.hamanager.HAControllerImpl
  viewAboutToChange
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  batchProcess
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeat
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatCluster
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatCurrentLeader
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatDeadServer
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatNewLeader
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
```

```
TraceOneMethodOfClass: com.ibm.ws.objectgrid.container.ServerAgent
  heartbeatNewServer
BlamePointTracerDifferentMethods "OGcontainer|{classname}|{method}"
```

Classes para monitoramento dos contêineres

ShardImpl
A classe ShardImpl possui o método processMessage. O método processMessage é o método para pedidos do cliente. Com este método, é possível obter contagens de tempo de resposta e de pedidos do lado do servidor. Ao observar as contas em todos os servidores e monitorar a utilização do heap, é possível determinar se a grade está equilibrada.

CheckpointIterator
A classe CheckpointIterator possui a chamada de método activateListener que coloca primários no modo peer. Quando os primárias são colocadas no modo peer, a réplica é atualizada com o primário após a conclusão do método. Quando uma réplica está se regenerando a partir de um primário completo, esta operação pode levar um período estendido de tempo. O sistema não é totalmente recuperado até a conclusão da operação, portanto, é possível utilizar esta classe para monitorar o progresso da operação.

CommittedLogSequenceListenerProxy
A classe CommittedLogSequenceListenerProxy possui dois métodos de interesse. O método applyCommitted é executado para cada transação e o sendApplyCommitted é executado enquanto a réplica está executando pull de informações. A proporção de com que frequência estes dois métodos são executados pode fornecer a você alguma indicação de quão bem a réplica é capaz de continuar com o primário.

Configurações de Arquivos PBD para Monitoramento dos Clientes

É possível utilizar uma ou mais das seguintes configurações em seu arquivo PBD para monitorar os clientes.

```
TraceOneMethodOfClass: com.ibm.ws.objectgrid.client.ORBClientCoreMessageHandler
 sendMessage
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.corba.cluster.ClusterStore
  bootstrap
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.corba.cluster.ClusterStore
  epochChangeBootstrap
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.map.BaseMap evictMapEntries
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.cluster.orb.routing.
  SelectionServiceImpl routeFailed
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.cluster.orb.routing.
  SelectionServiceImpl routeFailed
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.SessionImpl getMap
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TraceOneMethodOfClass: com.ibm.ws.objectgrid.ObjectGridImpl getSession
BlamePointTracerDifferentMethods "OGclient|{classname}|{method}"
TurnOn: ObjectMap
SetFlag: ObjectMap
IdentifyClassAs: com.ibm.ws.objectgrid.ObjectMapImpl ObjectMap
TraceComplexMethodsifFlagged: ObjectMap BlamePointTracerDifferentMethods
"OGclient|{classname}|{method}"
```

Classes para monitoramento de clientes

ORBClientCoreMessageHandler
A classe ORBClientCoreMessageHandler é responsável por enviar pedidos de aplicativos para os contêineres. É possível monitorar o método sendMessage para o tempo de resposta do cliente e o número de pedidos.

ClusterStore
A classe ClusterStore contém as informações de roteamento no lado do cliente.

BaseMap
A classe BaseMap possui o método evictMapEntries que é chamado quando o evictor deseja remover entradas do mapa.

SelectionServiceImpl
A classe SelectionServiceImpl toma as decisões de roteamento. Se o cliente está tomando decisões de failover, é possível utilizar esta classe para ver as ações que são concluídas a partir das decisões.

ObjectGridImpl
A classe ObjectGridImpl possui o método getSession que é possível monitorar para ver o número de pedidos para este método.

Monitorando o eXtreme Scale com o Hyperic HQ

O Hyperic HQ é uma solução de monitoramento de terceiro que está disponível gratuitamente como uma solução de software livre ou como um produto corporativo. O WebSphere eXtreme Scale inclui um plug-in que permite que os agentes do Hyperic HQ descubram os servidores de contêiner do eXtreme Scale e relatem e agreguem as estatísticas usando os beans de gerenciamento do eXtreme Scale. É possível usar o Hyperic HQ para monitorar as implementações do eXtreme Scale independentes.

Antes de Iniciar

- Esse conjunto de instruções destina-se para o Hyperic Versão 4.0. Se você tiver uma versão mais recente do Hyperic, consulte a Documentação do Hyperic para obter informações sobre os nomes de caminho e como iniciar os agentes e servidores.
- Download do servidor Hyperic e das instalações do agente. Uma instalação do servidor deve estar em execução. Para detectar todos os servidoreseXtreme Scale, um agente Hyperic deve estar em execução em cada máquina na qual um servidor eXtreme Scale está em execução. Consulte o Web site Hyperic para obter informações de download e suporte para documentação.
- É necessário ter acesso aos arquivos `objectgrid-plugin.xml` e `hqplugin.jar`. Esses arquivos estão no diretório `objectgridRoot/hyperic/etc`.

Sobre Esta Tarefa

Ao integrar o eXtreme Scale com o software de monitoramento Hyperic HQ, poderá monitorar e exibir graficamente as métricas sobre o desempenho do seu ambiente. Configure essa integração ao usar uma implementação de plug-in em cada agente.

Procedimento

1. Inicie os servidores eXtreme Scale. O plug-in Hyperic procura nos processos locais para se conectar ao Java Virtual Machines que está executando o eXtreme Scale. Para se conectar corretamente com o Java Virtual Machines, cada servidor deve ser iniciado com a opção **-jmxServicePort**. Para obter

informações sobre como iniciar os servidores com a opção **-jmxServicePort**, consulte "Script startOgServer" na página 334.

2. Coloque os arquivos `extremescale-plugin.xml` e `wxshyperic.jar` nos diretórios de plug-in do servidor e do agente apropriados na configuração Hyperic. Para integração com o Hyperic, ambas as instalações do agente e do servidor devem ter acesso ao plug-in e aos arquivos Java archive (JAR). Embora o servidor possa trocar as configurações dinamicamente, é necessário concluir a integração antes de iniciar qualquer um dos agentes.
   a. Coloque o arquivo `extremescale-plugin.xml` no diretório `plugin` do servidor, que está no seguinte local:

      `hyperic_home/server_home/hq-engine/server/default/deploy/hq.ear/hq-plugins`
   b. Coloque o arquivo `extremescale-plugin.xml` no diretório `plugin` do agente, que está no seguinte local:

      `agent_home/bundles/gent-4.0.2-939/pdk/plugins`
   c. Coloque o arquivo `wshyperic.jar` no diretório `lib` do agente, que está no seguinte local:

      `agent_home/bundles/gent-4.0.2-939/pdk/lib`
3. Configure o agente. O arquivo `agent.properties` atua como um ponto de configuração para o tempo de execução do Agente. Essa propriedade está no diretório *agent_home*`/conf`. As seguintes chaves são opcionais, mas importantes para o plug-in do eXtreme Scale:
   - `autoinventory.defaultScan.interval.millis=<`*time_in_milliseconds*`>`

     Configura o intervalo em milissegundos entre as descobertas do Agente.
   - `log4j.logger.org.hyperic.hq.plugin.extremescale.XSServerDetector=DEBUG`

     : Ativa as instruções de depuração detalhadas a partir do plug-in eXtreme Scale.
   - `username=<`*username*`>`: Configura o nome do usuário do Java Management Extensions (JMX) se a segurança estiver ativada.
   - `password=<`*password*`>`: Configura a senha JMX se a segurança estiver ativada.
   - `sslEnabled=<`*true*`|`*false*`>`: Informa se o plug-in deve usar ou não o Secure Sockets Layer (SSL). Por padrão, o valor é `false`.
   - `trustPath=<`*path*`>`: Configura um caminho de confiança para a conexão SSL.
   - `trustType=<`*type*`>`: Configura um tipo de confiança para a conexão SSL.
   - `trustPass=<`*password*`>`: Configura uma senha de confiança para a conexão SSL.
4. Inicie a descoberta do agente. Os agentes Hyperic enviam informações de descobertas e de métricas para o servidor. Utilize o servidor para customizar visualizações de dados e de objetos de inventário lógicos do grupo para gerar informações úteis. Depois que o servidor estiver disponível, é necessário executar o script de ativação ou iniciar o serviço do Windows para o agente:
   - Linux `agent_home/bin/hq-agent.sh start`
   - Windows Inicie o agente com o serviço do Windows.

   Depois de iniciar os agentes, os servidores são detectados e os grupos configurados. É possível efetuar login no console do servidor e escolher quais recursos incluir no banco de dados de inventário para o servidor. Por padrão, o console do servidor está no seguinte URL: `http://<`*server_host_name*`>:7080/`
5. As estatísticas devem ser ativadas para Hyperic para coletar dados estatísticos.

Use a ação de controle **SetStatsSpec** no console do Hyperic para o eXtreme Scale. Navegue para o recurso e, em seguida, use a lista suspensa **Ação de Controle** na página tabulada **Controle** para especificar uma configuração `SetStatsSpec` com `ALL=enabled` na caixa de texto **Argumentos de Controle**.

Os servidores de catálogos não são detectados pelo filtro configurado no console do Hyperic. Consulte as informações sobre a propriedade **statsSpec** no "Arquivo de Propriedades do Servidor" na página 183, que ativa as estatísticas assim que os contêineres forem iniciados. Várias opções para ativar estatísticas são descritas no "Monitoramento com Beans Gerenciados (MBeans)" na página 400 e no "Monitorando com o Utilitário de Amostra xsAdmin" na página 385

6. Monitorar servidores com o console do Hyperic. Depois que os servidores forem incluídos no modelo de inventário, os serviços não serão mais necessários.
   - **Visualização Painel**: Quando você visualizou os eventos de detecção de recursos, efetuou login na visualização de painel principal. A visualização de painel é uma visualização genérica que atua como um centro de mensagens que pode ser customizado. É possível exportar gráficos ou objetos de inventário para este painel principal.
   - **Visualização Recursos**: É possível consultar e visualizar o modelo de inventário inteiro a partir desta página. Após os serviços terem sido incluídos, é possível visualizar todos os servidores eXtreme Scale adequadamente rotulados e listados juntos sob a seção Servidores. É possível clicar nos servidores individuais para visualizar as métricas básicas.
7. Visualize o inventário do servidor inteiro na página Visualização de Recursos. Nesta página, será possível selecionar vários servidores ObjectGrid e agrupá-los. Depois de agrupar um conjunto de recursos, as métricas comuns podem ser exibidas em gráfico para mostrar as sobreposições e diferenças entre os membros de grupo. Para exibir uma sobreposição, selecione as métricas na exibição do Grupo de Servidor. Em seguida, a métrica é exibida na área de gráfico. Para exibir uma sobreposição para todos os membros de grupo, clique no nome da métrica sublinhada. É possível exportar qualquer gráfico, visualizações de nó e sobreposições comparativas no painel principal com o menu **Ferramentas**.

Capítulo 10. Ajuste e Desempenho

Lista de Verificação Operacional

Use a lista de verificação operacional para preparar o ambiente para implementar o WebSphere eXtreme Scale.

Tabela 27. Lista de Verificação Operacional

Item de Lista de Verificação	Para obter informações adicionais

Se você estiver usando o AIX, ajuste as seguintes configurações do sistema operacional:
TCP_KEEPINTVL
A configuração TCP_KEEPINTVL faz parte de um protocolo de soquete keep-alive que permite a detecção de uma interrupção de rede. A propriedade especifica o intervalo entre os pacotes que são enviados para validar a conexão. Ao usar o WebSphere eXtreme Scale, configure o valor para `10`. Para verificar a configuração atual, execute o seguinte comando:
`# no –o tcp_keepintvl`
Para alterar a configuração atual, execute o seguinte comando:
`# no –o tcp_keepintvl=10`
A configuração TCP_KEEPINTVL está em unidades de meio segundo.
TCP_KEEPINIT
A configuração TCP_KEEPINIT faz parte de um protocolo de soquete keep-alive que permite a detecção de uma interrupção de rede. A propriedade especifica o valor de tempo limite inicial para a conexão TCP. Ao usar o WebSphere eXtreme Scale, configure o valor para `40`. Para verificar a configuração atual, execute os seguintes comandos:
`# no –o tcp_keepinit`
Para alterar a configuração atual, execute o seguinte comando:
`# no –o tcp_keepinit=40`
A configuração TCP_KEEPINIT está em unidades de meio segundo.
	• Para obter informações de ajusta do AIX, consulte Ajustando Sistemas AIX.

Atualize o arquivo `orb.properties` para modificar o comportamento de transporte da grade. O arquivo `orb.properties` está no diretório `java/jre/lib`.	"Arquivo de Propriedades ORB" na página 192

Tabela 27. Lista de Verificação Operacional (continuação)

Item de Lista de Verificação	Para obter informações adicionais
Use parâmetros no script startOgServer. Em particular, use os seguintes parâmetros: • Defina as configurações de heap com o parâmetro **-jvmArgs**. • Defina o caminho de classe e as propriedades do aplicativo com o parâmetro **-jvmArgs**. • Defina os parâmetros **-jvmArgs** para configurar o monitoramento de agente. **Configurações de Porta** O WebSphere eXtreme Scale precisa abrir portas para comunicações para alguns transportes. Essas portas são todas definidas dinamicamente. Porém, se um firewall estiver em uso entre os contêineres, será necessário especificar as portas. Use as seguintes informações sobre as portas: **Porta Listener** É possível usar o argumento **-listenerPort** para especificar a porta que é usada para comunicação entre processos. **Porta do grupo principal** É possível usar o argumento **-haManagerPort** para especificar a porta que é usada para detecção de falha. Este argumento é o mesmo que peerPort. Observe que os grupos principais não precisam se comunicar entre as zonas, portanto, pode não ser necessário configurar essa porta se o firewall estiver aberto para todos os membros de uma única zona. **Porta de serviço JMX** É possível usar o argumento **-JMXServicePort** para especificar a porta que o serviço JMX deve usar. **Porta SSL** Transmitir `-Dcom.ibm.CSI.SSLPort=1234` como um argumento **-jvmArgs** configura a porta SSL para `1234`. A porta SSL é a porta protegida equivalente à porta listener. **Porta do cliente** Usada apenas no serviço de catálogo. É possível especificar esse valor com o argumento **-catalogServiceEndPoints**. O formato do valor desse parâmetro está no formato: `serverName:hostName:clientPort:peerPort`	"Script startOgServer" na página 334
Verifique se as configurações de segurança estão definidas corretamente: • Transporte (SSL) • Aplicativo (Autenticação e Autorização) Para verificar as configurações de segurança, é possível tentar usar um cliente malicioso para conectar-se com a sua configuração. Por exemplo, quando a configuração SSL necessária for definida, um cliente que possuir a configuração TCP_IP ou um cliente com o truststore incorreto não conseguirá se conectar ao servidor. Quando a autenticação é necessária, um cliente com nenhuma credencial, como um ID de usuário e senha, não conseguirá se conectar ao servidor. Quando a autorização é forçada, um cliente com nenhuma autorização de acesso não conseguira acessar os recursos do servidor.	"Integração de Segurança com Provedores Externos" na página 368
Escolha como você monitorará seu ambiente. • xsAdmin – As portas JMX dos servidores de catálogo precisam estar visíveis para a ferramenta XSAdmin. As portas do contêiner também precisam estar acessíveis para alguns comandos que reúnem informações a partir dos contêineres. • É possível escolher entre as seguintes ferramentas de monitoramento do fornecedor: – Tivoli Enterprise Monitoring Agent – CA Wily Introscope – Hyperic HQ	• "Monitorando com o Utilitário de Amostra xsAdmin" na página 385 • "Segurança do Java Management Extensions (JMX)" na página 366 • "Monitorando com o IBM Tivoli Enterprise Monitoring Agent para WebSphere eXtreme Scale" na página 408 • "Monitorando o eXtreme Scale com o Hyperic HQ" na página 418 • "Monitorando Aplicativos eXtreme Scale com o CA Wily Introscope" na página 414

Ajuste de Sistemas Operacionais e Rede

O ajuste da rede pode reduzir o atraso da pilha do Protocolo de Controle de Transmissões (TCP) por meio da mudança das configurações de conexão e melhorar o rendimento por meio da mudança dos buffers TCP.

Sistemas Operacionais

Um sistema Windows precisa do mínimo de ajuste enquanto que o sistema Solaris precisa do máximo de ajuste. As informações a seguir referem-se a cada sistema especificado e podem aumentar o desempenho do WebSphere eXtreme Scale. Você deve ajustar de acordo com sua carga de rede e do aplicativo.

Windows

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\
Tcpip\Parameters
MaxFreeTcbs = dword:00011940
MaxHashTableSize = dword:00010000
MaxUserPort = dword:0000fffe
TcpTimedWaitDelay = dword:0000001e
```

Solaris

```
ndd -set /dev/tcp tcp_time_wait_interval 60000
fndd -set /dev/tcp tcp_keepalive_interval 15000
ndd -set /dev/tcp tcp_fin_wait_2_flush_interval 67500
ndd -set /dev/tcp tcp_conn_req_max_q 16384
ndd -set /dev/tcp tcp_conn_req_max_q0 16384
ndd -set /dev/tcp tcp_xmit_hiwat 400000
ndd -set /dev/tcp tcp_recv_hiwat 400000
ndd -set /dev/tcp tcp_cwnd_max 2097152
ndd -set /dev/tcp tcp_ip_abort_interval 20000
ndd -set /dev/tcp tcp_rexmit_interval_initial 4000
ndd -set /dev/tcp tcp_rexmit_interval_max 10000
ndd -set /dev/tcp tcp_rexmit_interval_min 3000
ndd -set /dev/tcp tcp_max_buf 4194304
```

AIX

```
/usr/sbin/no -o tcp_sendspace=65536
/usr/sbin/no -o tcp_recvspace=65536
/usr/sbin/no -o udp_sendspace=65536
/usr/sbin/no -o udp_recvspace=65536
/usr/sbin/no -o somaxconn=10000
/usr/sbin/no -o tcp_nodelayack=1
/usr/sbin/no –o tcp_keepinit=40
/usr/sbin/no –o tcp_keepintvl=10
```

LINUX

```
sysctl -w net.ipv4.tcp_timestamps=0
sysctl -w net.ipv4.tcp_tw_reuse=1
sysctl -w net.ipv4.tcp_tw_recycle=1
sysctl -w net.ipv4.tcp_fin_timeout=30
sysctl -w net.ipv4.tcp_keepalive_time=1800
sysctl -w net.ipv4.tcp_rmem="4096 87380 8388608"
sysctl -w net.ipv4.tcp_wmem="4096 87380 8388608"
sysctl -w net.ipv4.tcp_max_syn_backlog=4096
```

HP-UX

```
ndd -set /dev/tcp tcp_ip_abort_cinterval 20000
```

Planejamento para Portas de Rede

O WebSphere eXtreme Scale é um cache distribuído que necessita de portas de abertura para se comunicar com o Object Request Broker (ORB) e a pilha do Protocolo de Controle de Transmissões (TCP) entre Java Virtual Machines e outras máquinas. Você deve planejar e controlar suas portas, principalmente em um ambiente de firewall, como quando você está utilizando um serviço de catálogo e contêineres em várias portas.

Domínio do Serviço de Catálogo

Um domínio do serviço de catálogo requer as seguintes portas para ser definido:

peerPort
Especifica a porta para o gerenciador de alta disponibilidade (HA) se comunicar entre servidores de catálogo de peer sobre uma pilha TCP.

clientPort
Especifica a porta para servidores de catálogo acessarem dados do serviço de catálogo.

JMXServicePort
Especifica qual porta o serviço Java Management Extensions (JMX) deve usar.

listenerPort
Define a porta listener ORB para contêineres e clientes se comunicarem com o serviço de catálogo por meio de ORB.

A forma como você define essas portas depende de se você está usando o modo independente ou se está iniciando os servidores de catálogo do eXtreme Scale em um ambiente do WebSphere Application Server:

- **Para o modo independente:**

  Use o comando startOgServer para especificar as portas anteriormente listadas com a opção no modo independente, conforme mostrado no seguinte exemplo:

  ```
  –catalogServiceEndPoints cs1:host1:clientPort:peerPort, cs2:host2:clientPort:peerPort,
  cs3:host3:clientPort:peerPort -listenerPort <orbPort> -JMXServicePort <jmxPort>
  ```

  Consulte “Iniciando o Serviço de Catálogo em um Ambiente Independente” na página 329 para obter mais informações sobre como iniciar o serviço de catálogo no modo independente.
- **Para um ambiente do WebSphere Application Server:**

  É possível definir um domínio do serviço de catálogo no console administrativo. Consulte “Criando Domínios do Serviço de Catálogo no WebSphere Application Server” na página 344 para obter mais informações.

Servidores de Contêineres

Os servidores de contêineres do WebSphere eXtreme Scale também precisam de várias portas para operar. Por padrão, o servidor de contêineres do eXtreme Scale gera sus porta do gerenciador HA e porta listener ORB automaticamente com portas dinâmicas. No ambiente de firewall, como é vantajoso planejar e controlar portas, as opções são fornecidas para iniciar os servidores de contêineres do eXtreme Scale com porta HAManager e porta listener ORB especificadas com uma opção no comando startOgServer, como mostra o exemplo a seguir:

```
-HaManagerPort <peerPort>
-listenerPort <orbPort>
```

O planejamento adequado do controle da porta é um benefício, mas existe uma dificuldade inerente no planejamento e gerenciamento dessas portas quando centenas de Java Virtual Machines são iniciadas em uma máquina. Qualquer conflito de porta causará uma falha da inicialização do servidor.

Quando a segurança está ativada, uma porta Secure Socket Layer (SSL) é uma inclusão necessária às portas listadas anteriormente. Usar `Dcom.ibm.CSI.SSLPort=<sslPort>` como um argumento **-jvmArgs** configura a porta SSL para `<sslPort>`. Leia mais sobre configurações de segurança para eXtreme Scale para ajudá-lo com o planejamento de portas.

Propriedades do ORB e Configurações do Descritor de Arquivo

As considerações de ajuste incluem propriedades do Object Request Broker (ORB) e configurações do descritor de arquivo.

Propriedades do ORB

O ORB é usado pelo WebSphere eXtreme Scale para se comunicar por meio de uma pilha TCP. O arquivo `orb.properties` necessário está no diretório `java/jre/lib`. Para um carga pesada de objetos grandes, ative a fragmentação de ORB especificando as seguintes configurações:

```
com.ibm.CORBA.FragmentSize=<tamanho correto>
```

Evite o crescimento de ThreadPool especificando as seguintes configurações:

```
com.ibm.CORBA.ThreadPool.IsGrowable=false
```

Configure os tempos limite para evitar excesso de encadeamentos em uma situação anormal especificando as seguintes configurações:

- com.ibm.CORBA.RequestTimeout
- com.ibm.CORBA.ConnectTimeout
- com.ibm.CORBA.FragmentTimeout

Descritor de Arquivo

Sistemas UNIX e Linux possuem um limite para a quantidade de arquivos abertos permitidos por processo. O sistema operacional especifica a quantidade de arquivos abertos permitidos. Se este valor for configurado com um valor muito baixo, um erro de alocação de memória ocorrerá no AIX, e muitos arquivos abertos serão registrados no log.

Na janela de terminal do sistema UNIX, configure este valor para um valor mais alto que o valor do sistema padrão. Para máquinas SMP maiores com clones, configure para ilimitado.

Para configurações AIX, defina este valor como `-1` (ilimitado) com o comando `ulimit -n -1`.

Para configurações Solaris, defina este valor como `16384` com o comando `ulimit -n 16384`.

Para exibir o valor atual, use o comando `ulimit –a`.

Non-blocking I/O (NIO) Com o ORB

Atualmente, é possível executar com NIO ou ChannelFramework em cenários independentes do WebSphere eXtreme Scale. Para executar com a Non-blocking I/O (NIO) no ORB IBM, é necessário configurar TransportMode para ChannelFramework do ORB IBM. Por padrão, o ORB IBM é executado no modo Conectável. O WebSphere eXtreme Scale fornece uma propriedade do servidor para configurar TransportMode para ChannelFramework

Importante:

O WebSphere Application Server suporta o modo Conectável apenas em releases atuais. Quando o WebSphere eXtreme Scale é executado integrado com o WebSphere Application Server, ele deve seguir o modo Conectável. Como o

WebSphere eXtreme Scale também usa o SSL/Transport Security do WebSphere Application Server, ele também suporta atualmente apenas o modo Conectável.

Quando Usar NIO

Aqui está uma pequena seção em um conceito.

Ativando a NIO ou ChannelFramework no ORB

O WebSphere eXtreme Scale fornece uma propriedade de servidor para configurar o TransportMode para ChannelFramework para ativar a non-blocking I/O (NIO) no ORB.

Antes de Iniciar

Localize o arquivo de propriedades do servidor existente ou cria um arquivo de propriedades do servidor. É possível ativar ChannelFramework no serviço de catálogo e nos servidores de contêiner. Para obter mais detalhes, consulte o Arquivo de propriedades do servidor.

Sobre Esta Tarefa

Atualmente, é possível executar com NIO ou ChannelFramework em cenários independentes do WebSphere eXtreme Scale. Para executar com a Non-blocking I/O (NIO) no ORB IBM, você deve configurar o TransportMode do ORB IBM para ChannelFramework. Por padrão, o ORB IBM é executado no modo Conectável. O WebSphere eXtreme Scale fornece uma propriedade de servidor para configurar o TransportMode para ChannelFramework.

Importante:

O WebSphere Application Server suporta o modo Conectável apenas em releases atuais. Quando o WebSphere eXtreme Scale é executado integrado com o WebSphere Application Server, ele deve seguir o modo Conectável. Como o WebSphere eXtreme Scale também usa o SSL/Transport Security do WebSphere Application Server, ele também suporta atualmente apenas o modo Conectável.

Procedimento

1. Inclua a propriedade enableChannelFramework=true no seu arquivo de propriedades do servidor.
2. Certifique-se de que o arquivo de propriedades do servido não contradiga o arquivo de propriedades do ORB.

   Se o arquivo de propriedades do servidor ativar o ChannelFramework TransportMode, mas o TransportMode for configurado como Conectável no arquivo orb.properties, o servidor não substituirá a configuração de orb.properties. Você verá uma mensagem de aviso no log de que há duas configurações. Para permitir que a propriedade enableChannelFramework=true tenha efeito, ajuste as propriedades que indicam que TransportMode está configurado como Conectável: altere com.ibm.CORBA.TransportMode=Pluggable para ChannelFramework ou remova a propriedade.
3. Forneça o arquivo de propriedades do servidor atualizado no serviço de catálogo ou na inicialização do servidor de contêiner. Para obter mais detalhes sobre como usar os arquivos de propriedades do servidor para iniciar um servidor, consulte Arquivo de propriedades do servidor.

Resultados

Quando um serviço de catálogo ou servidor de contêiner usar o channelFramework TransportMode, ele imprimirá a seguinte mensagem para o log.

```
CWOBJ0052I: A propriedade IBM ORB TransportMode foi configurada para ChannelFramework
```

Se você vir a seguinte mensagem no log, examine suas propriedades do ORB, conforme descrito anteriormente.

```
CWOBJ0055W: A propriedade IBM ORB TransportMode foi configurada para ChannelFramework
no arquivo de propriedades do servidor, mas o arquivo orb.properties existente já
tinha um TransportMode configurado. O TransportMode não será substituído.
```

Observe que quando você ativa ChannelFramework, o valor máximo para ServerSocketQueueDepth é 512. Se a configuração orb.properties ServerSocketQueueDepth for maior que 512, o servidor automaticamente configurará orb.properties ServerSocketQueueDepth para 512 e o notificará imprimindo uma mensagem informativa para o log. Nenhuma ação é necessária.

```
CWOBJ0053I: A propriedade IBM ORB ServerSocketQueueDepth foi configurada para
512 para ser executada
corretamente com o ChannelFramework TransportMode.
```

Ajuste da JVM para o WebSphere eXtreme Scale

O ajuste da Java Virtual Machine (JVM) pode resultar em uma melhoria significativa da sua implementação do WebSphere eXtreme Scale.

Na maioria dos casos, o WebSphere eXtreme Scale requer pouca ou nenhuma configuração especial da JVM. Se você tiver uma grande quantidade de objetos sendo armazenados em um WebSphere eXtreme Scale, ajuste o tamanho de heap para um nível apropriado para evitar a execução sem memória.

Plataformas

Os testes de desempenho ocorreram principalmente nas caixas AIX (32 processadores), Linux (4 processadores) e Windows (8 processadores). Com caixas AIX high-end, cenários fortemente multiencadeados podem ser armazenados e os pontos de contenção podem ser identificados e corrigidos.

Requisitos do Object Request Broker (ORB)

O IBM SDK inclui uma implementação IBM ORB que foi testada com o WebSphere Application Server e o WebSphere eXtreme Scale. Para facilitar o processo de suporte, use um JVM fornecido pela IBM. Outras implementações de JVM utilizam um ORB diferente. O IBM ORB é fornecido apenas pronto para uso com as Java virtual machines fornecidas pela IBM. O WebSphere eXtreme Scale requer um ORB em funcionamento para operar. É possível utilizar o WebSphere eXtreme Scale com ORBs de terceiros, mas se ocorrer um problema no ORB, será necessário entrar em contato com o fornecedor do ORB para obter suporte. A implementação do IBM ORB é compatível com Java virtual machines de terceiros e pode ser substituído, se necessário.

Coleta de Lixo

O WebSphere eXtreme Scale cria objetos temporários associados a cada transação, como pedido e reposta e sequência de log. Como esses objetos afetam a eficiência da coleta de lixo, o ajuste da coleta de lixo é fundamental.

Para a máquina virtual IBM para Java, use o coletor optavgpause para cenários com alta taxa de atualização (100% das transações modificam entradas). O coletor gencon funciona bem melhor que o coletor optavgpause nos cenários em que os dados são atualizados relativamente com pouca frequência (10% do tempo ou menos). Experimente ambos os coletores para ver qual funciona melhor no seu cenário. Em caso de algum problema de desempenho, ative a coleta de lixo detalhada para verificar a porcentagem de tempo que está sendo consumida pela coleta. Ocorreram cenários em que 80% do tempo foi gasto na coleta de lixo até que um que o ajuste resolvesse o problema.

Para obter mais informações sobre a configuração da coleta de lixo, consulte Ajustando a IBM Virtual Machine para Java.

Desempenho da JVM

O WebSphere eXtreme Scale pode executar em diferentes versões de J2SE (Java 2 Platform, Standard Edition). O WebSphere eXtreme Scale Versão 6.1 suporta o J2SE Versão 1.4.2 e superior. Para produtividade e desempenho de desenvolvedor aprimorados, utilize o J2SE 5 ou mais recente para obter vantagem das anotações e da coleta de lixo aprimorada. O WebSphere eXtreme Scale funciona em Java virtual machines de 32 ou 64 bits.

O WebSphere eXtreme Scale é testado com um subconjunto das máquinas virtuais disponíveis, todavia, a lista suportada não é exclusiva. É possível executar WebSphere eXtreme Scale em qualquer Versão 1.4.2 ou posterior mas, se for identificado um problema na JVM, você deve entrar em contato com o fornecedor da JVM para obter suporte. Se possível, use a JVM a partir do tempo de execução do WebSphere em qualquer plataforma que o WebSphere Application Server suporta.

Para a maioria dos cenários nos quais o WebSphere eXtreme Scale é usado, o Java Platform Standard Edition 6 da JVM funciona melhor que o Edition 5 ou 1.4. O Java 2 Platform, Standard Edition Versão 1.4 tem um fraco desempenho, principalmente em cenários que usam o coletor gencon. A Java Platform Standard Edition 5 executa bem, as a Java Platform, Standard Edition 6 executa melhor.

Heaps Grandes

Quando o aplicativo precisa gerenciar uma grande quantidade de dados para cada partição, a coleta de lixo pode ser um fator. Na maioria das vezes, o desempenho de leitura é bom mesmo com heaps muito grandes (20 GB ou mais) quando um coletor geracional é utilizado. Contudo, depois que o heap de estabilidade é preenchido, ocorre uma pausa proporcional ao tamanho do heap ativo e ao número de processadores na máquina. Essa pausa pode ser grande em máquinas menores com grandes heaps.

O WebSphere eXtreme Scale suporta o WebSphere Real Time Java. Com o WebSphere Real Time Java, a resposta de processamento de transação do WebSphere eXtreme Scale é mais consistente e previsível e o impacto da coleta de lixo e do planejamento do encadeamento é enormemente minimizado. O impacto é reduzido ao nível no qual o tempo de desvio de resposta padrão é inferior a 10% da Java regular.

Consulte “Usando o WebSphere Real Time” na página 435 para obter informações adicionais.

Contagem de Encadeamentos

A contagem de encadeamentos depende de poucos fatores. Há um limite de quantos encadeamentos um único shard pode gerenciar. Um shard é uma instância de uma partição e pode ser primário ou de réplica. Com mais shards para cada JVM, poderá haver mais encadeamentos, com cada shard fornecendo mais caminhos simultâneos para os dados. Embora cada shard é simultâneo o máximo possível, essa simultaneidade ainda é limitada.

Ajuste da JVM

Você deve levar em conta vários aspectos específicos do ajuste da Java Virtual Machine (JVM) para melhorar o desempenho do WebSphere eXtreme Scale.

A recomendação é de heaps de 1 a 2 Gb com um JVM para cada 4 núcleos. Os tamanhos de heap dependem da natureza dos objetos que estão sendo armazenados nos servidores, o que é discutido posteriormente neste documento.

Recomendações de Tamanho de Heap e Coleta de Lixo

O melhor número de tamanho de heap depende de três fatores:

1. Quantidade de objetos ativos no heap.
2. Complexidade dos objetos ativos no heap.
3. Quantidade de núcleos disponíveis para a JVM.

Por exemplo, um aplicativo que armazena matrizes de 10 Kbytes pode executar um heap muito maior do que um aplicativo que usa gráficos complexos de POJOs.

Todas as JVMs modernas usam algoritmos de coleta de lixo paralelos, o que significa que usar mais núcleos pode reduzir as pausas na coleta de lixo. Assim, caixas com 8 núcleos serão coletadas mais rapidamente do que uma caixa com 4 núcleos.

Uso de Memória Real Contra a Especificação do Heap

Uma JVM com heap de 1 Gb usa aproximadamente 1,3 Gb de memória real. Em nosso laboratório, não conseguimos executar dez JVMs de 1 Gb em uma caixa com 16 Gb de RAM. Depois de os heaps JVM serem preenchidos com 800 MB adicionais, a caixa começou a efetuar a paginação.

Coleta de Lixo

Para JVMs IBM, use o coletor avgoptpause para cenários com alta taxa de atualização (100% de entradas de modificação de transações). O coletor gencon funciona muito melhor que o coletor avgoptpause nos cenários em que os dados são atualizados relativamente com pouca freqüência (10% do tempo ou menos). Experimente ambos os coletores para ver qual funciona melhor no seu cenário. Em caso de algum problema de desempenho, ative a coleta de lixo detalhada para verificar a porcentagem de tempo que está sendo consumida pela coleta. Ocorreram cenários em que 80% do tempo foi gasto na coleta de lixo até que um que o ajuste resolvesse o problema.

Desempenho da JVM

O WebSphere eXtreme Scale pode executar em diferentes versões de J2SE (Java 2 Platform, Standard Edition). O ObjectGrid Versão 6.1 suporta J2SE Versão 1.4.2 e

posterior. Para produtividade e desempenho de desenvolvedor aprimorados, utilize o J2SE 5 ou mais recente para obter vantagem das anotações e da coleta de lixo aprimorada. O ObjectGrid trabalha em JVMs de 32 ou 64 bits.

Os clientes do ObjectGrid Versão 6.0.2 podem se conectar a uma grade do ObjectGrid Versão 6.1. Utilize os clientes do ObjectGrid Versão 6.1 para o J2SE Versão 1.4.2 ou clientes melhores. O único motivo para utilizar um cliente do ObjectGrid Versão 6.0.2 é o suporte para o J2SE Versão 1.3.

O WebSphere eXtreme Scale é testado com um subconjunto das máquinas virtuais disponíveis, todavia, a lista suportada não é exclusiva. É possível executar WebSphere eXtreme Scale em qualquer Versão 1.4.2 ou posterior mas, se for identificado um problema na JVM, você deve entrar em contato com o fornecedor da JVM para obter suporte. Se possível, use a JVM a partir do tempo de execução do WebSphere em qualquer plataforma que o WebSphere Application Server suporta.

A Java Platform, Standard Edition 6 é a melhor JVM. A Java 2 Platform, Standard Edition, v 1.4 é executada de maneira deficiente, especialmente para cenários onde o coletor gencon faz uma diferença. A Java Platform Standard Edition 5 é executada bem, mas a Java Platform, Standard Edition 6 é executada melhor.

Ajuste de orb.properties

A recomendação é usar o seguinte arquivo orb.properties para produção. Em nosso laboratório, usamos este arquivo em grades de até 1500 JVMs. O arquivo orb.properties está na pasta lib do JRE que está sendo usado.

```
# IBM JDK properties for ORB
org.omg.CORBA.ORBClass=com.ibm.CORBA.iiop.ORB
org.omg.CORBA.ORBSingletonClass=com.ibm.rmi.corba.ORBSingleton

# WS Interceptors
org.omg.PortableInterceptor.ORBInitializerClass.com.ibm.ws.objectgrid.corba.ObjectGridInitializer

# WS ORB & Plugins properties
com.ibm.CORBA.ForceTunnel=never
com.ibm.CORBA.RequestTimeout=10
com.ibm.CORBA.ConnectTimeout=10

# Needed when lots of JVMs connect to the catalog at the same time
com.ibm.CORBA.ServerSocketQueueDepth=2048

# Clients and the catalog server can have sockets open to all JVMs
com.ibm.CORBA.MaxOpenConnections=1016

# Thread Pool for handling incoming requests, 200 threads here
com.ibm.CORBA.ThreadPool.IsGrowable=false
com.ibm.CORBA.ThreadPool.MaximumSize=200
com.ibm.CORBA.ThreadPool.MinimumSize=200
com.ibm.CORBA.ThreadPool.InactivityTimeout=180000

# No splitting up large requests/responses in to smaller chunks
com.ibm.CORBA.FragmentSize=0
```

Contagem de Encadeamentos

A contagem de encadeamentos depende de poucos fatores. Há um limite de quantos encadeamentos um único shard pode gerenciar. Com mais shards para cada JVM, pode haver mais encadeamentos e mais simultaneidade. Cada shard adicional fornece mais caminhos simultâneos para os dados. Cada shard é um concorrente possível mas, mesmo assim, existe um limite.

Configurando a Detecção de Failover

É possível configurar a quantidade de tempo entre verificações do sistema para servidores com falha com a configuração do intervalo de pulsação.

Sobre Esta Tarefa

A configuração de failover varia dependendo do tipo de ambiente que você está usando. Se você estiver utilizando um ambiente independente, é possível configurar o failover com a linha de comandos. Se você estiver usando um ambiente do WebSphere Application Server Network Deployment, é necessário configurar o failover no console administrativo do WebSphere Application Server Network Deployment.

Procedimento

- Configurar o failover para ambientes independentes.

  É possível configurar os intervalos de pulsação na linha de comandos usando o parâmetro **-heartbeat** no arquivo de script startOgServer.bat | startOgServer.sh. Configure esse parâmetro para um dos seguintes valores:

  Tabela 28. Intervalos de Pulsações

Valor	Ação	Descrição
0	Típica (padrão)	Failovers são tipicamente detectados em 30 segundos.
-1	Agressiva	Failovers são tipicamente detectados em 5 segundos.
1	Moderada	Failovers são tipicamente detectado em 180 segundos.

  Um intervalo de pulsação agressivo pode ser útil quando os processos e a rede estão estáveis. Se a rede ou os processos não são configurados de maneira ideal, as pulsações podem ser perdidas, o que pode resultar em uma falsa detecção de falhas.

- Configurar o failover para ambientes do WebSphere Application Server.

  O WebSphere Application Server Network Deployment Versão 6.0.2 e superior pode ser configurado para permitir que o WebSphere eXtreme Scale execute failover muito rapidamente. O tempo de failover padrão para falhas "hard" é de aproximadamente 200 segundos. Uma falha "hard" é um travamento de computador físico ou de servidor, uma desconexão de cabo de rede ou um erro do sistema operacional. As falhas causadas por travamentos de processo ou por falhas de software normalmente executam failover em menos de um segundo. A detecção de falha para falhas "soft" ocorre quando os soquetes de rede a partir de um processo inativo são fechados automaticamente pelo sistema operacional para o servidor que hospeda o processo.

  Configuração de pulsação do grupo principal

  O WebSphere eXtreme Scale que executa um processo do WebSphere Application Server herda as características de failover a partir das configurações de grupo principal do servidor de aplicativos. As seguintes seções descrevem como definir as configurações de pulsação do grupo principal para versões diferentes do WebSphere Application Server Network Deployment:

  – **Atualize as configurações do grupo principal para o WebSphere Application Server Network Deployment Versão 6.*x* e 7.*x*:**

    O intervalo de pulsação pode ser especificado em segundos nas versões do WebSphere Application Server da Versão 6.0 a Versão 6.1.0.12, ou em milissegundos iniciando na Versão 6.1.0.13. Também é necessário especificar o número de pulsações perdidas. Esse valor indica quantas pulsações podem estar ausentes antes que um Java Virtual Machine (JVM) equivalente seja considerado uma falha. O tempo de detecção de falha "hard" é, aproximadamente, o produto do intervalo de pulsações e o número de pulsações ausentes.

Essas propriedades são especificadas usando as propriedades customizadas no grupo principal usando o console administrativo do WebSphere. Consulte Propriedades customizadas do grupo principal para obter detalhes da configuração. Estas propriedades devem ser especificadas para todos os grupos principais utilizados pelo aplicativo:

- O intervalo de pulsação é especificado usando a propriedade customizada IBM_CS_FD_PERIOD_SEC para segundos ou a propriedade customizada IBM_CS_FD_PERIOD_MILLIS para milissegundos (requer o V6.1.0.13 ou superior).
- O número de pulsações ausentes é especificado usando a propriedade customizada IBM_CS_FD_CONSECUTIVE_MISSED.

O valor padrão para a propriedade IBM_CS_FD_PERIOD_SEC é 20 e para a propriedade IBM_CS_FD_CONSECUTIVE_MISSED é 10. Se a propriedade IBM_CS_FD_PERIOD_MILLIS for especificada, ela substituirá qualquer conjunto de propriedades customizadas IBM_CS_FD_PERIOD_SEC. Os valores destas propriedades são valores de número inteiro positivo.

Use as seguintes configurações para alcançar um tempo de detecção de falha de 1500 milissegundos para os servidores WebSphere Application Server Network Deployment Versão 6.x:

- Configure `IBM_CS_FD_PERIOD_MILLIS = 750` (WebSphere Application Server Network Deployment V6.1.0.13 e superior)
- Configure `IBM_CS_FD_CONSECUTIVE_MISSED = 2`

– **Atualize as configurações do grupo principal para o WebSphere Application Server Network Deployment Versão 7.0:**

O WebSphere Application Server Network Deployment Versão 7.0 fornece duas configurações de grupo principal que podem ser ajustadas para aumentar ou diminuir a detecção de failover:

- **Período de transmissão de pulsação.** O padrão é 30000 milissegundos.
- **Período de tempo limite de pulsação.** O padrão é 180000 milissegundos.

Para obter mais detalhes sobre como alterar essas configurações, consulte o Centro de Informações do WebSphere Application Server Network Deployment: Configurações de Falha e Detecção de Descoberta.

Use as seguintes configurações para alcançar um tempo de detecção de falha de 1500 milissegundos para os servidores WebSphere Application Server Network Deployment Versão 7:

- Configure o período de transmissão de pulsação para 750 milissegundos.
- Configure o tempo limite da pulsação para 1500 milissegundos.

O que Fazer Depois

Quando estas configurações são modificadas para fornecer tempos de failover curtos, há alguns problemas de ajuste de sistema a considerar. Primeiro, Java não é um ambiente em tempo real. É possível que os encadeamentos sejam atrasados se a JVM estiver experimentando longos tempos de coleta de lixo. Os encadeamentos também podem ser atrasados se a máquina que hospeda o JVM estiver sobrecarregada (devido ao próprio JVM ou outros processos que são executados na máquina). Se os encadeamentos forem atrasados, as pulsações talvez não sejam enviadas a tempo. No pior dos casos, elas podem ser atrasadas pelo tempo necessário de failover. Se os encadeamentos forem atrasados, ocorrerão falsas detecções de falhas. O sistema deve ser ativado e dimensionado para garantir que falsas detecções de falhas não aconteçam na produção. O teste de carregamento adequado é a melhor maneira de garantir isto.

Nota: A versão atual do eXtreme Scale suporta o WebSphere Real Time.

Tipos de Detecção de Failover

WebSphere eXtreme Scale pode detectar falhas de forma confiável.

Pulsação

1. Os soquetes são mantidos abertos entre o Java Virtual Machines e se um soquete for fechado inesperadamente, esse fechamento inesperado será detectado como uma falha do peer Java Virtual Machine. Essa detecção captura casos de falha, como a Java Virtual Machine saindo rapidamente. Tal detecção permite a recuperação desses tipos de falhas normalmente em menos de um segundo.
2. Outros tipos de falhas incluem: um pânico no sistema operacional, falha física do servidor ou falha de rede. Essas falhas são descobertas por meio de pulsação.

Pulsações são enviadas periodicamente entre pares de processos: Quando um número fixo de pulsações está ausente, é assumida uma falha. Essa abordagem detecta falhas em N*M segundos, em que N é o número de pulsações ausentes e M é o intervalo no qual as pulsações devem ser configuradas. A especificação direta de M e N não é suportada, e ao invés disso, um mecanismo de régua de controle é utilizado para permitir um intervalo de combinações de M e N testadas a ser utilizado.

Falhas

Um processo pode falhar de várias maneiras. o processo poderia falhar porque algum limite de recurso foi atingido, tal como o tamanho máximo do heap ou alguma lógica de controle de processo terminou um processo. O sistema operacional poderia falhar, fazendo com que todos os processos em execução no mesmo sistema fossem perdidos. O hardware pode falhar, embora com menos frequência, como a placa da interface de rede (NIC), o que faria o sistema operacional ser desconectado da rede. Neste contexto, todas estas falhas podem ser categorizadas em um dos dois tipos: falha de processo e perda de conectividade.

Falha de Processo

O WebSphere eXtreme Scale reage para processar falhas muito rapidamente. Quando um processo falha, o sistema operacional é responsável pela limpeza de qualquer recurso pendente que o processo estava utilizando. Essa limpeza inclui a alocação e conectividade da porta. Quando um processo falha, um sinal é enviado imediatamente por meio das conexões que estavam sendo utilizadas por esse processo para fechar cada conexão.

Perda de Conectividade

A perda de conectividade ocorre quando o sistema operacional se desconecta. Como resultado, o sistema operacional não pode enviar sinais a outros processos. Há diversos motivos pelos quais pode ocorrer uma perda de conectividade, mas eles podem ser divididos em duas categorias: falha de host e ilhamento.

Falha de Host

Se uma máquina host perder a energia, ela ficará indisponível imediatamente.

Insulamento

Este cenário apresenta a condição de falha mais complicada para o software tratar corretamente porque o processo é presumido como indisponível, embora não esteja.

Falha de Contêiner

Falhas contêiner geralmente são descobertas por contêineres peer por meio do mecanismo de grupo principal. Quando um contêiner ou conjunto de contêineres falha, o serviço de catálogo migra os fragmentos que foram hospedados nesse contêiner ou contêineres. O serviço de catálogo procura uma réplica síncrona primeiro, antes de migrar para uma réplica assíncrona. Depois da migração dos fragmentos primários para os novos contêineres de host, o serviço de catálogo procura novos contêineres de host para as réplicas que agora estão faltando.

Nota: Ilhamento de contêiner - O serviço de catálogo migra shards dos contêineres quando descobre-se que o contêiner está indisponível. Se esses contêineres se tornarem disponíveis, o serviço de catálogo considerará os contêineres elegíveis para disposição como no fluxo de inicialização normal.

Latência de detecção de failover de contêiner

As falhas podem ser categorizadas em falhas brandas e falhas graves. Falhas brandas normalmente são causadas quando um processo falha. Tais falhas são detectadas pelo sistema operacional, que pode recuperar recursos utilizados, tais como soquetes de rede, muito rapidamente. A detecção de falha típica para falhas brandas é de menos de um segundo. A falhas graves podem levar até 200 segundos até detectar utilizando o ajuste de pulsação padrão. Tais falhas incluem: falhas de máquina física, desconexões de cabo de rede ou falhas de sistema operacional. Deste modo, o eXtreme Scale deve contar com a pulsação para detectar falhas graves que podem ser configuradas.

Várias Falhas de Contêiner

Uma réplica nunca é colocada no mesmo processo que seu primário porque, se o processo for perdido, isso resultará na perda do primário e da réplica. A política de implantação define um atributo que o serviço de catálogo utiliza para determinar se uma réplica pode ser colocada na mesma máquina que o primário. Em um ambiente de implantação em uma única máquina, talvez você queira ter dois contêineres e replicar entre eles. Entretanto, em produção, utilizar uma máquina única é suficiente porque a perda de tal host resulta na perda de ambos os contêineres. Para alterar entre o modo de desenvolvimento em uma máquina única e um modo de produção com várias máquinas, desative o modo de desenvolvimento no arquivo de configuração da política de implementação.

Falha de Serviço de Catálogo

Como a grade de serviço do catálogo é uma grade do eXtreme Scale, ela também usa o mecanismo de agrupamento principal da mesma forma que o processo de falha de contêiner. A principal diferença é que o domínio do serviço de catálogo usa um processo de eleição de peer para definir o shard principal em vez do algoritmo do serviço de catálogo usado para os contêineres.

Observe que o serviço de posicionamento e o serviço de agrupamento principal são serviços Um de N. Um serviço Um de N é executado em um membro do

grupo de alta disponibilidade. O serviço de local e a administração são executados em todos os membros do grupo de alta disponibilidade. O serviço de disposição e o serviço de agrupamento principal são singletons porque são responsáveis pela configuração do sistema. O serviço de local e a administração são serviços somente leitura e existe em qualquer lugar para fornecer escalabilidade.

O serviço de catálogo usa a replicação para tornar-se tolerante a falhas. Se um processo de serviço de catálogo falhar, o serviço deverá ser reiniciado para restaurar o sistema para o nível de disponibilidade desejado. Se todos os processos que estão hospedando o serviço de catálogo falharem, o eXtreme Scale possui uma perda de dados críticos. Essa falha resulta em um reinício necessário de todos os contêineres. Como o serviço de catálogo pode ser executado em muitos processos, essa falha é um evento improvável. Entretanto, se você estiver executando todos os processos em uma única caixa, dentro de um único chassi de blade, ou de um único comutador de rede, será mais provável que uma falha ocorra. Tente remover os modos de falha comuns das caixas que estão hospedando o serviço de catálogo para reduzir a possibilidade de falha.

Tabela 29. Resumo de Descoberta de Falha e Recuperação

Tipo de perda	**Mecanismo de descoberta (detecção)**	**Método de recuperação**
Perda de processo	E/S	Reiniciar
Perda de servidor	Pulsação	Reiniciar
Interrupção da rede	Pulsação	Reestabelecer rede e conexão
Interrupção do lado do servidor	Pulsação	Parar e reiniciar servidor
Servidor ocupado	Pulsação	Aguardar até servidor estar disponível

Usando o WebSphere Real Time

O uso do WebSphere eXtreme Scale com WebSphere Real Time aumenta a consistência e a previsibilidade a um custo de rendimento de desempenho em comparação com a política de coleta de lixo padrão empregada no IBM Java™ SE Runtime Environment (JRE) padrão. O custo versus a proposta de benefício pode variar. WebSphere eXtreme Scale cria vários objetos temporários que são associados a cada transação. Esses objetos temporários lidam com pedidos, respostas, sequências de log e sessões. Sem o WebSphere Real Time, o tempo de resposta de transação pode ultrapassar centenas de milissegundos. Entretanto, usar o WebSphere Real Time com o WebSphere eXtreme Scale pode aumentar a eficiência da coleta de lixo e reduzir o tempo de resposta em 10% do tempo de resposta de configuração independente.

WebSphere Real Time em um Ambiente Independente

É possível usar o WebSphere Real Time com o WebSphere eXtreme Scale. Ao ativar o WebSphere Real Time, é possível obter uma coleta de lixo mais previsível com um tempo de resposta e rendimento estável e consistente em um ambiente independente do eXtreme Scale.

Vantagens do WebSphere Real Time

WebSphere eXtreme Scale cria vários objetos temporários que são associados a cada transação. Esses objetos temporários lidam com pedidos, respostas, sequências de log e sessões. Sem o WebSphere Real Time, o tempo de resposta de transação pode ultrapassar centenas de milissegundos. Entretanto, usar o WebSphere Real Time com o WebSphere eXtreme Scale pode aumentar a eficiência da coleta de lixo e reduzir o tempo de resposta em 10% do tempo de resposta de configuração independente.

Ativando o WebSphere Real Time

Instale o WebSphere Real Time e o WebSphere eXtreme Scale independente nos computadores onde você planeja executar o eXtreme Scale . Configure a variável de ambiente JAVA_HOME para apontar para o Java SE Runtime Environment (JRE) padrão.

Configure a variável de ambiente JAVA_HOME para apontar para o WebSphere Real Time instalado. Em seguida, ative o WebSphere Real Time da seguinte forma.

1. Edite o arquivo de instalação padrão `objectgridRoot/bin/setupCmdLine.sh | .bat` ao remover o comentário da seguinte linha.

   ```
   WXS_REAL_TIME_JAVA="-Xrealtime -Xgcpolicy:metronome
   -Xgc:targetUtilization=80"
   ```

2. Salve o arquivo.

Agora você ativou o WebSphere Real Time. Se desejar desativar o WebSphere Real Time, poderá incluir o comentário de volta para a mesma linha.

Práticas Recomendáveis

WebSphere Real Time permite que transações do eXtreme Scale tenham um tempo de resposta mais previsível. Os resultados mostram que o desvio de um tempo de resposta de uma transação do eXtreme Scale aumenta significativamente com o WebSphere Real Time em comparação com o Java padrão com seu coletor de lixo padrão. A ativação do WebSphere Real Time com o eXtreme Scale é ótima se a estabilidade e o tempo de resposta de seu aplicativo forem essenciais.

As boas práticas descritas nesta seção explicam como tornar o WebSphere eXtreme Scale mais eficiente por meio do ajuste e de práticas de código dependendo de sua carga esperada.

- Configure o nível de uso correto do processador para o aplicativo e coletor de lixo.

  O WebSphere Real Time fornece a capacidade de controlar o uso do processador para que o impacto da coleta de lixo em seu aplicativo seja controlado e minimizado. Use o parâmetro `-Xgc:targetUtilization=NN` para especificar a porcentagem `NN` do processador que é usado pelo aplicativo a cada 20 segundos. O padrão para WebSphere eXtreme Scale é 80%, mas é possível modificar o script no arquivo `objectgridRoot/bin/setupCmdLine.sh` para configurar um número diferente, como `70`, que fornece mais capacidade de processador para o coletor de lixo. Implemente servidores suficientes para manter a carga do processador abaixo de 80% para seus aplicativos.
- Configure um tamanho maior de memória de heap.

  O WebSphere Real Time usa mais memória do que o Java comum, assim, planeje seu WebSphere eXtreme Scale com uma memória de heap e configure o tamanho

de heap ao iniciar os servidores de catálogo e os contêineres com o parâmetro `–jvmArgs –XmxNNNM` no comando ogStartServer. Por exemplo, é possível usar o parâmetro `–jvmArgs –Xmx500M` para iniciar os servidores de catálogo e usar o tamanho de memória apropriado para iniciar os contêineres. O tamanho de memória pode ser configurado de 60 a 70% do tamanho de dados esperado por JVM. Se você não configurar esse valor, ocorrerá um erro OutOfMemoryError. Opcionalmente, use o parâmetro `–jvmArgs –Xgc:noSynchronousGCOnOOM` para evitar um comportamento indesejável quando a JVM ficar sem memória.

- Ajuste os encadeamentos para a coleta de lixo.

  O WebSphere eXtreme Scale cria muitos objetos temporários associados a cada transação e encadeamentos Remote Procedure Call (RPC). A coleta de lixo possui benefícios de desempenho se o computador tiver ciclos de processadores suficientes. O número padrão de encadeamentos é 1. É possível alterar o número de encadeamentos com o argumento `–Xgcthreads n`. O valor sugerido para esse argumento é o número de núcleos que estão disponíveis em relação ao número de Java virtual machines por computador.

- Ajuste o desempenho para aplicativos de curta duração com oWebSphere eXtreme Scale.

  O WebSphere Real Time é ajustado para aplicativos de execução longa. Geralmente, é necessário executar transações contínuas do WebSphere eXtreme Scale por duas horas para obter dados de desempenho confiáveis. É possível usar o parâmetro `–Xquickstart` para melhorar o desempenho de aplicativos de curta duração. Esse parâmetro informa ao compilador just-in-time (JIT) para usar um nível menor de otimização.

- Minimize a fila do cliente do WebSphere eXtreme Scale e a retransmissão do cliente do WebSphere eXtreme Scale.

  A principal vantagem de usar o WebSphere eXtreme Scale com o WebSphere Real Time é que o tempo de resposta da transação é altamente confiável, o que normalmente representa várias vezes melhorias na magnitude de ordem no desvio do tempo de resposta da transação. Todos os pedidos do cliente enfileirados e retransmissões de pedido do cliente por meio de outro software afetam o tempo de resposta que está além do controle do WebSphere Real Time e do WebSphere eXtreme Scale. É necessário alterar seus encadeamentos e parâmetros de soquete para manter uma carga estável e leve sem nenhum atraso significativo e diminuição da profundidade da fila.

- Escreva aplicativos WebSphere eXtreme Scale para usar o encadeamento do WebSphere Real Time.

  Sem modificar seu aplicativo, é possível obter tempo de reposta de transação WebSphere eXtreme Scale altamente confiável com diversas melhorias da magnitude de ordem no desvio do tempo de resposta. É possível explorar ainda mais a vantagem do encadeamento de seus aplicativos transacionais a partir do encadeamento Java normal para RealtimeThread o que oferece melhor controle sobre a prioridade de encadeamento e controle de planejamento.

  Seu aplicativo atualmente inclui o código a seguir.

  ```
  public class WXSCacheAppImpl extends Thread implements WXSCacheAppIF
  ```

  É possível substituir opcionalmente este código pelo seguinte.

  ```
  public class WXSCacheAppImpl extends RealtimeThread implements
  WXSCacheAppIF
  ```

WebSphere Real Time no WebSphere Application Server

É possível utilizar o WebSphere® Real Time com eXtreme Scale em um ambiente do WebSphere Application Server Network Deployment versão 7.0. Ao ativar o

WebSphere Real Time, é possível obter uma coleta de lixo mais previsível com tempo de resposta e rendimento estáveis e consistentes das transações.

Vantagens

O uso do WebSphere eXtreme Scale com WebSphere Real Time aumenta a consistência e a previsibilidade a um custo de rendimento de desempenho em comparação com a política de coleta de lixo padrão empregada no IBM Java™ SE Runtime Environment (JRE) padrão. O custo versus a proposta de benefício pode variar com base em vários critérios. A seguir estão alguns dos principais critérios:

- Recursos do servidor - Memória disponível, tamanho e velocidade da CPU e velocidade e uso da rede
- Carregamentos do servidor - Carregamento de CPU sustentado, carregamento de CPU de pico
- Configuração Java – Tamanhos de heap, uso de destino, encadeamentos de coleta de lixo
- Configuração do modo de cópia do WebSphere eXtreme Scale - Matriz de byte vs. armazenamento POJO
- Características específicas do aplicativo – Uso de encadeamento, requisitos de resposta e tolerância, tamanho do objeto, entre outras.

Além dessa política de coleta de lixo de metrônomo disponível no WebSphere Real Time, existem políticas de coleta de lixo opcionais disponíveis no IBM Java™ SE Runtime Environment (JRE) padrão. Essas políticas, optthruput (padrão), gencon, optavgpause e subpool são projetadas especificamente para solucionar ambientes e requisitos de aplicativo diferentes. Para obter mais informações sobre essas políticas, consulte “Ajuste da JVM para o WebSphere eXtreme Scale” na página 427. Dependendo dos requisitos, recursos e restrições do aplicativo e do ambiente, a criação de um protótipo de uma ou mais dessas políticas de coleta de lixo pode garantir que você atenda aos seus requisitos e determine a política ideal.

Recursos com o WebSphere Application Server Network Deployment

1. A seguir estão algumas versões suportadas.
   - WebSphere Application Server Network Deployment versão 7.0.0.5 e acima.
   - WebSphere Real Time V2 SR2 para Linux e acima. Consulte IBM WebSphere Real Time V2 para Linux para obter mais informações.
   - WebSphere eXtreme Scale versão 7.0.0.0 e acima.
   - Sistemas operacionais Linux de 32 e 64 bits.
2. Servidores WebSphere eXtreme Scale não podem ser colocados junto com um WebSphere Application Server DMgr.
3. Real Time não suporta DMgr.
4. Real Time não suporta WebSphere Node Agents.

Ativando o WebSphere Real Time

Instale o WebSphere Real Time e o WebSphere eXtreme Scale nos computadores nos quais você pretende executar o eXtreme Scale. Atualize o WebSphere Real Time Java para SR2.

É possível especificar as configurações de JVM para cada servidor por meio do console do WebSphere Application Server versão 7.0 da seguinte forma.

Escolha **Servidores → Tipos de Servidor → WebSphere Application Servers → <servidor instalado necessário>**

Na página resultante, escolha "Definição de Processo".

Na próxima página, clique em Java Virtual Machine na parte superior da coluna à direita. (Aqui é possível configurar tamanhos de heap, coleta de lixo e outros sinalizadores para cada servidor.)

Configure os seguintes sinalizadores no campo "Argumentos de JVM Genéricos":

```
-Xrealtime -Xgcpolicy:metronome
-Xnocompressedrefs -Xgc:targetUtilization=80
```

Aplique e salve as mudanças.

Para utilizar o Real Time no WebSphere Application Server 7.0 com servidores eXtreme Scale, incluindo os sinalizadores JVM acima, você deverá criar a variável de ambiente JAVA_HOME.

Configure JAVA_HOME da seguinte forma.

1. Expanda "Ambiente".
2. Selecione "Variáveis do WebSphere".
3. Certifique-se de "Todos os Escopos" esteja marcado em “Mostrar Escopo".
4. Selecione o servidor necessário na lista suspensa. (Não selecione DMgr ou servidores de agente do nó.)
5. Se a variável de ambiente JAVA_HOME não estiver listada, selecione "Novo" e especifique JAVA_HOME para o nome da variável. No campo "Valor", insira o nome do caminho completo para o Real Time.
6. Aplique e salve suas mudanças.

Práticas Recomendáveis

Para um conjunto de boas práticas, consulte a seção de boas práticas em “Usando o WebSphere Real Time” na página 435. Existem algumas modificações importantes que devem ser observadas nesta lista de boas práticas para um ambiente WebSphere eXtreme Scale independente durante a implementação em um ambiente WebSphere Application Server Network Deployment.

Você deve colocar todos os parâmetros da linha de comandos adicionais da JVM no mesmo local que os parâmetros da política de coleta de lixo especificados na seção anterior.

Um destino inicial aceitável para carregamentos do processador sustentados é 50% com carregamentos de pico de duração curta atingindo até 75%. Além disso, você deve incluir capacidade adicional antes de ver a degradação mensurável na previsibilidade e consistência. É possível aumentar um pouco o desempenho se você puder tolerar tempos de resposta mais longos. Se você exceder 80% do limite, isso pode levar à degradação significativa da consistência e da previsibilidade.

Ajustando o Provedor de Cache Dinâmico

O provedor de cache dinâmico do WebSphere eXtreme Scale suporta os seguintes parâmetros de configuração para o ajuste de desempenho.

Sobre Esta Tarefa

- **com.ibm.websphere.xs.dynacache.ignore_value_in_change_event**: Ao registrar um listener de evento de mudança com o provedor de cache dinâmico e gerar uma instância ChangeEvent, haverá um gasto adicional associado à desserialização da entrada do cache para que o valor possa ser colocado dentro o ChangeEvent. Configurar este parâmetro opcional na instância do cache para `true` ignorará a desserialização da entrada do cache ao gerar o ChangeEvents. O valor retornado será nulo no caso de uma operação de remoção ou de uma matriz de byte que contém a forma serializada do objeto. As instâncias InvalidationEvent possuem uma penalidade de desempenho semelhante que pode ser evitada ao configurar com.ibm.ws.cache.CacheConfig.ignoreValueInInvalidationEvent para `true`.
- **com.ibm.websphere.xs.dynacache.enable_compression**: Por padrão, o provedor de cache dinâmico do eXtreme Scale compacta as entradas de cache na memória para aumentar a densidade do cache, que pode economizar uma quantia significativa de memória para aplicativos como o armazenamento em cache do servlet. Se você souber que a maioria dos dados do cache não poderão ser compactados, considere a possibilidade de configurar este valor para false.

Ajustando o Agente de Dimensionamento de Cache para Estimativas Exatas de Consumo de Memória

Começando com a Versão 7.1, o WebSphere eXtreme Scale suporta o dimensionamento do consumo de memória de BackingMaps em grades distribuídas. (O dimensionamento do consumo de memória não é suportado para instâncias de grades locais.) Na maioria dos casos, o valor relatado por WebSphere eXtreme Scale para um determinado mapa está muito próximo do valor relatado pela análise de dump do heap. A complexidade de um objeto de mapa pode impedir dimensionamentos exatos. De fato, a mensagem CWOBJ4543 é exibida no log para qualquer objeto de entrada de cache que não pode ser dimensionado com exatidão porque é excessivamente complexo. Seguir as boas práticas descritas aqui para evitar complexidade de mapa desnecessária aprimorará a exatidão da medida de dimensionamento.

Procedimento

- Ative o agente de dimensionamento.

  Se estiver usando uma JVM Java 5 ou superior, aproveite o agente de dimensionamento, que permite que o WebSphere eXtreme Scale obtenha informações adicionais da JVM para melhorar suas estimativas. O agente pode ser carregado incluindo o seguinte argumento na linha de comandos JVM:

  ```
  -javaagent:WXS lib directory/wxssizeagent.jar
  ```

  Para uma topologia integrada, inclua o argumento na linha de comandos do processo do WebSphere Application Server.

  Para uma topologia distribuída, inclua o argumento na linha de comandos dos processos (contêineres) do eXtreme Scale e do processo do WebSphere Application Server.

  Quando carregada corretamente, a seguinte mensagem é gravada no arquivo `SystemOut.log`.

  ```
  CWOBJ4541I: O dimensionamento de memória de BackingMap aprimorado está ativado.
  ```

- Prefira os tipos de dados Java no lugar de tipos de dados customizados, onde possível.

  O WebSphere eXtreme Scale pode dimensionar com exatidão o custo de memória dos seguintes tipos:

  - java.lang.String e matrizes em que String é a classe do componente (String[])
  - Todos os tipos de wrapper primitivos (Byte, Short, Character, Boolean, Long, Double, Float, Integer) e matrizes em que os wrappers primitivos são o tipo de componente (por exemplo, Integer[], Character[])
  - java.math.BigDecimal e java.math.BigInteger e matrizes em que essas duas classes são o tipo de componente (BigInteger[] e BigDecimal[])
  - Tipos temporais (java.util.Date, java.sql.Date, java.util.Time, java.sql.Timestamp)
  - java.util.Calendar e java.util.GregorianCalendar
- Evite a internação do Objeto, quando possível.

  Quando um Objeto é inserido em um mapa, o WebSphere eXtreme Scale assume que ele possui a única referência ao Objeto e a todos os Objetos diretamente referidos por ele. Se você inserir 1000 Objetos customizados em um mapa e cada um tiver uma referência à mesma instância de String, o WebSphere eXtreme Scale dimensionará essa instância de String 1000 vezes, superestimando o tamanho real do mapa no heap. No entanto, o WebSphere eXtreme Scale compensará corretamente para os seguintes cenários de internação comuns:
  - Referências a Enumerações de Java 5
  - Referências a Classes que seguem o Padrão de Enumeração Typesafe. As classes que seguem este padrão terão apenas construtores privados definidos, terão no mínimo um campo final estático privado de seu próprio tipo e, se implementarem Serializável, implementarão readResolve().
  - Internação de wrapper Primitivo Java 5. Por exemplo, usar Integer.valueOf(1) em vez do novo Integer(1)

  Portanto, se você tiver que fazer a internação, use uma das técnicas precedentes.
- Use tipos customizados com atenção.

  Ao usar tipos customizados, prefira tipos de dados primitivos para campos vs tipos de Objeto.

  Além disso, prefira os tipos de Objeto listados na entrada 2 no lugar de suas próprias implementações customizadas.

  Ao usar tipos customizados, mantenha a árvore de Objeto para um nível. Ao inserir um Objeto customizado em um mapa, o WebSphere eXtreme Scale calculará apenas o custo do OBjeto inserido, que inclui os campos primitivos e todos os Objetos aos quais ele faz referência diretamente. O WebSphere eXtreme Scale não seguirá referências adicionais na árvore de Objeto. Se você inserir um Objeto no mapa e o WebSphere eXtreme Scale detectar que as referências não foram seguidas durante o processo de dimensionamento, você obterá uma mensagem codificada CWOBJ4543 que incluirá o nome da Classe que não pôde ser totalmente dimensionada. Quando isso ocorrer, trate as estatísticas de dimensionamento no mapa como dados de tendência, em vez de confiar nas estatísticas de dimensionamento como um total exato.
- Use CopyMode.COPY_TO_BYTES, se possível.

  Usar CopyMode.COPY_TO_BYTES remove qualquer dúvida sobre dimensionar os Objetos de valor que estão sendo inseridos no mapa, mesmo quando uma árvore de Objeto tem níveis demais para serem dimensionados normalmente (resultando na mensagem CWOBJ4543).

Dimensionamento do Consumo do Cache de Memória

Iniciando com o release 7.1, o WebSphere eXtreme Scale pode fazer uma estimativa exata do uso de memória do heap Java de um determinado BackingMap em bytes. Aproveite este recurso para ajudar a dimensionar corretamente as configurações de

heap e as políticas de despejo da Java virtual machine. O comportamento deste recurso varia com a complexidade dos Objetos que estão sendo colocados no mapa de retorno e com a forma como o mapa é configurado. Atualmente, este recurso é suportado apenas para grades distribuídas. As instâncias de grade local não suportam o dimensionamento de bytes usado.

O eXtreme Scale armazena todos os seus dados dentro do espaço de heap dos processos JVM que compõem uma grade. Para um determinado mapa, o espaço de heap que ele consome pode ser dividido nos seguintes componentes:

- O tamanho de todos os Objetos Chave atualmente no mapa
- O tamanho de todos os Objetos de Valor atualmente no mapa
- O tamanho de todos os objetos EvictorData em uso pelos plug-ins Evictor nesse mapa
- A sobrecarga da estrutura de dados subjacente

O número de bytes usados relatados pelas estatísticas de dimensionamento é a soma desses quatro custos. Esses valores são calculados por entrada nas operações de inserção, atualização e remoção de mapa, o que significa que o eXtreme Scale sempre tem um valor atual para o número de bytes que um determinado BackingMap está consumindo.

Quando as grades são particionadas, cada partição contém uma parte do BackingMap. Como as estatísticas de dimensionamento são calculadas no nível mais baixo do código do eXtreme Scale, cada partição de um BackingMap rastreia seu próprio tamanho. É possível usar as APIs de Estatísticas do eXtreme Scale para rastrear o tamanho acumulativo do mapa e também o tamanho de suas partições individuais.

No geral, trate os dados de dimensionamento como “dados de tendência", em oposição a uma medida exata do espaço de heap usada pelo mapa. Por exemplo, se o tamanho relatado de um mapa dobra de 5 MB para 10 MB, visualize o consumo de memória do mapa como tendo dobrado. A medida real de 10 MB pode não ser exata por vários motivos discutidos aqui. Se você levar em conta os motivos e seguir as boas práticas, a exatidão das medidas de tamanho se aproximará daquelas do pós-processamento de um dump de heap Java.

O principal problema com a exatidão é que o Modelo de Memória Java não é restritivo o suficiente para permitir medidas de memória que devem ser exatas. O problema fundamental é que um Objeto pode estar ativo no heap devido a várias referências. Por exemplo, se a mesma instância de Objeto de 5 kb for inserida em três mapas separados, qualquer um desses três mapas impedirá que o Objeto seja posto na lixeira. Nesta situação, qualquer uma das medidas a seguir seria justificável:

- O tamanho de cada mapa é aumentado em 5 kb
- O tamanho do primeiro mapa no qual o Objeto é colocado é aumentado em 5 kb
- Os outros dois mapas não são aumentados. O tamanho de cada mapa é aumentado em uma fração do tamanho do Objeto

Esta ambiguidade é o motivo pelo qual essas medidas devem ser consideradas dados de tendência, a não ser que você tenha removido a ambiguidade por meio de opções de design, boas práticas e do entendimento das opções de implementação feitas com este recurso.

O eXtreme Scale supõe que um determinado mapa retém a única referência de vida longa para os Objetos de chave e de valor que ele contém. Se o mesmo objeto de 5 kb for colocado em três mapas, o tamanho de cada mapa será aumentado em 5 kb. O aumento, geralmente, não é um problema, pois o recurso é suportado apenas por grades distribuídas. Se você inserir o mesmo Objeto em três mapas diferentes em um cliente remoto, cada mapa obterá sua própria cópia do Objeto. As configurações COPY MODE transacionais padrão também geralmente garantem que cada mapa tenha sua própria cópia de um determinado Objeto.

O internamento de objetos será o maior desafio para os aplicativos da maioria dos clientes. O internamento de objetos é quando o código do aplicativo intencionalmente garante que todas as referências a um determinado valor do Objeto realmente apontem para a mesma instância do Objeto no heap. Um exemplo disso pode ser a classe a seguir.

```
public class ShippingOrder implements Serializeable,Cloneable{

    public static final STATE_NEW = "new";
    public static final STATE_PROCESSING = "processing";
    public static final STATE_SHIPPED = "shipped";

    private String state;
    private int orderNumber;
private int customerNumber;

public Object clone(){
       ShippingOrder toReturn = new ShippingOrder();
       toReturn.state = this.state;
       toReturn.orderNumber = this.orderNumber;
       toReturn.customerNumber = this.customerNumber;
       return toReturn;
    }

    private void readResolve(){
        if (this.state.equalsIgnoreCase("new")
            this.state = STATE_NEW;
        else if (this.state.equalsIgnoreCase("processing")
            this.state = STATE_PROCESSING;
        else if (this.state.equalsIgnoreCase("shipped")
            this.state = STATE_SHIPPED:
    }
}
```

Não importa a configuração do eXtreme Scale, o custo real para essa classe será superestimado pelas estatísticas de dimensionamento. Se houver um milhão de objetos de ordem de Remessa, o código de dimensionamento refletirá o custo de um milhão de Cadeias retendo as informações de estado. Na realidade, há realmente apenas três Cadeias e elas são membros estáticos da classe. Seu custo de memória nunca deve ser incluído em nenhum mapa do eXtreme Scale, mas não existe uma boa maneira para detectar esta situação no tempo de execução. Há dúzias de maneiras pelas quais um internamento semelhante pode ser atingido, que é o motivo pelo qual é tão difícil detectar. Não é prático para o eXtreme Scale proteger contra todas elas. No entanto, é possível para o eXtreme Scale proteger contra os tipos mais comumente usados.

O eXtreme Scale será automaticamente ajustado para 5 enumerações Java e o padrão Typesafe Enum, conforme descrito em http://java.sun.com/j2se/1.5.0/docs/guide/language/enums.html.

Para otimizar o uso da memória por meio do internamento do Objeto, interne apenas os Objetos customizados que fazem parte dessas duas categorias. Seguir

esta prática aprimorará a exatidão das estatísticas de consumo de memória. Além disso, no Java 5, o internamento automático para tipos de wrapper primitivos, como Inteiro, foi introduzido por meio do uso de métodos valueOf() estáticos. O eXtreme Scale contará automaticamente para este internamento.

Use um dos métodos a seguir para acessar as estatísticas de consumo de memória.

API de Estatísticas

Use o método MapStatsModule.getUsedBytes(), que fornece estatísticas para um único mapa, incluindo o número de entradas e a taxa de ocorrências.

Para obter detalhes, consulte "Módulos Estatísticos" na página 384.

Beans Gerenciados (MBeans)

Use a estatística Mbean gerenciada por MapUsedBytes. É possível usar vários tipos diferentes de Java Management Extensions (JMX) MBeans para administrar e monitorar as implementações. Cada MBean faz referência a uma entidade específica, como um mapa, um eXtreme Scale, um servidor, um grupo de replicação ou um membro do grupo de replicação.

Para obter detalhes, consulte "Administrando Programaticamente com Beans Gerenciados (MBeans)" na página 353.

Módulos PMI (Performance Monitoring Infrastructure)

É possível monitorar o desempenho de seus aplicativos com os módulos PMI. Especificamente, use o módulo PMI de mapa para contêineres integrados no WebSphere Application Server.

Para obter detalhes, consulte "Módulos PMI" na página 392.

Console do WebSphere eXtreme Scale

O console, introduzido na Versão 7.1, permite visualizar as estatísticas de consumo de memória. Consulte "Monitorando com o Console da Web" na página 400.

Todos esses métodos acessam a mesma medida subjacente do consumo de memória de uma determinada instância BaseMap. O tempo de execução de WebSphere eXtreme Scale tenta (o melhor esforço) calcular o número de bytes de memória de heap consumidos pelos objetos de chave e de valor armazenados no mapa e também a sobrecarga do próprio mapa. É possível ver quanta memória de heap cada mapa está consumindo por meio de toda a grade distribuída.

Na maioria dos casos, o valor relatado por WebSphere eXtreme Scale para um determinado mapa estará muito próximo do valor relatado pela análise de dump de heap. O WebSphere eXtreme Scale dimensiona exatamente sua própria sobrecarga, mas não pode considerar cada Objeto possível que pode ser colocado em um mapa. Seguir as boas práticas descritas no "Ajustando o Agente de Dimensionamento de Cache para Estimativas Exatas de Consumo de Memória" na página 440 aprimorará a exatidão do tamanho em medidas de bytes fornecidas por WebSphere eXtreme Scale.

Capítulo 11. Resolução de Problemas

Além dos logs e do rastreio, de mensagens e notas sobre o release discutidos nesta seção, é possível usar ferramentas de monitoramento para descobrir problemas como o local de dados no ambiente, a disponibilidade de servidores na grade e assim por diante. Se você estiver executando um ambiente do WebSphere Application Server, poderá usar a Performance Monitoring Infrastructure (PMI). Se você estiver executando em um ambiente independente, poderá usar uma ferramenta de monitoramento do fornecedor, como CA Wily Introscope ou Hyperic HQ. Também é possível usar e customizar o utilitário de amostra xsAdmin para exibir informações de texto sobre o ambiente.

Logs e Rastreio

É possível utilizar logs e rastreio para monitorar e solucionar problemas em seu ambiente. Os logs estão em locais diferentes dependendo da sua configuração. Pode ser necessário fornecer rastreio para um servidor quando você trabalha com o suporte IBM.

Logs com o WebSphere Application Server

Consulte o WebSphere Application ServerCentro de Informações para obter mais informações.

Logs com o WebSphere eXtreme Scale em um Ambiente Independente

Com os servidores de catálogos e contêineres independentes, é possível configurar o local dos logs e qualquer especificação de rastreio. Os logs do servidor de catálogo estão no local onde você executou o comando do servidor de início.

Configurando o local de log para os servidores de contêiner

Por padrão, os logs para um contêiner estão no diretório onde o comando do servidor foi executado. Se você iniciar os servidores no diretório *<eXtremeScale_home>*/bin, os arquivos de log e de rasteio estão nos diretórios logs/*<server_name>* no diretório bin. Para especificar um local alternativo para os logs do servidor de contêineres, como o arquivo server.properties, com os seguintes conteúdos:

```
workingDirectory=<directory>
traceSpec=
systemStreamToFileEnabled=true
```

A propriedade workingDirectory é o diretório-raiz para os logs e o arquivo de rastreio opcional. O WebSphere eXtreme Scale cria um diretório com o nome do servidor de contêineres com um arquivo SystemOut.log, um arquivo SystemErr.log e um arquivo de rastreio se o rastreio foi ativado com a opção traceSpec. Para usar um arquivo de propriedades durante a inicialização de contêiner, use a opção **-serverProps** e forneça o local do arquivo de propriedades do servidor.

Consulte “Iniciando Servidores WebSphere eXtreme Scale Independentes” na página 328 e “Script startOgServer” na página 334 para obter informações adicionais.

As mensagens de informações comuns a serem procuradas no arquivo `SystemOut.log` são o início das mensagens de confirmação. Para obter mais informações sobre uma mensagem específica, consulte “Mensagens” na página 450.

Rastreio com o WebSphere Application Server

Consulte o WebSphere Application ServerCentro de Informações para obter mais informações.

Rastreio no Serviço de Catálogo

É possível configurar o rastreio em um serviço de catálogo usando os parâmetros **-traceSpec** e **-traceFile** durante a inicialização do serviço de catálogo. Por exemplo:

```
startOgServer.sh catalogServer -traceSpec
ObjectGridPlacement=all=enabled -traceFile
/home/user1/logs/trace.log
```

Se você iniciar o serviço de catálogo no diretório *<eXtremeScale_home>*`/bin`, os arquivos de log e de rasteio estarão no diretório `logs/`*<catalog_service_name>* no diretório `bin`. Consulte o “Iniciando o Serviço de Catálogo em um Ambiente Independente” na página 329 para obter mais detalhes sobre o início de um serviço de catálogo.

Rastreio em um servidor de contêineres Independente

É possível ativar o rastreio em um servidor de contêiner de duas formas. É possível criar um arquivo de propriedades do servidor conforme explicado na seção de logs ou é possível ativar o rastreio utilizando a linha de comandos na inicialização. Para ativar o rastreio de contêiner com um arquivo de propriedades do servidor, atualize a propriedade **traceSpec** com a especificação de rastreio necessária. Para ativar o rastreio de contêiner utilizando parâmetros de início, utilize os parâmetros **-traceSpec** e **-traceFile**. Por exemplo:

```
startOgServer.sh c0 -objectGridFile ../xml/myObjectGrid.xml
-deploymentPolicyFile ../xml/myDepPolicy.xml -catalogServiceEndpoints
server1.rchland.ibm.com:2809 -traceSpec
ObjectGridPlacement=all=enabled -traceFile /home/user1/logs/trace.log
```

Se você iniciar o servidor no diretório *<eXtremeScale_home>*`/bin`, os arquivos de log e de rasteio estarão nos diretórios `logs/`*<server_name>* no diretório `bin`. Consulte o “Iniciando Processos do Contêiner” na página 331 para obter mais detalhes sobre o início de um processo de contêiner.

Rastreio com a Interface ObjectGridManager

Outra opção é configurar o rastreio durante o tempo de execução em uma interface ObjectGridManager. A configuração de um rastreio em uma interface ObjectGridManager pode ser usada para obter rastreio em um cliente eXtreme Scale enquanto ele se conecta com um eXtreme Scale e confirma as transações. Para configurar o rastreio em uma interface ObjectGridManager, forneça uma especificação de rastreio e um log de rastreio.

```
ObjectGridManager manager= ObjectGridManagerFactory.getObjectGridManager();
...
manager.setTraceEnabled(true);
manager.setTraceFileName("logs/myClient.log");
manager.setTraceSpecification("ObjectGridReplication=all=enabled");
```

Ativando o Rastreio com o Utilitário xsadmin

Para ativar o rastreio com o utilitário xsadmin, use a opção **setTraceSpec**. Use o utilitário xsadmin para ativar o rastreio em um ambiente independente durante o

tempo de execução e não durante a inicialização. É possível ativar o rastreio em todos os servidores e serviços de catálogo ou filtrar os servidores com base no nome do ObjectGrid, e assim por diante. Por exemplo, para ativar o rastreio ObjectGridReplication com acesso ao servidor de serviço de catálogo, execute:

```
<eXtremeScale_home>/bin>xsadmin.bat -setTraceSpec "ObjectGridReplication=all=enabled"
```

Também é possível desativar o rastreio ao configurar a especificação de rastreio para `*=all=disabled.`:

Consulte o "Monitorando com o Utilitário de Amostra xsAdmin" na página 385 para obter mais informações.

Diretório e Arquivos ffdc

Os arquivos FFDC servem para que o suporte IBM auxilie na depuração. Estes arquivos podem ser solicitados pelo suporte IBM se ocorrer um problema.

Esses arquivos aparecem no diretório `ffdc` e contêm arquivos semelhantes ao seguinte:

```
server2_exception.log
server2_20802080_07.03.05_10.52.18_0.txt
```

Opções de Rastreio

É possível ativar o rastreio para fornecer informações sobre o seu ambiente para o suporte IBM.

Sobre o Rastreio

O rastreio do WebSphere eXtreme Scale é dividido em vários componentes diferentes. Assim como o rastreio do WebSphere Application Server, é possível especificar o nível de rastreio a ser utilizado. Os níveis comuns de rastreio incluem: all, debug, entryExit e event.

Um exemplo de cadeia de rastreio é o seguinte:

```
ObjectGridComponent=level=enabled
```

É possível concatenar as cadeias de rastreio. Use o sinal de asterisco (`*`) para especificar um valor de curinga, como `ObjectGrid*=all=enabled`. Se for necessário fornecer um rastreio para o suporte IBM, uma cadeia de rastreio específica será solicitada. Por exemplo, se ocorrer um problema com a replicação, a cadeia de rastreio `ObjectGridReplication=debug=enabled` pode ser solicitada.

Especificação de Rastreio

ObjectGrid
Mecanismo de cache principal geral.

ObjectGridCatalogServer
Serviço de catálogo geral.

ObjectGridChannel
Comunicações de topologia de implementação estática.

7.1+ **ObjectGridClientInfo**
Informações do cliente do DB2.

7.1+ **ObjectGridClientInfoUser**
Informações sobre o usuário do DB2.

ObjectgridCORBA
Comunicações de topologia de implementação dinâmica.

ObjectGridDataGrid
A API do AgentManager.

ObjectGridDynaCache
O provedor de cache dinâmico do WebSphere eXtreme Scale.

ObjectGridEntityManager
A API do EntityManager. Utilize com a opção Projector.

ObjectGridEvictors
Evictors integrados do ObjectGrid.

ObjectGridJPA
Carregadores do Java Persistence API (JPA).

ObjectGridJPACache
Plug-ins do Cache JPA

ObjectGridLocking
Gerenciador de bloqueios de entrada de cache do ObjectGrid.

ObjectGridMBean
Beans de gerenciamento.

7.1+ **ObjectGridMonitor**
Infraestrutura de monitoramento histórico.

ObjectGridPlacement
Serviço de disposição de shards do servidor de catálogos.

ObjectGridQuery
Consulte ObjectGrid.

ObjectGridReplication
Serviço de replicação.

ObjectGridRouting
Detalhes de roteamento do cliente/servidor.

ObjectGridSecurity
Rastreio de segurança.

ObjectGridStats
Estatísticas do ObjectGrid.

ObjectGridStreamQuery
API de Consulta do Fluxo.

ObjectGridWriteBehind
Atributo write-behind do ObjectGrid.

Projector
O mecanismo com a API do EntityManager.

QueryEngine
O mecanismo de consulta para a API de Consulta do Objeto e a API de Consulta do EntityManager.

QueryEnginePlan
Diagnósticos do plano de consulta.

IBM Support Assistant for WebSphere eXtreme Scale

É possível usar o IBM Support Assistant para coletar dados, analisar sintomas e acessar informações do produto.

IBM Support Assistant Lite

O IBM Support Assistant Lite for WebSphere eXtreme Scale fornece uma coleta de dados automática e suporte à análise de sintomas para cenários de determinação de problemas.

O IBM Support Assistant Lite reduz o período de tempo que leva para reproduzir um problema com os níveis de rastreamento Reliability, Availability and Serviceability adequados configurados (os níveis de rastreamento são automaticamente configurados pela ferramenta) para simplificar a determinação de problemas. Se você precisar de assistência adicional, o IBM Support Assistant Lite também reduz o esforço necessário para enviar as informações de log apropriadas para o IBM Support.

O IBM Support Assistant Lite é incluído em cada instalação do WebSphere eXtreme Scale Versão 7.1.0

IBM Support Assistant

O IBM® Support Assistant (ISA) fornece acesso rápido a recursos do produto, de educação e de suporte que podem ajudar você a responder questões e resolver sozinho problemas com os produtos de software IBM, sem precisar entrar em contato com o IBM Support. Plug-ins diferentes específicos do produto permitem customizar o IBM Support Assistant para os produtos particulares que você instalou. O IBM Support Assistant também podem coletar dados do sistema, arquivos de log e outras informações para ajudar o IBM Support a determinar a causa de um problema particular.

O IBM Support Assistant é um utilitário a ser instalado em sua estação de trabalho, não diretamente no sistema do servidor WebSphere eXtreme Scale em si. Os requisitos de memória e de recurso para o Assistant podem afetar negativamente o desempenho do sistema do servidor WebSphere eXtreme Scale. Os componentes de diagnóstico móveis incluídos são projetados para um impacto mínimo para a operação normal de um servidor.

É possível usar o IBM Support Assistant para ajudá-lo das seguintes maneiras:

- Para pesquisar em fontes de conhecimento e de informações IBM e não IBM em vários produtos IBM para responder uma questão ou resolver um problema
- Para localizar informações adicionais por meio de recursos da Web específicos do produto; incluindo páginas iniciais do produto e de suporte, grupos de notícias e fóruns de clientes, qualificações e recursos de treinamento e informações sobre como resolver problemas e as perguntas mais comuns
- Para aumentar sua capacidade para diagnosticar problemas específicos do produto com as ferramentas de diagnóstico de destino disponíveis no Support Assistant

- Para simplificar a coleta de dados de diagnóstico para ajudar você e a IBM a resolver seus problemas (coletar dados gerais ou dados específicos do produto/sintoma)
- Para ajudar no relatório de incidentes de problemas ao IBM Support por meio de uma interface online customizada, incluindo a capacidade de conectar os dados de diagnóstico mencionados acima ou qualquer outra informação a incidentes novos ou existentes

Finalmente, é possível usar o recurso Updater integrado para obter suporte para produtos de software e recursos adicionais assim que eles forem disponibilizados. Para configurar o IBM Support Assistant para ser usado com o WebSphere eXtreme Scale, primeiro instale o IBM Support Assistant usando os arquivos fornecidos na imagem transferida por download da página da Web Visão Geral do IBM Support em: http://www-947.ibm.com/support/entry/portal/Overview/Software/Other_Software/IBM_Support_Assistant. A seguir, use o IBM Support Assistant para localizar e instalar qualquer atualização do produto. Também é possível optar por instalar plug-ins disponíveis para outro software IBM em seu ambiente. Mais informações e a versão mais recente do IBM Support Assistant estão disponíveis a partir da página da Web do IBM Support Assistant em: http://www.ibm.com/software/support/isa/.

Mensagens

Quando encontrar uma mensagem em um log ou em outras partes de uma interface de produto, será possível procurar pela mensagem pelo prefixo do componente para obter mais informações.

Localizando Mensagens

Ao encontrar uma mensagem em um log, copie o número da mensagem com seu prefixo de letra e número e procure no centro de informações (por exemplo, `CWOBJ1526I`). Ao procurar pela mensagem, será possível encontrar uma explicação adicional da mensagem e possíveis ações a serem tomadas para resolver o problema.

Consulte o centro de informações para obter um índice de mensagens do produto.

Notas sobre o Release

São fornecidos links para o Web site de suporte do produto, para documentação do produto e para atualizações de última hora, limitações e problemas conhecidos para o produto.

- "Acessando Últimas Atualizações, Limitações e Problemas Conhecidos"
- "Acessando Requisitos de Software e de Sistema" na página 451
- "Acessando a Documentação do Produto" na página 451
- "Acessando o Web Site de Suporte do Produto" na página 451
- "Entrando em Contato com o Suporte ao Software IBM" na página 451

Acessando Últimas Atualizações, Limitações e Problemas Conhecidos

As notas sobre o release estão disponíveis como notas técnicas no site de suporte do produto. Para ver uma lista de todas as notas técnicas do WebSphere eXtreme Scale, vá para a Página da Web de Suporte. Clicar nos links fornecidos aqui

resultará em uma pesquisa, na página da Web de Suporte, das notas relevantes sobre o release, que serão retornadas como uma lista.

- 7.1+ Para visualizar uma lista das notas sobre o release para a Versão 7.1, vá para a Página da Web de Suporte.
- Para visualizar uma lista de notas sobre o release para a Versão 7.0, vá para a Página da Web de Suporte.
- Para visualizar uma lista das notas sobre o release para a Versão 6.1, vá para a página da wiki Notas sobre o Release.

Acessando Requisitos de Software e de Sistema

Os requisitos de hardware e software são documentados nas páginas a seguir:

- Requisitos do Sistema Detalhados

Acessando a Documentação do Produto

Para obter o conjunto de informações inteiro, acesse apágina da Biblioteca.

Acessando o Web Site de Suporte do Produto

Para procurar as notas técnicas, downloads, correções e outras informações relacionadas a suporte mais recentes, acesse apágina de Suporte.

Entrando em Contato com o Suporte ao Software IBM

Se você encontrar um problema com o produto, primeiro, tente as seguintes ações:

- Siga as etapas descritas na documentação do produto
- Procure a documentação relacionada na ajuda on-line
- Consulte as mensagens de erro na referência de mensagens

Se você não conseguir resolver o problema com nenhum dos métodos anteriores, entre em contato com o Suporte Técnico IBM.

Avisos

Referências nesta publicação a produtos, programas ou serviços IBM não significam que a IBM pretende torná-los disponíveis em todos os países onde opera. Qualquer referência a produtos, programas ou serviços IBM não significa que apenas produtos, programas ou serviços IBM possam ser utilizados. Qualquer produto, programa ou serviço funcionalmente equivalente, que não infrinja nenhum direito de propriedade intelectual da IBM poderá ser utilizado em substituição a este produto, programa ou serviço. A avaliação e verificação da operação em conjunto com outros produtos, exceto aqueles expressamente designados pela IBM, são de inteira responsabilidade do Cliente.

A IBM pode ter patentes ou solicitações de patentes pendentes relativas a assuntos tratados nesta publicação. O fornecimento desta publicação não lhe garante direito algum sobre tais patentes. Pedidos de licença devem ser enviados, por escrito, para:

Gerência de Relações Comerciais e Industriais da IBM Brasil
Av. Pasteur, 138-146
Botafogo
Rio de Janeiro, RJ
CEP 22290-240

Licenciados deste programa que desejam obter informações sobre este assunto com objetivo de permitir: (i) a troca de informações entre programas criados independentemente e outros programas (incluindo este) e (ii) a utilização mútua das informações trocadas, devem entrar em contato com:

Gerência de Relações Comerciais e Industriais da IBM Brasil
Av. Pasteur, 138-146
Botafogo
Rio de Janeiro, RJ
CEP 22290-240

Tais informações podem estar disponíveis, sujeitas a termos e condições apropriadas, incluindo em alguns casos o pagamento de uma taxa.

Marcas Registradas

Os termos a seguir são marcas registradas da IBM Corporation nos Estados Unidos e/ou em outros países:

- AIX
- CICS
- Cloudscape
- DB2
- Domino
- IBM
- Lotus
- RACF
- Redbooks
- Tivoli
- WebSphere
- z/OS

Java e todas as marcas registradas baseadas em Java são marcas registradas da Sun Microsystems, Inc. nos Estados Unidos e/ou em outros países.

LINUX é uma marca registrada de Linus Torvalds nos Estados Unidos e/ou em outros países.

Microsoft, Windows, Windows NT® e o logotipo do Windows são marcas registradas da Microsoft Corporation nos Estados Unidos e/ou em outros países.

UNIX é uma marca registrada do The Open Group nos Estados Unidos e em outros países.

Outros nomes de empresas, produtos e serviços podem ser marcas registradas ou marcas de serviços de terceiros.

Índice Remissivo

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Impresso no Brasil