iSeries 零售商店方案应用程序

版本 5.1 Windows 版

iSeries 零售商店方案应用程序

版本 5.1 Windows 版

目录

第 1 章 iSeries 零售商店方案应用程序简介

该方案包是使用 IBM WebSphere Development Studio Client for iSeries（用于 iSeries 服务器的基于 Eclipse 的技术）开发的样本应用程序。该方案适用于想要将 iSeries 服务器用于下列目的的开发者:

- Java 开发
- Web 开发
- 管理 RPG 代码并将 RPG 代码转换为 Web 应用程序

该应用程序由复原至 iSeries 服务器的两个 .savf 文件和导入至客户机产品的五个 .zip 文件组成。它们构成一系列的 Web 页面，这些页面具有指向进程中的各个点的 URL。

重要提示: 查阅我们的支持页面以获取所有服务包、PTF 和其它信息: **ibm.com**/software/awdtools/wdt400/support/。

此方案指导您完成产品的各个部分，并且重点在于特定于 iSeries 的组件，例如:

- IBM WebFacing Tool
- Web 服务
- 用于 iSeries 的 Web 开发工具（包括 “Web 交互作用” 向导和 “程序调用” 向导）
- 用于 iSeries 的 Java 开发工具
- IBM Toolbox for Java

该方案说明两家公司（一个批发商和一个服装零售商店）的情形，其中这两家公司是商业伙伴，并且都将它们的 iSeries 服务器用于业务逻辑和数据。在过去，双方的业务是通过电子邮件、电话和传真来互相开展以检查库存、提交订单并跟踪订单直至完成。现在，这两家公司希望使用 Web 来完成常规业务交易。

零售商店想要有能够实现以下目的的 Web 站点:

- 客户能够使用它来采购产品
- 职员能够使用它从批发商处订购存货

批发商想要:

- 在线接收零售商订单以进行跟踪
- 为多个潜在的客户提供服务

在此方案中，您将充当这两家公司的编程顾问的角色，帮助它们将业务移至 Web 上。

根据用户的类型，应用程序具有两个不同的入口点。作为客户，您通过查看商店所能提供的产品（在本示例中为休闲服装）进行。如果想要采购，可以单击链接来访问订购屏幕。在订购之后，会生成一个摘要页面，您可以继续购物、取消订单或提交订单。

作为管理员，您具有应用程序的安全用户标识，该标识需要在部署期间为应用程序定义安全策略。管理员的入口点是登录屏幕，在此屏幕中可以查看订单、查看库存并从批发商处采购。可以选择商品、查看最新批发价和订购想要的尺码和数量。应用程序验证批发商是否有您想要的尺码和数量，并确认订单或告诉您此时不能履行订单。

在应用程序背后，许多操作在产品的各个部分中发生。下表说明了该产品中负责应用程序的每个部分的进程和组件。接着是有关如何执行每个任务的详细信息。

表 1.

客户应用程序任务	管理员应用程序任务	底层进程
显示产品价格		使用 iSeries RPG 程序来创建 Web 服务并使用 Web 开发工具以查看和显示价格。
商店下达订单		将 servlet 和 JSP 文件与 iSeries Java 开发工具、iSeries Web 开发工具和 IBM Toolbox for Java 配合使用来访问和查看 iSeries 服务器上的库存以及订购和显示采购摘要。
	查看库存。	使用 IBM WebFacing Tool 来将现有 RPG 程序转换为 Web 应用程序并使用 Web 开发工具来定制 Web 页面。
	通过提供商品标识和数量来从批发商处订购。	创建当单击**采购**按钮时调用的 Web 服务。
查看商店的初始 Web 页面	查看用于订购存货的初始 Web 页面。	使用用于 iSeries 的 Web 开发工具来创建双方的主页。

基于项目的方案

iSeries 方案应用程序由五个项目组成，它们名为 SV000501、SV000514、SV000618、SV001585 和 SV001586。本指南包含有关如何运行预封装的应用程序的指示信息。本指南还包含基本模块和高级模块，这些模块显示如何自己构建各种项目。基本模块适用于相对而言不熟悉应用程序开发和 Development Studio Client 的开发者。高级模块适用于在应用程序开发方面有经验且较熟悉 Development Studio Client 的开发者。

注: 虽然有五个项目，但只有四个模块，原因在于项目 SV001586 和 SV000514 是组合式“Web 服务”模块。

本方案以五个项目的形式出现:

SV000501 项目: 创建 Web 项目来查看未完成的订单、库存和产品详细信息 – 此项目是使用 iSeries Web 开发工具和 IBM WebFacing Tool 创建的，并且它是为此类 RPG 程序员设计的: 即他们了解有限的 Web 应用程序开发但又想使用 IBM WebFacing Tool 来将他们的 RPG 应用程序放到 Web 上。

SV001585 项目: 在 iSeries 服务器上创建生成客户订单的 HTML 代码、servlet 和 JSP 文件 – 此项目使用 IBM Toolbox for Java 的 SQL 和 JDBC 类、iSeries Java 开发工具的 RecordIOManager bean 和 iSeries “程序调用” 向导。这些元素显示访问和处理 iSeries 服务器上存在的数据和程序的各种方法。此项目是为想要开发 Web 页面来访问 iSeries 数据和代码的 Java 程序员和 Web 应用程序开发者设计的。另外，您还应具备 iSeries 服务器管理和 RPG 编程的应用知识。

SV000514 项目: 创建 iSeries Web 服务来提供产品价格 – “Web 服务” 向导使用 iSeries “程序调用” 向导生成的 Java bean 来调用 iSeries 服务器上的一个或多个程序

过程，并将信息传送回浏览器。此项目是为想要使用 Web 服务来创建独立的模块化应用程序（这些应用程序可通过万维网来进行描述、发布、定位和调用）的 RPG 程序员设计的。

SV001586 项目：创建 Web 服务来通过 iSeries 服务器向批发商下订单 – Web 服务接受商品标识和所需数量，然后向批发商下订单。此项目是 SV000514 的一个组件，是为想要创建 Web 服务的 RPG 程序员设计的。

SV000618 项目：创建 Web 项目来建立订单表单、库存表单和 IBM WebFacing Tool 生成的采购订单之间的接口 – 此项目需要 iSeries Web 开发工具，并且涉及到创建 HTML 和 JSP 文件以便使用和连接在 SV000514 和 SV001586 中开发的 Web 服务。该项目是为想要使用 Web 服务且具备 RPG 和 Java 编程知识的开发者设计的。

第 2 章 技术概念

为了逐步执行方案应用程序，需要熟悉许多技术概念，当您不熟悉 Web 应用程序开发时更是如此。以下是逐步执行应用程序时将会遇到的某些内容的简要列表。

企业归档文件（EAR）

EAR 文件是具有 .ear 扩展名的标准 “Java 归档”（JAR）文件。它们可以包含多个 Web 项目，并且您可以使用它们来将 Web 应用程序封装和部署至 WebSphere Administrative Server（WAS）。**注意**: 在 J2EE SDK 应用程序部署工具的 GUI 版本中，先创建一个 EAR 文件，然后将 JAR 和 “Web 归档”（WAR）文件添加至该 EAR 文件。但是，如果您使用命令行包装器工具，则先创建 JAR 和 WAR 文件，然后创建 EAR 文件。

IBM WebFacing Tool

IBM WebFacing Tool 将现有的 5250 界面转换为基于浏览器的图形用户界面。只需要对原始 iSeries 应用程序进行很少的修改或不进行修改，就可以将程序的使用扩展到因特网或内部网。

Java 归档文件（JAR）

JAR 文件是 Java 文件的压缩包，类似于 .zip 文件。它包含 Java applet 的类、图像和声音文件，它们被收集到单个文件中并被压缩以便快速下载至浏览器。

Java Server Pages（JSP）

JSP 提供在静态 HTML 页面中显示动态内容的功能。用 Java 编写的 JSP 与服务器和平台无关。通过有效地将 Web 表示与 Web 内容分开，JSP 可以帮助开发者快速更改他们的 Web 页面的设计和显示。

程序调用 bean

这些是 “程序调用” 向导生成的 Java bean。其中一种类型是 Java 应用程序使用的常规 Java bean。其它类型可由 Web 服务向导用于创建 Web 服务。

“程序调用” 向导

“程序调用” 向导帮助您创建调用 iSeries 程序或过程所需的 Java bean 及相关联的 PCML 文件。此向导会提示您输入关于程序或服务程序对象的信息以及对象的参数，然后创建期望的 Java bean（和 PCML 文件）。

报告程序生成器（RPG）

iSeries 程序员使用的过程化编程语言。可以使用 RPG 来创建业务应用程序，例如，开发票程序和订单输入程序。最新版本 ILE RPG IV 扩展了 RPG 语言的能力，并支持程序员在先前版本中的经验有用武之地。

Servlet

用 Java 编写的服务器端程序，它在支持 Java 的服务器或应用程序服务器（例如，IBM WebSphere Application Server）中运行。Servlet 执行服务器指定的任务，例如，通过生成 HTML 响应来响应请求。例如，可以在将数据发送到服务器的同时，在在线银行应用程序中使用 servlet 来响应用户。

Web 组件

可以使用“Web 组件”来定义 iSeries 对象（例如，数据输入字段和按钮），这些对象可以在 iSeries 服务器程序与 Web 页面之间交换信息。开发者可以使用“Web 组件”来捕获用户事件，例如，输入字段的语法检查和单击按钮。

“Web 交互作用”向导

此向导是 iSeries Web 开发工具的一部分。它创建和管理 ILE 程序与 Web 页面之间的交互作用。该向导控制在何处显示输入、输出和错误消息，并将数据从输入和输出字段定向到 ILE 程序。还可以使用“Web 交互作用”向导来将错误消息映射至发生错误的区域，以便用户可以容易地标识错误的起源。

Web 服务

Web 服务是为了在因特网上使用而设计和实现的自包含应用程序。它们是使用开放标准（例如，SOAP、WSDL 和 XML）创建的。您可以对许多业务情况（其中包括一个库存管理系统，在该系统中客户机可以通过因特网检查它们的库存级别）使用 Web 服务，或者在您想要直接从供应商处跟踪产品订单时也可以使用 Web 服务。

Web 服务定义语言（WSDL）

WSDL 是基于 XML 的语言，它定义 Web 服务的接口。WSDL 理解 Web 服务并管理 Web 服务与服务器程序之间的信息流。例如，开发者会使用 WSDL 来为显示更新的股市行情的 Web 站点创建接口。

WebSphere Studio Workbench

IBM WebSphere Development Studio Client for iSeries 是在 WebSphere Studio Workbench（IBM 的 Eclipse 平台的实现）上构建的。这种可扩展的通用工作台集成了构建和维护应用程序所需的所有工具。开发者可以使用 Development Studio Client 来使用插件将新对象合并到开发环境中，并且可以无缝地添加 Java 文件、图形和视频等等。

第 3 章 运行方案

可以在 Development Studio Client 工作台中或在任何平台（包括 iSeries 平台）版本的 WebSphere Application Server 上运行 *Wholesale* 和 *Retailstore* 应用程序。有关这些应用程序的概述，参见第 1 页的第 1 章，『iSeries 零售商店方案应用程序简介』。

在本章整个内容中，您应该能够:

- 将样本文件和 Development Studio Client 对象复原至 iSeries 服务器
- 创建 Web 项目和 WebFacing 项目以保存文件
- 将复原的文件从 iSeries 服务器导入至 Development Studio Client
- 配置 WebSphere Application Server（WAS）
- 配置 WebFacing 项目
- 在“WebSphere 测试环境”中作为客户运行应用程序
- 在“WebSphere 测试环境”中作为管理员运行应用程序
- 配置 WebSphere Application Server
- 保护管理员页面
- 创建 EAR 文件以进行外部 WAS 部署
- 将应用程序部署至 WAS
- 在 WAS 中运行应用程序

在开始之前

要从工作台测试应用程序，需要确保:

- 具有 V5R1 或更高版本的 iSeries 服务器
- 已应用最新的 WebFacing PTF。请参阅我们的支持页面以获取 PTF: http://www.ibm.com/software/awdtools/wdt400/support/
- 对 iSeries 服务器有 NET USE 访问权

安装样本文件

要使用 iSeries 方案应用程序，需要使用下列文件:

- Wholesale.savf
- Retailstor.savf
- Qdtssfl.savf
- SV000501.zip
- SV000514.zip
- SV000618.zip
- SV001585.zip
- SV001586.zip

.savf 文件包含 iSeries 数据和 RPG 程序，.zip 文件包含与 iSeries 程序交互以处理 iSeries 数据的 Web 应用程序。首先，需要将 .savf 文件复原至 iSeries 服务器，然后可以将 .zip 文件导入到 Development Studio Client 中并在工作台中运行应用程序。

复原 .savf 文件

要使用本指南中的样本，需要将 WHOLESALE 和 RETAILSTOR 库复原至 iSeries 服务器。即使对该产品的先前发行版已经复原过这两个库，您也应该执行此操作，这是因为它们的内容并不相同。

注: 用于安装样本库的 .savf 文件与 V5R1 或更高版本的 iSeries 服务器配合使用。由于考虑到此方案，将两个库复原到同一个 iSeries 服务器，但如果您要为实际业务开发此应用程序，则将这两个库复原至不同的 iSeries 服务器。将把 WHOLESALE 库复原到提供 Web 服务的 iSeries 服务器，把 RETAILSTOR 库复原到属于零售商店的 iSeries 服务器。

要复原 Wholesale.savf 文件:

1. 通过绿屏仿真器登录到 iSeries 服务器。
   a. 创建要包含保存文件的库。要在仿真器中创建新的库，输入 CRTLIB。
   b. 将库命名为 SCENARIO。
   c. 按 Tab 跳到下一行；指定 *TEST 作为库类型并按 Enter 键以保存更改。
   d. 使用 CRTSAVF 命令创建两个保存文件并在以下两行之间按 Enter:

      ```
      CRTSAVF FILE(SCENARIO/WHOLESALE)
      CRTSAVF FILE(SCENARIO/RETAILSTOR)
      ```

      这些行指定您想要在 Scenario 库中创建保存文件。
2. 在工作站上，打开“命令提示符”窗口。
   a. 您需要切换至 .savf 文件所在的目录。在缺省情况下，应输入 cd c:\wdsc\wdscsampl。如果已将该产品安装至另一驱动器，或者您选择不使用“wdsc”作为 Development Studio Client 的主目录，则在适当的位置切换至 wdscsampl 目录（您在其中安装该产品）。
   b. 在命令行上，输入: ftp *hostname*，其中 *hostname* 是 iSeries 服务器的名称（例如，PROD400）。
   c. 输入您用于 iSeries 服务器的用户标识和密码。
   d. 在命令行上，输入 cd /qsys.lib/scenario.lib 以切换到 Scenario 库。
   e. 输入下列行:

      ```
      bin
      put WHOLESALE.savf WHOLESALE.savf
      put RETAILSTOR.savf RETAILSTOR.savf
      quit
      ```

      这些行（退出之前）指定您想要从本地系统中取出保存文件并将它们放置到 iSeries 服务器中。
3. 返回到 iSeries 控制台，复原 Wholesale 库:
   a. 输入 RSTLIB 并按 F4 以定义想要如何复原该库。
   b. 在**保存的库**字段中，输入 WHOLESALE 并按 Tab 键。
   c. 在**设备**字段中，输入 *savf 并按 Tab 键
   d. 在下一个字段中按 Enter 键以显示其它值，并按 Tab 键跳到**保存文件**字段。

e. 在**保存文件**字段中，输入 WHOLESALE 并按 Tab 键。

f. 在**库**字段中，删除现有值并输入 scenario。

g. 按 Enter 键以将 WHOLESALE 库复原至 iSeries 服务器。

4. 对 Retailstor.savf 文件重复此过程:

a. 输入 RSTLIB 并按 F4 以定义想要如何复原该库。

b. 在**保存的库**字段中，输入 RETAILSTOR 并按 Tab 键。

c. 在**设备**字段中，输入 *savf 并按 Tab 键。

d. 在下一个字段中按 Enter 键以显示其它值，并按 Tab 键跳到**保存文件**字段。

e. 在**保存文件**字段中，输入 RETAILSTOR 并按 Tab 键。

f. 在**库**字段中，删除现有值并输入 scenario。

g. 按 Enter 键以保存操作并将 RETAILSTOR 库复原至 iSeries 服务器。

修改库列表

尽管已将 RETAILSTOR 和 WHOLESALE 库复原至 iSeries 服务器，但仍然需要确保它们在您登录时使用的库列表中。否则，Web 应用程序可能找不到这些库，并且您在运行或构建应用程序时可能会遇到问题。还需要确保 QGPL 库在库列表中。

为此，需要更新作业描述和用户概要文件，以使对库列表的更改是永久的。

不同的用户有修改他们的库列表的不同方法，但如果您不知道如何修改，请参阅下面的示例:

1. 在 iSeries 服务器上，输入 dsplibl 并按 Enter 键以显示库列表。
2. 检查 RETAILSTOR、WHOLESALE 和 QGPL 库是否在库列表中。如果它们在的话，可退出并关闭仿真器窗口。如果它们不在库列表中，则:

a. 按 F12 取消并退出当前屏幕。

b. 找出作业描述的名称。为此，输入 dspusrprf *username* 并按 Enter 键以显示用户概要文件，其中 *username* 是您用来登录 iSeries 服务器的标识。作业描述在用户概要文件的其中一个页面上（使用“页下移”）。在找到作业描述时，记下它并按 F12 以退出该屏幕。

c. 在命令行处输入 chgjobd 并按 F4。

d. 在**作业描述**字段中，输入作业描述名并按 F10 以查看附加参数。

e. 页下移至**初始库列表**。

f. 在**初始库列表**下面的字段中，将看到“输入加号以获取其它值”。输入加号并按 Enter 键。

g. 在**指定其它值**页面上，删除现有的任何输入并在每个值之间输入 RETAILSTOR、WHOLESALE 和 QGPL。因为应用程序首先需要查找 RETAILSTOR 库，所以必须遵循此顺序输入各个库。按 Enter 键。

h. 再次按 Enter 键以保存更改。

i. 在命令行处输入 chgusrprf 并按 Enter 键。

j. 输入用户名并按 Enter 键。

k. 按 F10 以查看附加参数。

l. 页下移至**作业描述**。

m. 如果作业描述的名称与您使用 dspusrprf 命令找到的作业描述不匹配，输入您使用该命令找到的作业描述的名称并按 Enter 键以保存更改。

在 Development Studio Client 中导入并运行应用程序

为 Development Studio Client 安装的程序文件中有五个压缩文件，缺省情况下位于 c:\wdsc\wdscsampl 目录中。这些文件包括:

- SV000514.zip
- SV000618.zip
- SV001585.zip
- SV001586.zip
- SV000501.zip

将每个文件单独导入到 Development Studio Client 工作台中并对每个文件指定相应的“企业归档”（EAR）文件。EAR 文件是具有 .ear 扩展名的标准“Java 归档”（JAR）文件。它们可以包含多个 Web 项目，并且您可以使用它们来将 Web 应用程序封装和部署至 WebSphere Administrative Server（WAS）。

前四个文件对应于 Web 项目，第五个文件对应于 WebFacing 项目。首先，创建 Web 项目，然后创建 WebFacing 项目。最后，将这些压缩文件的内容导入到五个项目中。

在本地系统上解压缩 .zip 文件

在可将方案 .zip 文件的内容导入到 Development Studio Client 工作台中之前，需要将它们解压缩到单独的目录中，以便可将它们作为文件系统（而不是 .zip 文件）导入。

注: 如果您正在使用的程序不是 WinZip，则使用该程序按您的计划完成任务。

1. 在本地系统上，浏览至 c:\wdsc\wdscsampl。如果选择安装至另一驱动器或另一基本文件夹，c:\wdsc 目录可能会变化。如果是这样的话，浏览至该目录，然后找到 \wdscsampl。
2. 双击以下 .zip 文件: SV000501.zip。
3. 单击“解压缩”按钮并浏览至压缩文件所在的同一文件夹，缺省情况下为 c:\wdsc\wdscsampl。
4. 选择**所有文件**单选按钮，然后单击**解压缩**。
5. 如果查看 wdscsampl 目录，您将会发现创建了 SV000501 文件夹且所有必需的文件都在其中。
6. 既然所有文件都已解压缩到 c:\scenario\SV000501 中，删除 SV000501.zip 文件以便在导入文件系统时不会误选该文件。
7. 对其它四个 .zip 文件（SV000618.zip、SV001585.zip、SV001586.zip 和 SV000514.zip）重复本节中的各个步骤。

创建 Web 项目

在本节中，将为 SV000514、SV000618、SV001585 和 SV001586 创建 Web 项目。对创建每个项目的步骤作了单独的概括。虽然这些步骤看上去可能是重复的，但请记住，它们存在一些细微但很重要的差别，在创建这四个项目时需要记住这一点（即关于哪些 EAR 文件与哪些项目相关联）。

要创建 Web 项目:

1. 从“开始”菜单打开 IBM WebSphere Development Studio Client for iSeries。
2. 在 Development Studio Client 中，通过在菜单栏中单击**窗口 > 打开透视图 > 其它 > Web** 并单击**确定**打开 Web 透视图。
3. 单击**文件 > 新建 > 动态 Web 项目**。
   a. 在**项目名称**字段中，输入 SV000514。
   b. 选择**配置高级选项**复选框并单击**下一步**。
   c. 单击 **EAR 项目**字段旁的**新建**按钮。
   d. 在**项目名**字段中，输入 SVWholeSaleEAR 并在两个对话框上单击**完成**。
4. 单击**文件 > 新建 > 动态 Web 项目**。
   a. 在**项目名称**字段中，输入 SV001586 并单击**下一步**（**配置高级选项**复选框应自动选中）。
   b. 验证已在 **EAR 项目**下拉字段中选择了 SWholeSaleEAR。
   c. 单击**完成**。
5. 单击**文件 > 新建 > 动态 Web 项目**。
   a. 在**项目名称**字段中，输入 SV000618 并单击**下一步**。
   b. 单击 **EAR 项目**字段旁的**新建**按钮。
   c. 在**新项目名称**字段中，输入 SVStoreEAR 并在两个对话框上单击**完成**。
6. 单击**文件 > 新建 > 动态 Web 项目**。
   a. 在**项目名称**字段中，输入 SV001585 并单击**下一步**。
   b. 验证已在 **EAR 项目**下拉字段中选择了 SVStoreEAR。
   c. 单击**完成**。

导入 Web 项目文件

为将文件系统导入到工作空间中，以下过程概述了如何导入 SV000514 中的文件。该过程与其它三个 Web 项目文件系统（SV001586、SV001585 和 SV000618）的过程相同。下一节将说明如何创建和导入 WebFacing 项目。

要导入文件:

1. 单击“浏览”并浏览至
2. 右键单击 **SV000514** 文件夹并选择**导入**。
3. 从**导入**向导中，单击**文件系统**，然后单击**下一步**。
4. 浏览至解压缩方案文件的目录；缺省情况下此目录应为 c:\wdsc\wdscsampl。
5. 单击 **SV000514** 并单击**确定**。
6. 单击**全部选中**以确保选择了 SV000501 文件系统的所有组件。
7. 在**目标文件夹**字段中，输入 **SV000514**（如果缺省情况下未填写的话）。
8. 选择**覆盖现有资源而不发出警告**复选框。
9. 单击**完成**并单击**是**（如果出现询问您有关修改上下文根的任何对话框）。
10. 对 SV001586、SV001585 和 SV000618 重复步骤 2 至 9，在完成这些步骤时一定要使用不同的项目名。

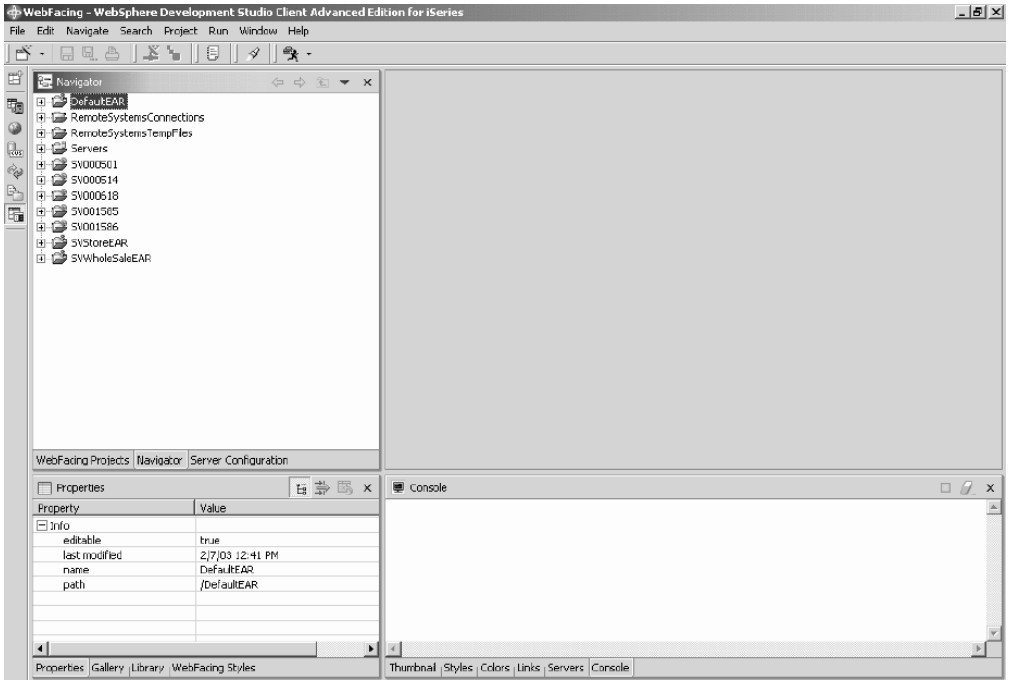
导入 WebFacing 项目文件和创建 WebFacing 项目

因为 SV000501 是一个 WebFacing 项目，所以项目 SV000501 不同于刚创建的四个 Web 项目。由于 WebFacing 项目包括不同的文件，并且具有特定于处理 iSeries 服务器中的 DDS 文件的独特结构，所以它们有所不同。

要导入这些文件和创建 WebFacing 项目:

1. 如果您不在“项目导航器”视图中，则单击**项目导航器**选项卡。
2. 在视图中右键单击并选择**导入**。
3. 在“导入”窗口中，选择 **WebFacing 项目**并单击**下一步**。
4. 浏览至解压缩方案文件的目录；缺省情况下此目录应为 c:\wdsc\wdscsampl。
5. 选择 **SV000501** 并单击**确定**。单击**下一步**。
6. 在窗口的**找到的 WebFacing 项目**区域中，选择 **SV000501** 复选框。
7. 在**企业应用程序项目（EAR）**字段中，输入 SVStoreEAR。
8. 单击**完成**。

既然您已导入了所有文件，您的工作空间看上去应该与下图类似:



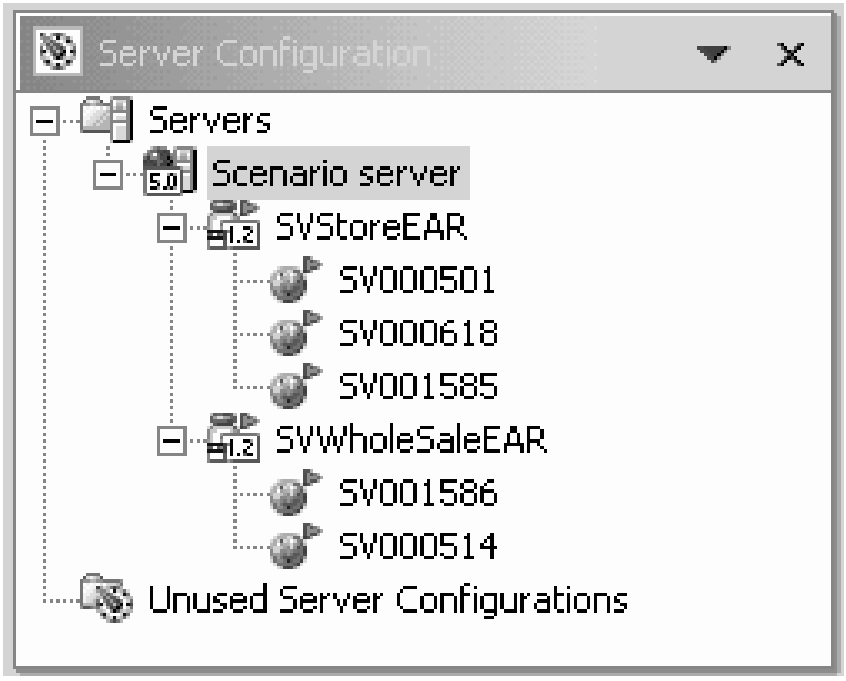
如果看到导入的项目旁有错误（由红色 X 标记），右键单击每个项目并选择**重新构建项目**。根据工作空间中的内容，这应该会清除大多数错误以便应用程序正确运行。可能存在您仍然会见到的任何错误，这是因为某些代码未完全迁移至 5.1 发行版的 Development Studio Client。可继续学习本章中的练习以帮助您了解该产品，或者可继续转至下一章节以尝试从头开始构建应用程序。

服务器配置

既然已导入所有源文件，则需要配置“WebSphere 测试环境”服务器以便它识别 SVWholesaleEAR 和 SVStoreEAR 应用程序。

要配置“WebSphere 测试环境”服务器:

1. 切换到“服务器”透视图。从工作台菜单栏中，单击**窗口 > 打开透视图 > 其它 > 服务器**并单击**确定**。
2. 在“服务器配置”视图中，右键单击**服务器**并选择**新建 > 服务器和服务器配置**。
3. 在**服务器名称**字段中，输入 Scenario server。
4. 在**文件夹**字段中，输入 Scenario folder。
5. 在**服务器类型**框中，展开 **WebSphere V5.0** 并单击**测试环境**（如果缺省情况下尚未选择它）。
6. 如果接收到任何消息，单击**完成**并单击**是**。
7. 要将 EAR 文件添加至服务器配置:
   a. 展开**服务器**。
   b. 右键单击 **Scenario server** 并选择**添加 > SVWholeSaleEAR**。
   c. 展开**服务器**。
   d. 再次右键单击 **Scenario server** 并选择**添加和除去项目**。
   e. 选择单击**全部添加**按钮以将 SVStoreEAR 和 SVWholeSaleEAR 移到窗口右边。
   f. 单击**完成**。
8. 在服务器配置视图中，展开 **Scenario server**，然后展开 **SVStoreEAR** 和 **SVWholeSaleEAR** 以查看列示的所有应用程序:



要确保服务器检取了项目:

1. 在“服务器”视图（在“服务器配置”视图的右边）中（如果看不到“服务器”视图，则单击**服务器**），右键单击 **Scenario server** 并选择**发布**。将出现一个**发布**对话框以显示发布进度。
2. 单击“确定”以将应用程序发布至 WebSphere Application Server 测试环境。
3. 当发布完成后，单击对话框中的**确定**。

4. 在同一个视图中，再次右键单击 **Scenario server** 并单击**启动**。“控制台”视图打开以显示服务器的活动。当您在视图底部看到“服务器 *servername* 打开以执行电子商务”这一行消息时，就表示服务器已启动。向下滚动以查看所有消息。

配置 WebFacing 项目

在运行应用程序或在处理 SV000501 模块之前，需要启动 WebFacing 服务器才能使应用程序运行，并且需要配置项目以便它使用正确的 iSeries 服务器。

要启动 WebFacing 服务器:

1. 打开绿屏 5250 仿真器并使用您的用户标识和密码进行登录。
2. 在命令行中输入 `strtcpsvr *webfacing`。
3. 在屏幕的底部，应该看到消息“WEBFACING 服务器正在启动”。

需要更改 WebFacing 属性，返回到 Development Studio Client 工作台中:

1. 切换到 Web 透视图（可通过单击屏幕左边聚集的图标来在各透视图之间切换）。
2. 单击**项目导航器**选项卡以便可查看项目结构。
3. 展开 **SV000501** 并双击 **Web 部署描述符**。
4. 单击**参数**选项卡。
5. 单击 **WFDefaultHost**。
6. 在**值**字段中，擦除现有值并输入 iSeries 服务器的名称。
7. 从工作台菜单栏中，单击“保存”图标或单击**文件 > 保存 Web 部署描述符**。如果接收到任何消息，则单击**确定**。关闭文件。

定义服务器信息

现在需要为全部五个项目定义 iSeries 服务器信息，并确保将它们配置为使用您的用户标识和密码在 iSeries 服务器上运行。要定义服务器信息:

1. 确保您在 Web 透视图中。
2. 在“项目导航器”视图中，右键单击 **SV000501** 并选择**指定 iSeries Web Tools 运行时配置**。
3. 在 **iSeries 服务器名**字段中，删除现有值并输入 iSeries 服务器的名称。
4. 删除其它现有值并输入您的用户标识和密码。
5. 单击**完成**（在必要时再次单击**完成**）。
6. 对于其它四个项目（SV000514、SV000618、SV001585 和 SV001586）重复步骤 2 至 5。

在工作台中运行应用程序

此时，可以在工作台的“WebSphere 测试环境”中运行应用程序。当对它正确运行感到满意时，可以将 EAR 文件导出至 iSeries 服务器并将它部署至 iSeries WebSphere Application Server。

在工作台中作为客户运行应用程序

作为客户，您将从零售商店的购物 Web 页面开始，浏览各种商品，然后对数量和尺码进行订购。

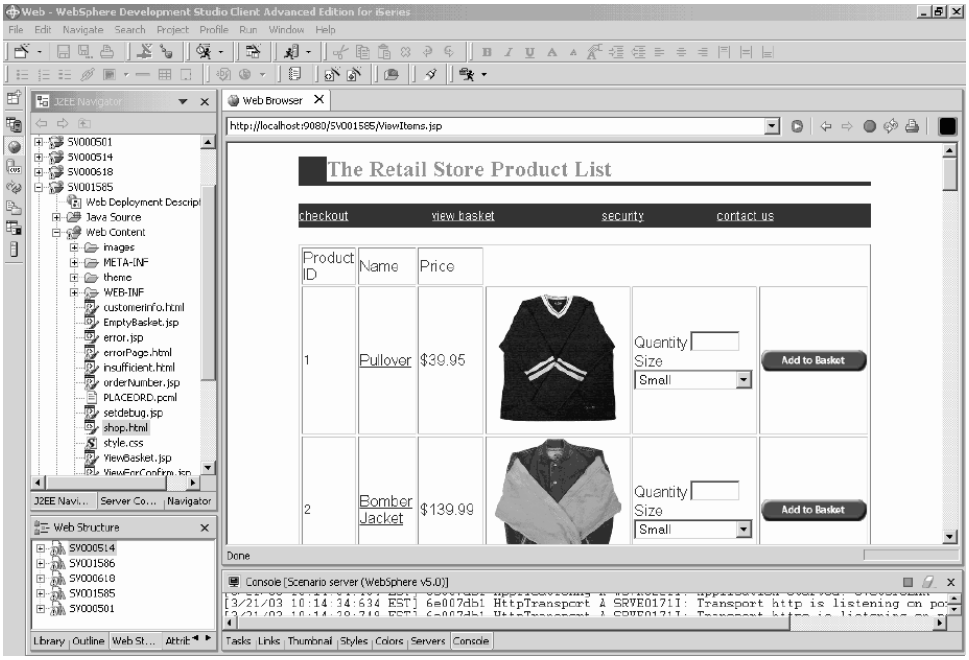
注意: 如果尝试在防火墙后运行应用程序，则可能会遇到问题，原因在于 web.xml 文件会查找以下文件: http://java.sun.com/j2ee/dtds/web-app_2_2.dtd。要解决此问题，在运行应用程序之前，将所有 web.xml 文件中的 DOCTYPE 语句更改为:

```
!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.2//EN"
"x:/WDSC/eclipse/plugins/com.ibm.etools.j2ee/dtds/web-app_2_2.dtd"
```

其中 x:/wdsc 是产品的安装目录。通常可以在 *Project name* **> Web Content > WEB-INF** 下面找到 web.xml 文件。在打开该文件之后，单击**源**选项卡以直接编辑该文件。

要在工作台中作为客户运行应用程序:

1. 在 Web 透视图中，展开 **SV001585 > Web Content**。
2. 右键单击 **shop.html** 并选择**在服务器上运行**。这会在工作台浏览器中启动应用程序。
3. 单击**完成**。
4. 通过单击 T 恤的图像进入应用程序。
5. 尝试在以下页面中输入值，假设您是订购 Pullover 或 Bomber Jacket 的客户，选择尺码和数量并将商品加入购物篮:



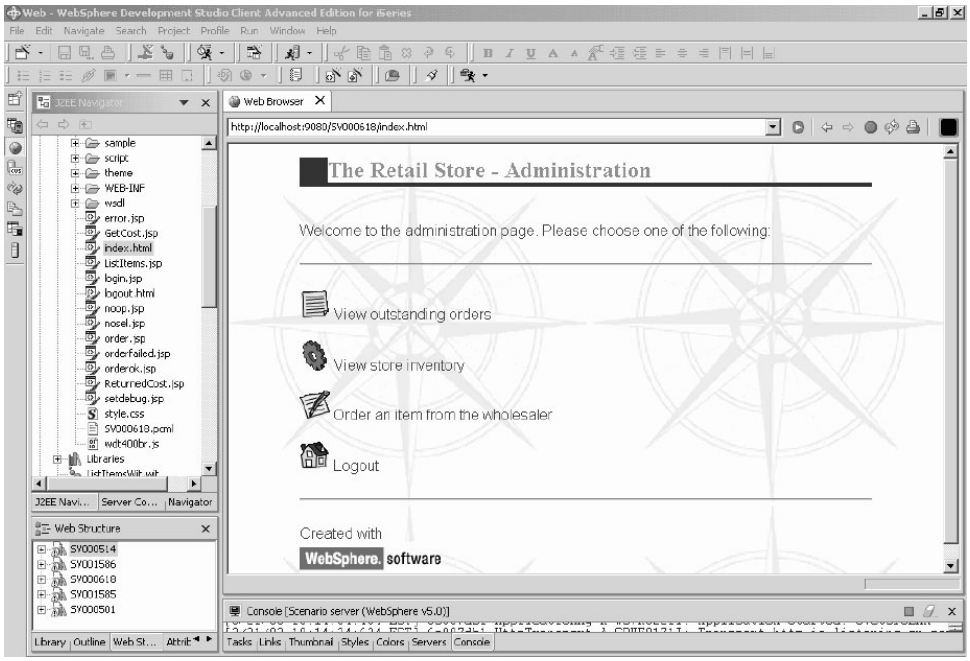
显示文本“已处理您的订单”的页面是样本的最后一个页面。当对样本的完成感到满意时，可以关闭浏览器。

在工作台中作为管理员运行应用程序

作为管理员，您将从零售商店的管理 Web 页面开始，并检查订单和商店库存。要在工作台中作为管理员运行应用程序:

1. 在 Web 透视图中，展开 **SV000618 > Web Content**。
2. 右键单击 **index.html** 并选择**在服务器上运行**。这会在工作台浏览器中启动应用程序。

3. 单击**查看商店库存**左边的图标以显示以下页面，假设您作为管理员决定要为商店采购什么商品:



尝试单击图标以显示管理页面。

将方案应用程序部署至 WebSphere Application Server

既然已在工作台测试环境中运行了 iSeries 方案应用程序，您可以将该应用程序部署至 WebSphere Application Server，就如同您在现实世界中运行该应用程序一样。但是，在部署应用程序之前需要作出一些调整，例如，保护管理员的页面和更改 Web 服务的 URL 以使应用程序指向正确的位置，如下面各节中所述。

注: 部署至 WebSphere Application Server 是可选的；您仍然可以继续转至下一章并完成各模块，而不必在 WebSphere Application Server 中测试应用程序。

配置 WebSphere Application Server

在本章的前面部分，我们讨论了第 14 页的『在工作台中运行应用程序』。既然已将文件部署至 iSeries 服务器，就可以使用 WebSphere Application Server 在 iSeries 上运行该应用程序了。

要将应用程序部署至 WebSphere Application Server（可选），需要确保:

- 已经在 iSeries 服务器上创建了 WebSphere Application Server V4.0、V5.0 或 WebSphere Application Server Express 实例（或具有对它们的访问权），并且该实例正在运行（仅当您想要测试对 WebSphere Application Server 的部署时）。
- 您知道 iSeries 服务器上 HTTP 和 WebSphere Application Server 实例的端口号。
- 已经在工作站上安装了 “WebSphere 管理控制台” 版本 4.0 或版本 5.0。

注: 对于 WAS V5.0，该控制台是基于浏览器的，因此，您的工作站上不需要控制台。参阅在线文档以查找有关各种 WAS 版本的更多信息。

支持 Web 的 iSeries 应用程序使用 WebSphere Application Server 来运行在 Web 用户的浏览器与 iSeries 程序或数据之间进行通信的 Java™ servlet 和 JavaServer Pages™（JSP）。要从同一个 iSeries 系统上提供 HTML 页面和 JSP 文件，在该系统上还需要有 HTTP Server。我们建议您使用 IBM HTTP Server（基于 Apache）。可以在以下位置找到有关如何使用此服务器的文档：IBM HTTP Server for iSeries Documentation Center，网址为 http://publib.boulder.ibm.com/pubs/html/iseries_http/v5r1/index.htm。

WebSphere Application Server 执行为各种过程生成的 JavaServer Pages、Java beans™ 和 Java servlet。IBM WebSphere Application Server for iSeries 和 “IBM WebSphere 管理控制台 iSeries 版” 的主要文档资源可从下列 Web 站点获取：

- IBM WebSphere Application Server V4.0 Advanced Edition for iSeries，网址为 http://publib.boulder.ibm.com/was400/40/AE/english/docs/
- IBM WebSphere Application Server V4.0 Advanced Single Server Edition for iSeries，网址为 http://publib.boulder.ibm.com/was400/40/AEs/english/docs/

强烈建议您熟悉 IBM WebSphere® Application Server 文档，特别是有关『*J2EE 模块*』、『*安装 WebSphere Application Server*』和『*设置 WebSphere 管理服务器的多个实例*』的章节。至少需要执行安装链接下面的步骤。

使用站点映射来查找有关如何为 “WebSphere 管理控制台” 安装、配置和获取必需的 PTF 的信息。

保护管理员页面

由于管理员页面 index.html 只应由已授权的人员访问，所以它应受到正确地保护。可以通过在 Web 应用程序逻辑编程或使用 WebSphere 的安全功能来完成此任务。在此方案中，我们使用 WebSphere 安全性来保护该页面。注意，我们使用 WebSphere Application Server V4.0 Advanced Edition。可以在 WebSphere Application Server Web 站点中找到有关它们的信息：

- IBM WebSphere Application Server V4.0 Advanced Edition for iSeries，网址为 http://publib.boulder.ibm.com/was400/40/AE/english/docs/
- IBM WebSphere Application Server V4.0 Advanced Single Server Edition for iSeries，网址为 http://publib.boulder.ibm.com/was400/40/AEs/english/docs/
- WebSphere 的红皮书页面：在以下网址搜索 WebSphere Application Server V5.0 和 WebSphere Application Server Express Edition V5.0 的红皮书：http://publib-b.boulder.ibm.com/redbooks.nsf/portals/Websphere

如果您正在使用 WebSphere Application Server 的其它版本，则参阅该版本的有关保护 Web 资源的文档。

可以在 Development Studio Client 或在 “应用装配工具” 中为 Web 资源（例如，Web 页面和 servlet）配置安全性。对于本方案，我们使用 Development Studio Client。

要复查此 Web 应用程序的安全性配置和属性：

1. 在 Development Studio Client 中，切换至 Web 透视图。
2. 在 “导航器” 视图中，展开 **SV000618 > Web Content > WEB-INF**。
3. 双击 **web.xml** 以打开 web.xml 视图。

4. 单击**安全性**选项卡。
5. 为了保护 index.html（管理页面），还定义了“安全性约束”。在视图的顶部，单击**安全性约束**。
6. 单击列表中 **SecurityContraint** 的第一个实例。
7. 在右边单击 **AdminPage**。
8. 单击**编辑**以调用 **Web 资源集合**对话框。注意，index.html 的 GET 和 POST 方法是预先选择的。
9. 单击**确定**。

在**安全角色**部分中，请注意名为“Administrator”的已定义安全角色。在部署期间，对个人指定此角色，从而为他们提供对 index.html 页面的访问权。在**已授权的角色**部分中，注意到我们已经为角色“Administrator”提供了对此安全性约束的访问权。在此安全性存在的情况下，只有指定了“Administrator”角色的用户在提供了正确的凭证（如用户标识和密码）之后才有权访问 index.html 页面。当资源受保护时，WebSphere Application Server 首先尝试对用户进行认证。认证是使用证书或通过提示用户输入用户标识和密码来完成的。提示可通过基本认证对话框或使用定制表单完成。

在此方案中，我们设计了自己的名为 login.jsp 的登录页面。要配置其认证提示，选择 web.xml 视图中的**页面**选项卡。在**登录**部分中，请注意**表单**是作为“认证”方法预先选择的。另外还请注意，“登录”页面的名称是 login.jsp。当登录不成功时，就会显示“错误”页面。在这种情况下，应用程序重新显示 login.jsp 页面。

为 iSeries WebSphere Application Server 部署创建 EAR 文件

需要创建 EAR 文件才能将应用程序部署至 iSeries WebSphere Application Server。EAR 文件是具有 .ear 扩展名的标准“Java 归档”（JAR）文件。它们可以包含多个 Web 项目，并且您可以使用它们来将 Web 应用程序封装和部署至 WebSphere Administrative Server（WAS）。

要创建 EAR 文件:

1. 切换到 Web 透视图。
2. 在“导航器”视图中，右键单击 **SVStoreEAR** 并选择**导出**。
3. 在“导出”窗口中，单击 **EAR 文件**并单击**下一步**。
4. 在**要将资源导出至何处?** 下面，单击**浏览**并浏览至 iSeries“集成文件系统”上可以保存 EAR 文件的目录。（需要将网络驱动器映射为 iSeries IFS）。
5. 在**文件名**字段中，输入 `SVRetailStorEAR.ear` 并单击**打开**。
6. 单击**完成**。
7. 对 SVWholeSaleEAR 重复步骤 2 至 6。

将 EAR 文件部署至 iSeries WebSphere Application Server

既然已创建了 EAR 文件，就可以将它们部署至 WebSphere Application Server。

1. 打开“WebSphere 管理控制台”。
2. 右键单击**企业应用程序**并选择**安装企业应用程序**。
3. 选择**安装应用程序（*ear）**单选按钮。
4. 单击上面的**浏览**按钮（下面的那个不可用）。
5. 浏览至导出 EAR 文件的 IFS 目录。

6. 选择 **SVRetailStorEAR.ear**。
7. 在**应用程序名**字段中输入 “RETAILSTOR”。
8. 单击**下一步**并输入管理员角色的 iSeries 服务器用户标识。
9. 重复单击**下一步**，直到到达标题为**选择 Web 模块的虚拟主机**的页面。
10. 对于全部三个 Web 模块，单击**选择虚拟主机**并从下拉列表中选择您首选的虚拟服务器。（如果您不确定要选择哪个虚拟主机，则使用**缺省**或**缺省主机**。）
11. 单击**下一步**。
12. 对于所有三个 Web 模块，单击**选择服务器**并选择想要使用的服务器。（如果您不能确定要选择哪个服务器，可以使用**缺省服务器**。）
13. 单击**下一步**。
14. 单击**完成**并单击对话框中的**确定**。
15. 再次右键单击**企业应用程序**并选择**安装企业应用程序**。
16. 单击下面的**浏览**按钮（上面的那个不可用）。
17. 浏览至放置 EAR 文件的 IFS 目录。
18. 选择 **SVWholesaleEAR.ear**。
19. 在**应用程序名**字段中输入 “WHOLESALE”。
20. 重复地单击**下一步**，直到出现**完成**为止。
21. 单击**完成**并单击对话框中的**确定**。

    注: 可能要等几分钟才能出现确认消息。

在 WebSphere Application Server 中运行应用程序

要充当客户的角色并运行零售商店入口点，在 Web 浏览器中输入以下 URL:

`http://your iSeriesHostName:yourHTTPPortNumber/SV001585/shop.html`

要充当管理员的角色并运行批发入口点，在 Web 浏览器中输入以下 URL:

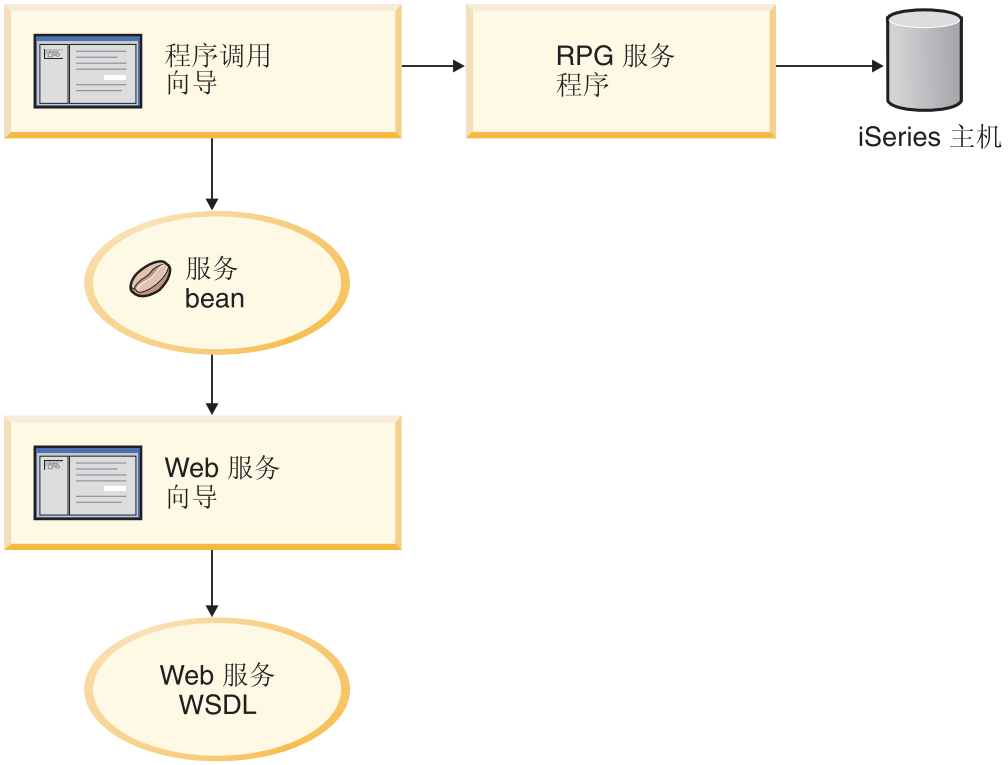
`http://your iSeriesHostName:yourHTTPPortNumber/SV000618/index.html`

如果您不知道 HTTP 端口号，请询问 WebSphere Application Server 管理员。

第 4 章 基本模块 1: 创建 Web 服务以返回产品价格（SV000514）

简介

在此模块中，您将根据 iSeries 服务器上存在的 RPG 程序创建 Web 服务以显示库存的产品价格。首先，创建一个 RPG 服务程序，该程序具有可以检索 iSeries 数据库中商品成本的过程（假设已给定商品号）。从 iSeries Java 开发工具使用“程序调用”向导来调用 RPG 程序，并创建服务 bean。然后使用“Web 服务”向导来创建 Web 服务，并使用生成的样本来验证 Web 服务。



在开始之前

只有在符合下列先决条件的情况下才能完成练习。第 7 页的第 3 章，『运行方案』对先决条件作了更详细的讨论。

- 对 iSeries 服务器具有 TCP/IP 访问权。
- 已应用最新的 iSeries 服务器 PTF。请参阅我们的支持页面以获取 PTF 和服务包信息：http://www.ibm.com/software/awdtools/wdt400/support/
- 已使用命令 `STRTCPSVR *ALL` 启动了 iSeries 服务器。
- 已使用命令 `STRTCPSVR *WEBFACING` 启动了 WebFacing 服务器。

- 您已完成了第 7 页的第 3 章，『运行方案』中的所有任务（可选的 WebSphere Application Server 任务除外，这些任务对于在工作台中测试应用程序不是必需的）。

创建新的 Web 项目

创建此 Web 服务的第一个步骤是创建新的 Web 项目来保存信息。

1. 从工作台 IDE 中切换至 Web 透视图或通过单击**窗口 > 打开透视图 > 其它 > Web > 确定**来打开 Web 透视图。
2. 单击**文件 > 新建 > 动态 Web 项目**。
3. 在**项目名称**字段中，输入 Project514。
4. 选择**配置高级选项**复选框并单击**下一步**。
5. 单击 **EAR 项目**字段旁的**新建**按钮。
6. 在**项目名称**字段中，输入 Project514EAR 并在两个对话框上单击**完成**。

现在，您可以在“项目导航器”视图中看到已将 Project514 和 Project514EAR 项目添加至您的工作空间中。

定义 iSeries 服务器信息

在创建了 Web 项目之后，需要定义项目使用哪个 iSeries 服务器来获取信息。

1. 右键单击 **Project514** 项目并选择**指定 iSeries Web Tools 运行时配置**。
2. 输入复原的 WHOLESALE 库所驻留的 iSeries 服务器的名称（例如，PROD400）。
3. 输入您用于此 iSeries 服务器的用户标识和密码。
4. 在**库**字段中输入 Wholesale 并单击**添加**。
5. 单击**完成**。

创建 RPG 服务程序

您想要应用程序在已给定商品号的情况下检索商品的价格。这是由包含称为 QryProdCost 的过程的 RPG 服务程序处理的。WHOLESALE 库包含一个名为 CWWSSRV 的 RPG 服务程序。此程序包含 QryProdCost 过程，该过程可以将商品号用作输入，打开 WHOLESALE 库中的 Inventory 文件，从库存数据库中检索价格，然后返回价格。为了接收此任务，界面提供了两个参数，一个用于商品号，另一个用于价格。如果找不到商品号或价格，则 RPG 程序返回消息至界面。

要创建此 Web 服务，首先使用“程序调用”向导创建一个 Java bean 以调用 QryProdCost RPG 过程。然后，使用 Web 服务来通过 Java bean 将该 RPG 过程作为 Web 服务启用。

要创建 Java bean:

1. 在 Web 透视图中，右键单击 **Project514** 并选择**新建 > 其它**。
2. 从窗口的左窗格中选择 **iSeries > Java**，然后在右窗格上选择**程序调用 Bean**。
3. 单击**下一步**以调用“程序调用”向导。
4. 在**添加程序**标题下的 **Java bean 名称**字段中，输入 Inventory。
5. 在**程序对象**字段中，输入 CWWSSRV（它是 RPG 服务程序的名称）。

6. 在**库**字段中，输入 WHOLESALE。
7. 从**程序类型**下拉列表中，选择 ***SRVPGM**。
8. 在**入口点**字段中，输入 QryProdCost。
9. 单击**确定**以添加程序定义。

创建参数和生成 Java bean

既然您已指定了程序，您就可以添加参数了。CWWSSRV RPG 程序包含下面两个参数:

- 商品号参数 – 程序使用此数字来查找数据库中的商品。
- 商品成本参数 – 程序使用此数字来查找数据库中商品的成本。

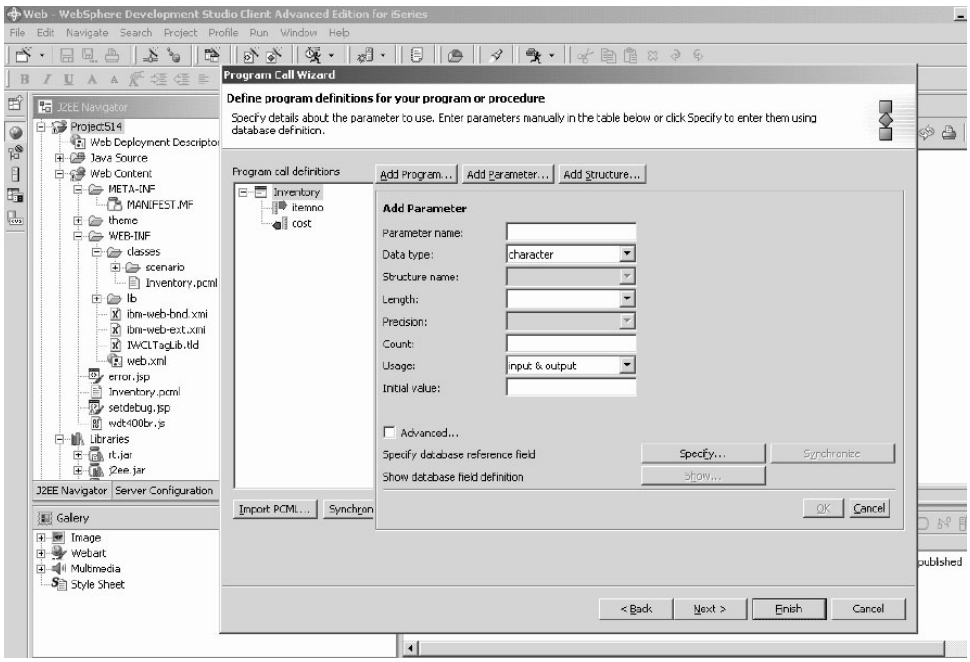
要添加商品号参数:

1. 在“程序调用”向导的左面板中，单击 **Inventory** 程序调用定义以选择它。此操作会重新植入该页面右边的各个字段。
2. 单击**添加参数**。
3. 在**参数名称**字段中，输入 itemno。
4. 从**数据类型**下拉列表中，选择**压缩十进制**。
5. 在**长度**字段中输入 5。
6. 在**精度**字段中输入 0。
7. 从**用法**下拉列表中，选择**输入**。
8. 单击**确定**以添加此参数。在左窗格中，注意到 **itemno** 出现在 **Inventory** 下面。现在，您已准备好添加第二个参数了。

要输入商品成本参数:

1. 在“程序调用”向导的左面板中，再次单击 **Inventory** 程序调用定义以选择它。此操作会重新植入该页面右边的各个字段。
2. 单击**添加参数**。
3. 在**参数名称**字段中输入 cost。
4. 从**数据类型**下拉列表中，选择**压缩十进制**。
5. 在**长度**字段中输入 7。
6. 在**精度**字段中输入 2。
7. 从**用法**下拉列表中，选择**输出**。
8. 单击**确定**以添加此参数。在左窗格中，注意到 **cost** 出现在 **Inventory** 下面。此时，向导看上去应与下图类似。注意，参数左边的图标显示参数是具有“输入”、“输

入和输出”还是“输出”类型:



要为这些参数定义包名，在“程序调用”向导中执行如下操作:

1. 单击**下一步**。
2. 在**包**字段中输入 scenario 作为包名。将其它字段保留为缺省值。
3. 清除 **Java 应用程序**复选框。

   注: 复查“这些文件将由向导生成”下面的文件列表，并请注意生成的 Java bean 的名称将是 InventoryServices.java。
4. 单击**完成**以生成文件。

从 Java bean 建立 Web 服务

在创建了调用 RPG 程序的 Java bean 之后，下一步就是将 bean 转换为 Web 服务，以便其它程序可以通过因特网访问这个 RPG 程序。

当创建 Web 服务时，“Web 服务”向导将生成分发给需要使用 Web 服务的用户的 WSDL 文件。要创建服务:

1. 在 Web 透视图的“导航器”视图中，展开 **Project514 > Java Source > scenario**。
2. 右键单击 **InventoryServices.java** 并选择**新建 > 其它**。
3. 在“新建”窗口中，单击 **Web 服务**，单击窗口右边的 **Web 服务**，然后单击**下一步**。
4. 选择**生成代理**和**测试生成的代理**复选框。
5. 单击**下一步**三次将会注意到生成了必需的文件。当您进至“Web 服务 Java Bean 身份”窗口，单击**完成**。如果见到任何错误，可能需要再次单击**完成**。如果无法完成向导，只需要单击**取消**，就一定会创建大多数必需的文件。

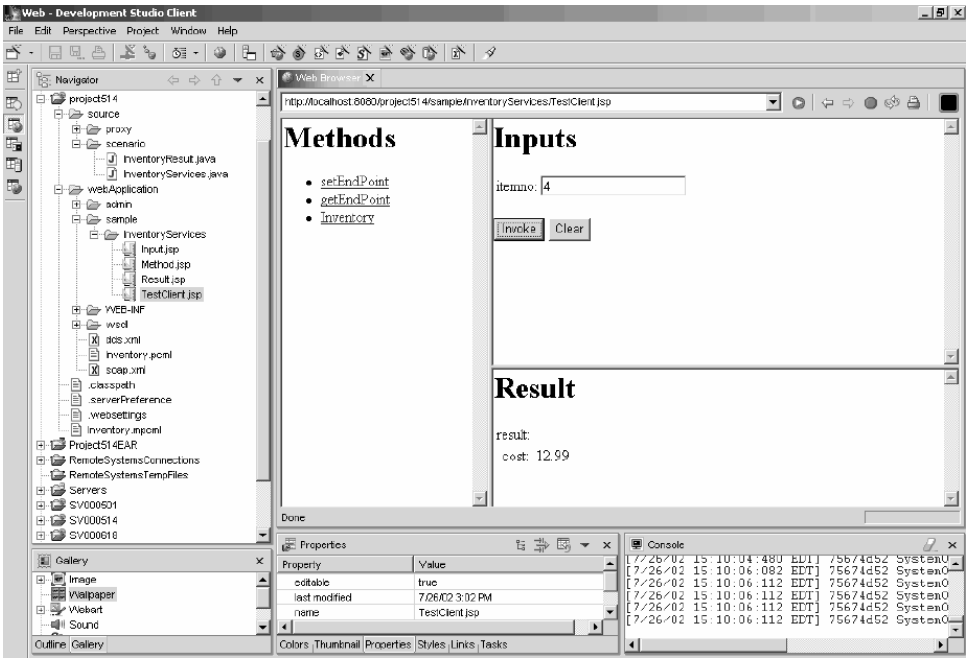
测试样本

当创建 Web 服务时，有一个指示信息是请求生成该样本。由于选择了该选项，所以“Web 服务”向导会创建具有测试页面的新项目，可以使用该测试页面来测试 Web 服务。在**导航器**视图中，可注意到此新项目名称是 **Project514Client**。另外还请注意，该工具在工作台的右边自动打开 TestClient.jsp。此文件位于 **Project514Client > Web Content > sample > InventoryServices** 下面。

要测试样本:

1. 在装入的页面中，向下滚动方法窗格，直到到达最下面为止。单击 **Inventory (java.math.BigDecimal)** 方法。
2. 在 **itemno** 字段中，输入 4 并单击**调用**。
3. 验证返回的输出是否为:

```
result:
cost: 12.99
```

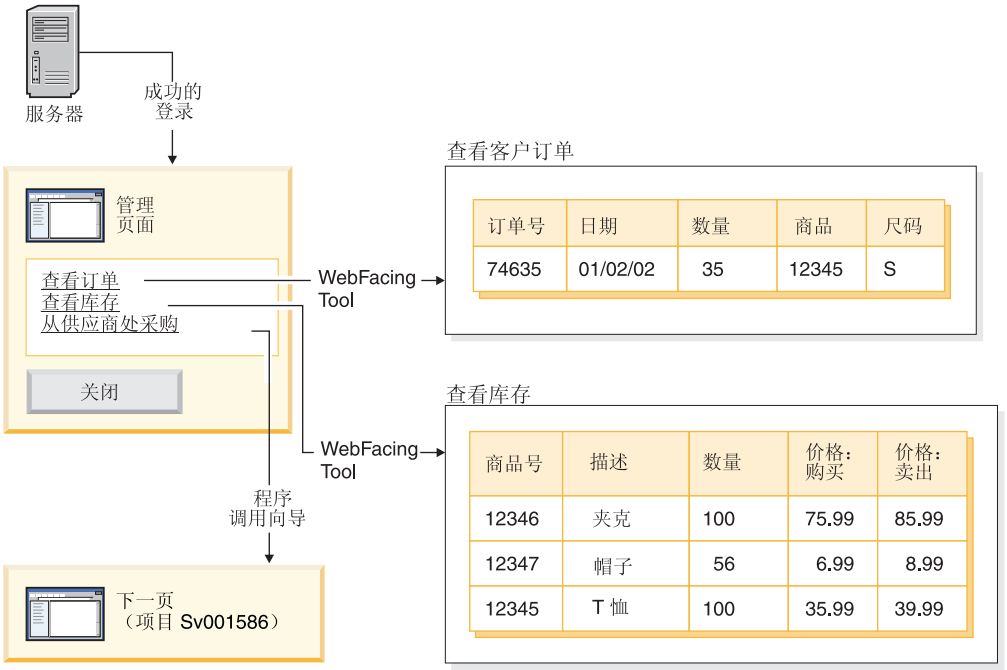


一旦确认了此结果，您就完成了该模块！您已经创建了 Web 服务，它将在应用程序界面中返回产品价格。

第 5 章 基本模块 2: 创建界面以查看库存和订单商品（SV000501）

简介

在此模块中，您将使用 IBM WebFacing Tool 创建一个界面，该界面显示管理员的操作选项。在成功登录到 iSeries 服务器之后，该界面会显示可用的库存，您就可以检查现有订单。之后，作为管理员您就可以从批发商处采购商品了。



在此项目中，您使用复原至 iSeries 服务器的两个程序和两个显示文件。程序名为 ViewInventory 和 ViewOrder。程序使用下面两个显示文件: ORDERDSP 和 QUERY。这两个文件包含为 WebFacing Tool 生成的 JSP 文件所定制的 Web 设置，并用于图像和超链接。图像 Web 设置允许您使用字段的内容来生成图像文件的名称并在 JSP 文件中显示图像。超链接 Web 设置允许您在单击 JSP 文件中的图像时调用另一个 Web 应用程序。可以使用 “CODE 编辑器” 或 “CODE 设计器” 来检查显示文件源代码以确定如何指定 Web 设置。

在开始之前

只有在符合下列先决条件的情况下才能完成练习。第 7 页的第 3 章，『运行方案』对先决条件作了更详细的讨论。

- 对 iSeries 服务器具有 TCP/IP 访问权。
- 已使用命令 `STRTCPSVR *ALL` 启动了 iSeries 服务器。
- 已使用命令 `STRTCPSVR *WEBFACING` 启动了 WebFacing 服务器。

- 您已完成了第 7 页的第 3 章，『运行方案』中的所有任务（可选的 WebSphere Application Server 任务除外，这些任务对于在工作台中测试应用程序不是必需的）。

创建 WebFacing 项目

首先需要执行的任务是创建 WebFacing 项目并指定相关的 CL 命令。要创建 WebFacing 项目:

1. 在工作台中，通过单击工作空间左边的其中一个透视图图标，或者单击**窗口 > 打开透视图 > 其它 > WebFacing**，然后单击**确定**来切换到 WebFacing 透视图。
2. 通过单击**文件 > 新建 > WebFacing 项目**来创建新的 WebFacing 项目。
3. 将项目命名为 `Project501`。
4. 单击**企业应用程序项目**旁的**现有**单选按钮。
5. 在**现有项目名称**字段中输入 `SVStoreEAR` 值；注意该值是区分大小写的。单击**下一步**。
6. 对于 **J2EE 级别**下拉列表，选择 **1.3**。单击**下一步**。
7. 在**连接**字段中，应自动填写 iSeries 服务器的名称。或者，单击“新建”并在对话框中输入 iSeries 服务器的名称，然后单击**完成**。
8. 单击**刷新 DDS 列表**并在弹出对话框中输入密码和用户标识以刷新列表。
9. 在库的生成列表中，展开 **RETAILSTOR**。
10. 单击 **QDDSSRC**，然后单击向右箭头（>>）以移动这些文件。
11. 单击**下一步**两次，直到到达**指定 CL 命令**页面。

现在，您需要为管理员添加命令和命令标注:

12. 在 **CL 命令**字段中，输入调用 `call viewinvent`。
13. 在**命令标注**字段中，删除现有值并输入 `View inventory`。
14. 选择**使用指定值登录**单选按钮。
15. 单击**添加**，并注意窗口底部列表中的添加内容。

现在，需要为客户添加第二个命令和命令标注:

16. 在 **CL 命令**字段中，删除现有值并输入 `call vieworder`。
17. 在**命令标注**字段中，删除现有值并输入 `View orders`。
18. 选择**提示登录**单选按钮。
19. 单击**添加**，并注意窗口底部列表中的添加内容。单击**下一步**。
20. 在**选择 Web 样式**窗口中，逐步滚动样式以查看哪种样式可用。由于稍后我们需要根据此样式检索信息，所以对于此模块选择 **avenue**。单击**下一步**。
21. 验证是否选择了**不。我现在只想创建项目**复选框并单击**完成**。
22. 如果看到“修复服务器配置”消息，则单击**确定**。此消息只是确认您想要把项目添加至 SVStoreEAR 文件。

由于您要将 Project501 构造为整个 SV000501 项目的镜像，所以现在需要将某些图像文件从 SV000501 复制到 Project501，以便 Project501 正确显示。（但如果需要的话，可以对界面添加您自己的图像。）要复制图像:

1. 切换到 Web 透视图。
2. 在“导航器”视图中，展开 **SV000501 > Web Content > images**。
3. （可选）展开 **generated** 以记下您正导入到项目中的所有图像。

4. 右键单击 **generated** 并选择**复制**。
5. 在“导航器”视图中向上滚动，直到可以再次看到 **Project501** 为止。
6. 展开 **Project501 > Web Content**。
7. 右键单击 **images** 文件夹并从弹出菜单中选择**粘贴**。
8. 现在，Project501 存在 **generated** 文件夹。展开 **generated** 以记下您添加的图像。

转换 DDS 源

既然您已创建了项目，就可以为 Web 页面将 DDS 显示文件转换为 JSP 文件。当转换 DDS 显示文件时，WebFacing Tool 会生成 JSP 和 XML 文件，这些文件替换 DDS 代码并使 Web 访问成为可能。生成的文件保存记录格式的数据，或者控制其外观，显示屏幕的 Web 版本，提示输入数据并处理输入错误。向导还生成应用程序主页来启动程序的支持 Web 的版本。

但是，首先需要改变源代码以引用正确的 iSeries 服务器。更具体地说，需要在其中一个 DDS 显示文件中执行修改，以便应用程序的链接将起作用。需要将缺省名称 SV000501 更改为项目名称 Project501:

1. 切换回 WebFacing 透视图并切换至“WebFacing 项目”视图（如果在缺省情况下它还未打开的话）。
2. 展开 **Project501 > DDS**。
3. 双击第二项 *<iSeriesserver>***RETAILSTOR/QDDSSRC(QUERY)**
4. 向下滚动至大约第 16 行，在此位置可以看到该行:

   ```
   A*%%WB 12 FLD 1 next ('/SV000501/DetailPage.do?PRODNO=&{PRODNO}')
   ```

5. 删除 `SV000501` 并输入 `Project501`。
6. 保存并关闭文件。

注: 如果在某个时候您想要再次运行 SV000501 项目，则需要将 Project501 值更改回 SV000501，原因是此文件存在于 iSeries 服务器上，并且同时由两个项目访问。

要转换 DDS 源:

1. 在 WebFacing 透视图中，单击 **WebFacing 项目**选项卡以切换至“WebFacing 项目”视图（如果在缺省情况下它还未打开的话）。
2. 展开 **Project501 > DDS**。
3. 选择 *iSeriesserver* **> RETAILSTOR/QDDSSRC(ORDERDSP)** 和 *iSeriesserver* **> RETAILSTOR/QDDSSRC(QUERY)**（通过单击第一个然后按住 Shift 键并单击第二个）。
4. 右键单击并选择**转换**以开始转换。如果在某些时候您与 iSeries 服务器断开连接，可能需要您输入用户标识和密码。

在工作台中配置 UTF-8 支持 - 仅适用于 WAS V4.0 用户

IBM WebFacing Tool 应用程序支持在屏幕上显示多种语言。由于各种语言使用不同的字符集，所以用 UTF-8 编码浏览器与 WebSphere Application Server 之间的数据流。为了使 IBM WebFacing Tool 正确工作，需要在工作台中的应用程序属性文件中配置 UTF-8 支持。

注: 本节只适用于 WAS 4.0 用户。WAS V5.0 自动执行此任务。

要配置 UTF-8 支持:

1. 切换到 Web 透视图（可通过单击屏幕左边聚集的图标来在各透视图之间切换）。
2. 单击**项目导航器**选项卡以便可查看项目结构。
3. 展开 **Project501** 并双击 **Web 部署描述符**。
4. 单击**环境**选项卡。
5. 单击**添加**按钮。
6.
7. 通过单击左上方任务栏中的“服务器”透视图的图标来切换到“服务器”透视图；或者从菜单栏中单击**透视图 > 打开 > 其它 > 服务器**，然后单击**确定**来切换到“服务器”透视图。
8. 在“导航器”视图中，展开 **Servers** 文件夹。
9. 双击 **defaultInstance.wsi** 以在缺省编辑器中打开它。
10. 单击**环境**选项卡并单击**添加**按钮。
11. 可编辑的值“(New Variable)”会出现。删除此缺省字符串并输入: `client.encoding.override`。
12. 在**值**字段中输入 `UTF-8`。
13. 单击保存图标或单击**文件 > 保存 Web 部署描述符**。

为 WebSphere Application Server 配置 UTF-8 支持

（可选）如果想要将 iSeries 应用程序部署至 WebSphere Application Server，还需要在 WebSphere Application Server 和工作台中配置 UTF-8 支持。

要在 WebSphere Application Server 4.0 Advanced Edition 中配置 UTF-8 支持

1. 启动“WebSphere 管理控制台”。
2. 展开**节点**图标并展开**节点名称 > 应用程序服务器 > 缺省服务器**。
3. 选择 **JVM 设置**选项卡并单击**高级 JVM 设置**按钮以打开**高级 JVM 设置**对话框。
4. 在**命令行自变量**字段中，输入:

   ```
   -Dclient.encoding.override=UTF-8
   ```

5. 单击**确定**并单击 **JVM 设置**选项卡下面的**应用**。
6. 为了使此更改对 WebSphere 应用程序生效，停止缺省服务器，然后重新启动它。要停止服务器，右键单击**缺省服务器**并选择**停止**。在此过程完成之后，右键单击**缺省服务器**并选择**启动**。

在 WebSphere Application Server 4.0 Advanced Single Server Edition 中配置 UTF-8 支持

1. 启动“WebSphere 管理控制台”。
2. 在基于浏览器的“管理控制台”中，展开**节点**图标并展开**节点名称 > 应用程序服务器 > 缺省服务器 > 过程定义 > JVM 设置**。
3. 滚动至“JVM 设置”页面的“高级设置”部分，并单击“系统属性”链接。将显示“系统属性”页面。
4. 单击**新建**以添加新的“系统属性”。

5. 在**名称**字段中，输入 client.encoding.override。
6. 在**值**字段中输入 UTF-8。
7. 单击**确定**。如果您在 **JVM 设置**页面顶部接收到带有链接的**需要保存配置**的消息，则单击该链接以访问**保存配置**页面。选择**保存**，然后单击**确定**。
8. 为了使此更改对 WebSphere 应用程序生效，停止应用程序服务器，然后重新启动它。停止和启动应用程序服务器的方式根据安装 WebSphere Application Server 的平台的不同而有所不同。参阅平台的 WebSphere Application Server 文档以了解有关停止和启动应用程序服务器的信息。

创建样式表

如果您想要将附加页面与级联样式表（CSS）集成在一起，需要定制 WebFacing 项目的样式或者级联样式表以使它们看上去是相似的。在完成定制样式表之后，可以使用“Web 交互作用”向导来创建详细的 Web 页面，并使用样式表来显示有关商店商品的信息，例如，价格和颜色。考虑到此模块，将合并 SV000501 项目中的 DetailPageResults.jsp 样式表。但为了将来引用，可以手工定制 CSS 文件（在下一节中概述）。

（可选）手工定制级联样式表

正如前面所提到的那样，您未对 Project501 合并 SV000501 项目中的 DetailPageResults.jsp 样式表，而是定制您自己的样式表:

1. 切换至“项目导航器”视图并展开 **Project501 > Web Content > styles > apparea**。
2. 双击 **apparea.css** 以在“CSS 设计器”中打开它。此样式表控制应用程序区域的外观。
3. 修改并保存该文件。
4. 返回到“项目导航器”视图中，浏览至 **Project501 > Web Content > styles > chrome**。
5. 双击 **avenue.css** 以在“CSS 设计器”中打开它。此样式表控制整个页面的外观。
6. 修改并保存该文件。

使用 Web 交互作用来扩展和增强 WebFacing 项目

在本节中，将增强 WebFacing 项目。当显示商品列表时，您想要单击商品图像以查找该商品的详细信息。为此，使用“Web 交互作用”向导来调用 RPG 程序以检索商品的详细信息并在另一个 Web 页面上显示这些信息。将执行下列步骤:

- 定义服务器信息
- 复制正确的样式表
- 创建交互作用
- 将程序和参数添加至交互作用
- 更改参数的用法

首先，需要定义服务器信息:

1. 切换到 Web 透视图。

2. 在“导航器”视图中，右键单击 **Project501** 并选择**指定 iSeries Web Tools 运行时配置**。
3. 输入 iSeries 服务器名、用户标识和密码。
4. 在**库**字段中输入 Retailstor，然后单击**添加**。
5. 单击**完成**。必要时再次单击**完成**。

在创建 Web 交互作用之前，需要复制正确的样式和样式表，以便 Project501 显示正确的 JSP 文件格式。（如果您完成了练习第 31 页的『（可选）手工定制级联样式表』，则不需要执行此任务）。首先复制样式文件夹，然后复制 DetailPageResults.jsp:

1. 在“导航器”视图中，展开 **SV000501 > Web Content**。
2. 右键单击 **Styles** 并选择**复制**。
3. 在“导航器”视图中向上滚动，直到再次看到 **Project501** 为止。
4. 展开项目，右键单击 **Web Content**，并从弹出菜单中选择**粘贴**。单击**是**以覆盖现有样式。
5. 向下滚动，回到 **SV000501 > Web Content**。
6. 右键单击 **DetailPageResults.jsp** 并选择**复制**。
7. 向上滚动，直到再次看到 **Project501** 为止。
8. 展开项目，右键单击 **Web Content**，并从弹出菜单中选择**粘贴**。

注意，新元素已添加到 Project501 的 Web Content 文件夹的下面。

现在，您可以创建 Web 交互作用:

1. 单击**文件 > 新建 > 其它**。
2. 在**新建**对话框中，在左边单击 **Web**，然后在右边单击 **Web 交互作用**。单击**下一步**。
3. 在 **Web 交互作用名称**字段中，输入 DetailPage 并单击**下一步**。
4. 选择**生成输入 JSP** 单选按钮。
5. 选择**使用输出页面**单选按钮（如果尚未选择它）并单击**添加**。
6. 从**输出 JSP** 对话框中，展开 **Web Content**，选择 **DetailPageResults.jsp**（预先格式化的输出页面），并单击**确定**。
7. 单击**下一步**。

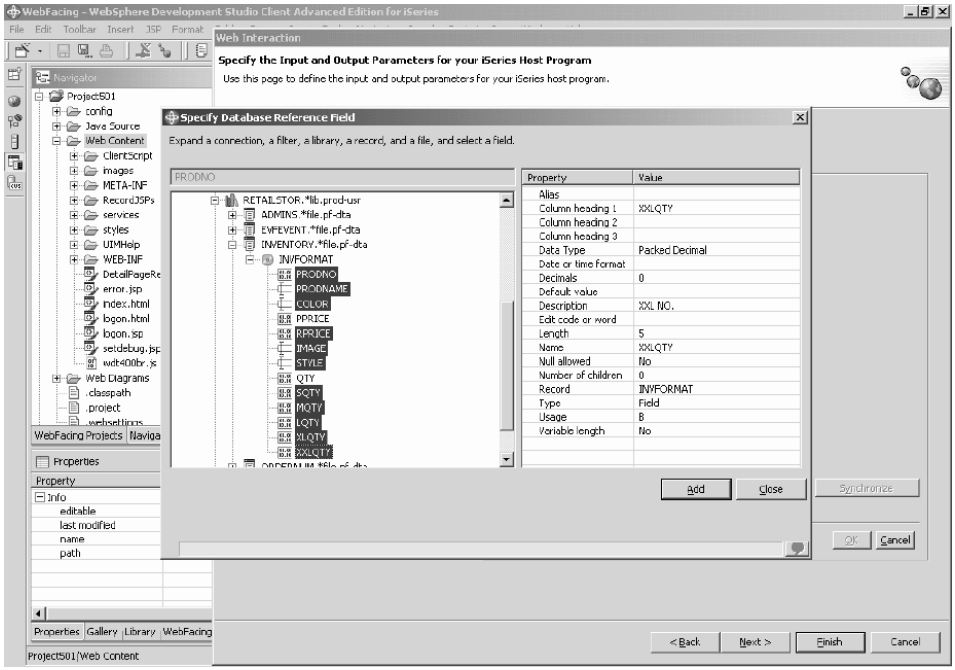
现在，可以将程序和参数添加至交互作用。需要将 11 个参数添加至同一个程序。不要手工添加每个参数及其单个值，可以用较快一点的方法添加它们:

1. 选择**使用 iSeries ILE 程序**（如果尚未选择它）。
2. 单击**添加程序**。
3. 在**程序别名**字段中，输入 DetailPage。
4. 对于**程序对象**字段，单击**浏览**。
   a. 展开 *iSeriesserver* **> *LIBL > RETAILSTOR**。
   b. 单击 **DETAILPAGE.*pgm.rpgle**（RETAILSTOR 下面的第一个文件）并单击**确定**。
5. 返回到“Web 交互作用”向导，单击**确定**（在右下方）。

注意，DetailPage 已添加在向导左边“程序调用定义”的下面。

6. 在向导左窗格的“程序调用定义”部分中，单击 **DetailPage** 以选择它。

7. 单击**添加参数**。
8. 在向导底部**指定数据库引用**字段旁边，单击**指定**。
9. 展开 *iSeriesserver* **> *LIBL > RETAILSTOR > INVENTORY.*file.pf-dta > INVFORMAT** 以显示 13 个参数的列表。需要通过对下列 11 个参数中的每个单击一次并单击**添加**，或者按住 CTRL 键并单击每个参数然后单击**添加**来添加它们: PRODNO、PRODNAME、COLOR、RPRICE、IMAGE、STYLE、SQTY、MQTY、LQTY、XLQTY 和 XXLQTY。基本来说，选择除 PPRICE 和 QTY 之外的所有字段。

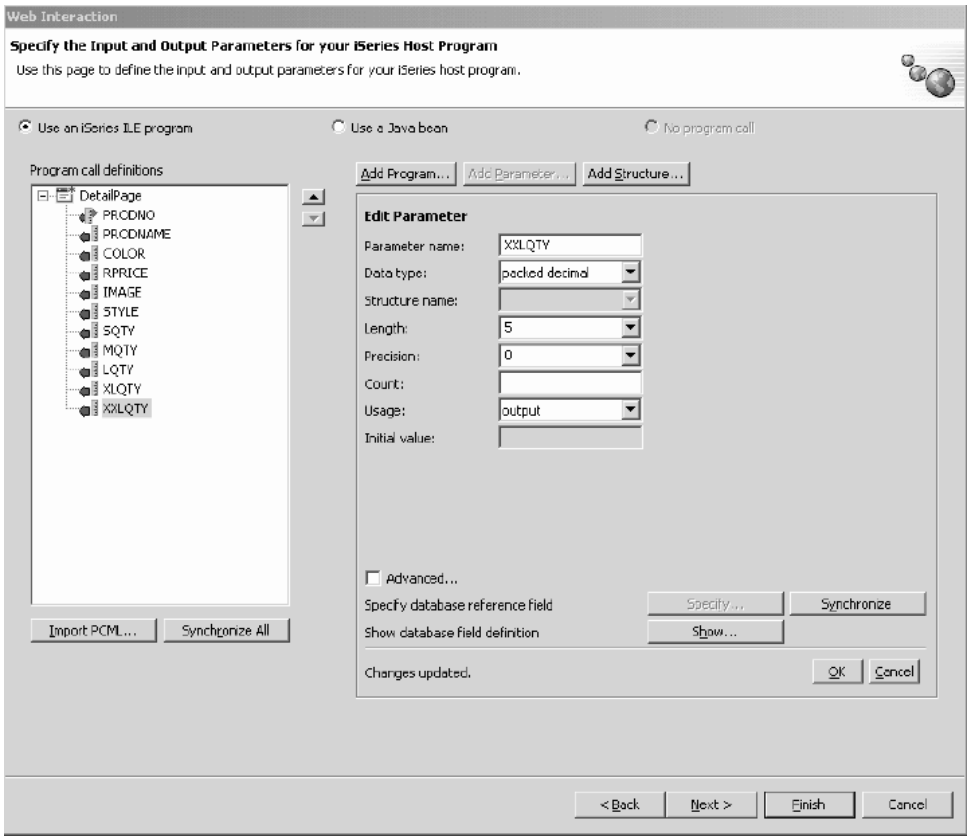


10. 单击**关闭**。

现在，需要将 11 个参数中的 10 个的用法更改为“输出”（除 PRODNO 之外的所有参数）。

1. 仍在“Web 交互作用”向导中，单击 **PRODNAME** 以选择它。
2. 在**用法**组合框中，将选择的值切换为**输出**并单击**确定**。对除第一个参数（PRODNO）之外的所有参数重复此步骤（以及前一步骤）。一旦调整了所有参数之后，工作空间应与下图类似。注意，参数左边的图标显示它是具有“输入”、

“输入和输出”还是“输出”类型。



3. 单击**下一步**以查看**输入表单**预览。
4. 单击**完成**以创建 Web 交互作用。
5. 选择**全部都是**或**确定**，并且如果接收到任何消息，则再次选择**完成**。

将项目链接至 Web 交互作用

既然您已创建了使用输入和输出参数的 Web 交互作用 JSP 文件，就需要定制该 JSP 文件以便它也可以使用 WebFacing 组件。需要输入代码才能创建来自 WebFacing 应用程序的链接以调用此 Web 交互作用。为此，需要在 `webface.js` 文件中添加 JavaScript 函数，以便您可以在新窗口中使用 PRODNO 参数调用 DetailPageServlet servlet。

要创建链接:

1. 在“导航器”视图中，展开 **Project501 > Web Content > ClientScript**。
2. 双击 **webface.js** 以在编辑器中打开它。
3. 滚动至文件的底部并输入下列行:

```
var mywindow
 function next(app)
{
mywindow = window.open(app,"Details","RESIZABLE=YES, HEIGHT=700, WIDTH=800");
}
```

4. 单击保存图标或单击**文件 > 保存 webface.js**。

在将来，如果想要创建类似于 SV000501 的应用程序，则还需要在 DDS 源中更改 Web 设置以启用添加的图像，并且还启用 JavaScript 函数的关闭窗口的链接。

改变此应用程序中包括的 RPG 代码以显示更改，但是，在将来的应用程序中将需要手工进行更改。而且，在更改了 Web 设置之后，将需要重新转换 DDS 源。

可以检查代码以重复该结果。

要查看 DDS 源:

1. 切换到 WebFacing 透视图。
2. 在 WebFacing 项目视图中，展开 **Project501 > DDS**。
3. 右键单击 **<iSeries 服务器 > RETAILSTOR/QDDSSRC(QUERY)** 并选择**打开方式 > CODE 设计器**。
4. 在“CODE 设计器”打开之后，展开 **SCREEN1 > ITEMSUB**。
5. 单击 **IMAGESRC** 以选择它。
6. 单击**源**选项卡。
7. 单击窗口右下部分中的 **Web 设置**选项卡。

   注: 检查 Web 设置属性，例如，用像素表示的宽度以及文件名。将来，您必须对 DDS 源作出同样的更改，然后重新转换源。
8. 注意源中的下列行:

```
A     PRODNO  R   0 5 6
A     PRODNAME  R   0 5 16
A     IMAGESRC   19A 0 5 33
A*%%WB 13 FLD 100|100|&{IMAGESRC}
A*%%WB 12 FLD 1 javascript:next
   ('/Project501/DetailPageServlet?PRODNO=&{PRODNO}')
```

   特别请注意，Project501 是在最后一行中指定的。如先前所述，如果您想要再次运行 SV000501 应用程序，将需要把 Project501 值更改回 SV000501。
9. 通过单击保存图标或从菜单栏单击**文件 > 保存**来保存文件。
10. 关闭文件并关闭“CODE 设计器”。

有关如何使用 DDS 源的更多信息，切换到工作台的“帮助”透视图并查看 IBM WebFacing Tool 文档。

发布文件并重新启动服务器

在本节中，您发布文件并重新启动以便 Project501 应用程序可以应用所有更改。

但是在重新启动服务器之前，需要验证是否已将 Project501.war 添加至 SVStoreEAR 文件。

要验证配置:

1. 在“导航器”视图中，展开 **Scenario** 文件夹。
2. 双击 **Scenario server.wsi**。

您可能会看到一个对话框，该对话框询问您是否想要编辑器自动修正项目条目。单击**是**。如果没有看到对话框，则不需要执行任何操作。不管是哪种情况，打开文件时都会自动修改它。

3. 保存并关闭文件。

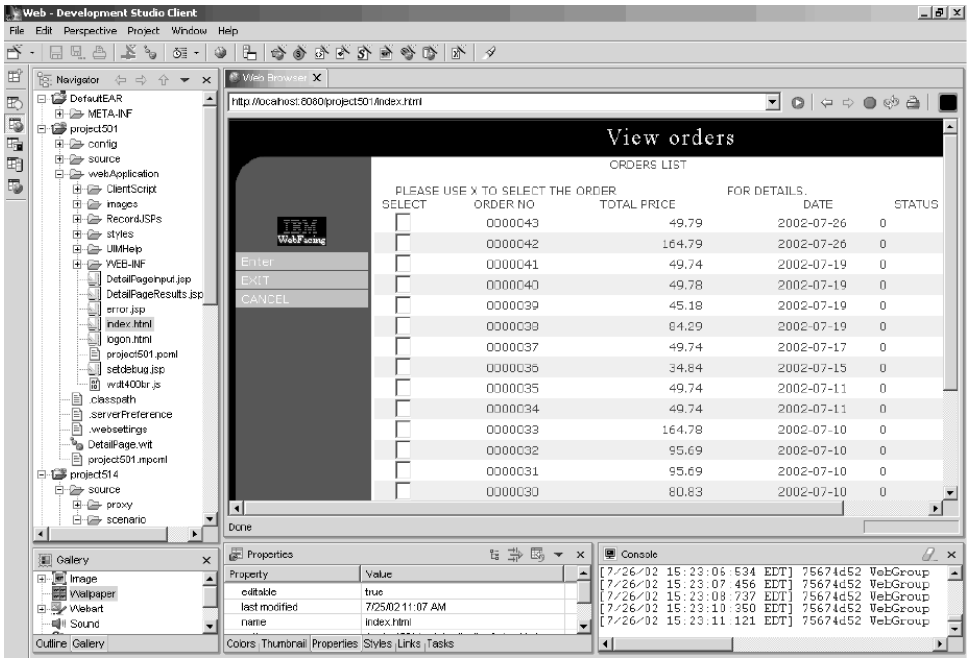
要重新启动服务器:

1. 切换到“服务器”透视图。
2. 在“服务器配置”视图左下方，展开 **Servers** 并双击 **Scenario server**。
3. 单击屏幕右下方的**服务器**选项卡以在“服务器”视图中查看服务器的状态。
4. 右键单击 **Scenario server** 并选择**发布**。发布完成之后单击**确定**。
5. 在“服务器”视图中右键单击服务器并选择**启动**或**重新启动**（以已启用的那一个为准）。如果提示您未保存文件，则取消对话框，保存所有打开的文件，然后再继续进行。
6. 检查“控制台”日志信息（自动打开）。当您在日志底部看到以下行时，就将知道服务器已启动：服务器 *server_name* 打开以执行电子商务。

测试界面

现在，您已完成了创建供查看订单和查看库存的界面所必需的步骤。要测试界面：

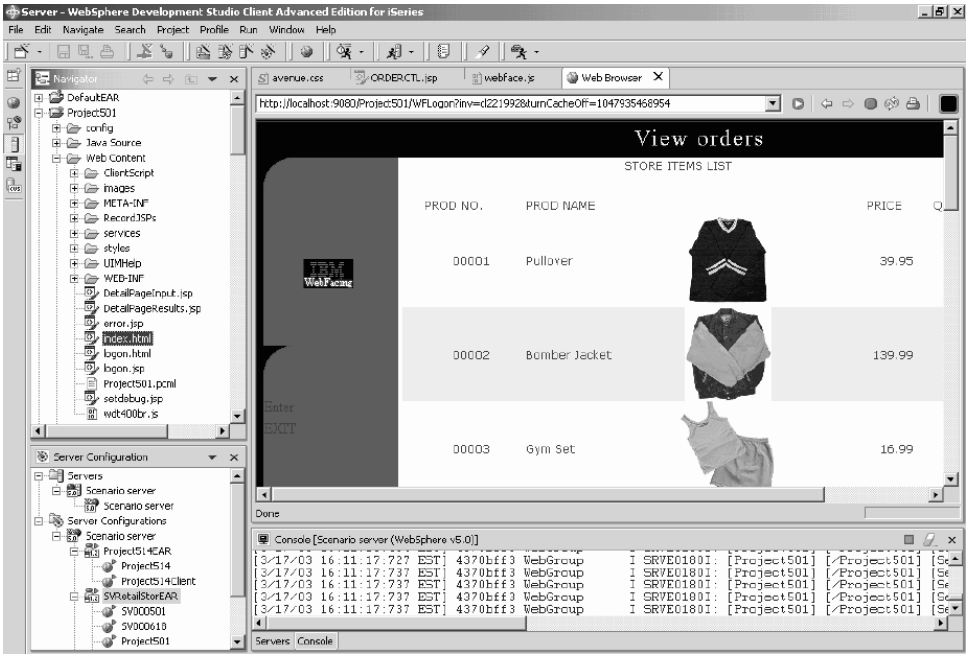
1. 切换到 Web 透视图。
2. 展开 **Project501 > Web Content**。
3. 右键单击 **index.html** 并选择**在服务器上运行**。
4. **服务器选择**对话框打开。验证已缺省选择了 **Scenario server** 并单击**完成**。
5. 单击**查看订单 – 在主浏览器窗口中启动**以调用管理员应用程序。在用 iSeries 用户标识和密码（对于开发此项目时使用的 iSeries 服务器）登录之后，将向您显示以下页面：



尝试在任何产品号旁边输入 X 以查看有关该产品的详细信息。

6. 单击向后箭头以进入 index.html 页面，并单击**查看库存 – 在主浏览器中启动**窗口以调用客户应用程序，将向您显示以下页面（在使用用户标识和密码登录到 iSeries

服务器之后）：



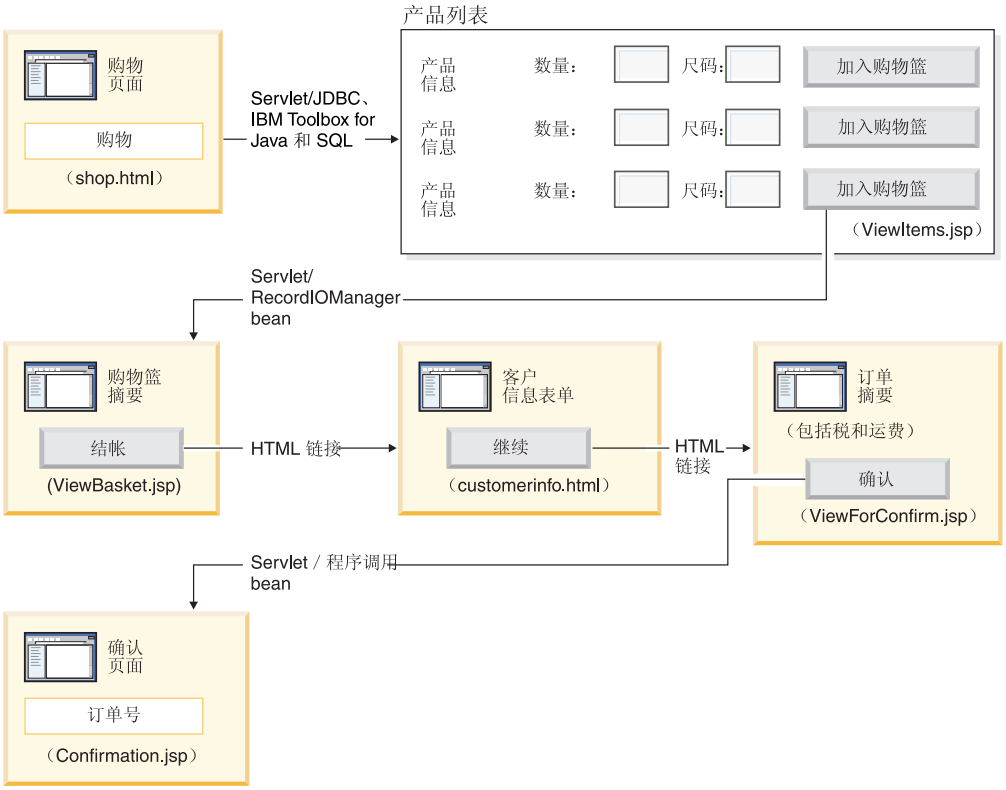
尝试单击图像以显示有关特定商品的详细信息。

现在，您已完成了该模块！您已创建了用于查看库存和订购商品的用户界面。

第 6 章 高级模块 1: 在 iSeries 服务器上创建生成客户订单的 HTML、servlet 和 JSP 文件（SV001585）

简介

此项目用于具有较好的 Java 编程知识并具有一些 iSeries 数据管理和 RPG 知识的用户。该项目显示如何使用 IBM Toolbox for iSeries 数据访问类、RecordIOManager bean 和程序调用 bean 以在 iSeries 服务器上创建生成客户订单的 HTML 代码、servlet 和 JSP。作为用户，您先从购物页面查看提供的产品并将商品加入您的购物篮。一旦您对所选择的所有商品（在您的“购物篮摘要”中）都很满意，单击结帐按钮，将引导您进入客户信息表单。一旦完成了该表单，该项目就返回带有确认按钮的订单摘要，它将带您进入显示您的订单和订单号的确认页面。



高级步骤总结

因为这是一个高级模块，所以指示信息不会指导您逐步执行创建项目的每个步骤，但是给出用来创建这种项目的 iSeries 特定开发步骤的概要步骤。以下是高级步骤:

1. 编写 HTML 购物 Web 页面。
2. 编写将可供客户采购的可用商品植入 Java bean（使用 JDBC 和 SQL）的 servlet。
3. 编写 JSP 文件来查看出售的商品并允许客户输入需要的商品数量和尺码并通过单击**加入购物篮**按钮来选择商品。

4. 使用 RecordIOManager bean 编写通过单击**加入购物篮**按钮调用的 servlet，该 servlet 通过减去商品所需的数量和尺码并将此选择添加至称为 “Basket” 的 Java bean 来更新 iSeries INVENTORY 数据库。如果该操作成功，则该 servlet 将响应重定向至 `ViewBasket.jsp`。如果该操作失败，则该 servlet 会显示错误页面。
5. 编写供客户输入个人信息的 HTML 表单。
6. 编写显示购物篮物品、税款以及运费和处理费用的 JSP 文件采购确认页面。该页面还必须包含客户的确认按钮。
7. 编写在客户单击确认按钮时要调用的 servlet，该 servlet 使用通过 “iSeries 程序调用” 向导创建的 Java bean。Java bean 的其中一个方法是调用 RPG 程序来在 iSeries 服务器上的 ORDERS 数据库中创建与客户的购物篮物品相对应的新订单。然后，该 servlet 返回订单号，将 Java bean 放置在 Web 应用程序的会话中，并装入包含该订单号的订单确认 JSP 文件。

在开始之前

只有在符合下列先决条件的情况下才能完成练习。第 7 页的第 3 章，『运行方案』对先决条件作了更详细的讨论。

- 对 iSeries 服务器具有 TCP/IP 访问权。
- 已使用命令 `STRTCPSVR *ALL` 启动了 iSeries 服务器。
- 已使用命令 `STRTCPSVR *WEBFACING` 启动了 WebFacing 服务器。
- 已将 WHOLESALE 和 RETAILSTOR 库复原至 iSeries 服务器。
- WHOLESALE、RETAILSTOR 和 QGPL 库都在库列表中。
- 您已完成了第 7 页的第 3 章，『运行方案』中的所有任务（可选的 WebSphere Application Server 任务除外，这些任务对于在工作台中测试应用程序不是必需的）。

创建 Web 页面、servlet 和 JSP 文件

要构造项目 SV001585 的组件:

1. 创建 Web 项目。
2. 使用 “页面设计器” 编写包含调用 GetItems servlet 的链接的 `shop.html` 页面。
3. 将 iSeries Toolbox for Java 类的 jt400.jar 文件导入 Web 项目 *lib* 文件夹。可以在 *x*:\wdsc\wssd\plugins\com.ibm.etools.iseries.toolbox\runtime（其中 *x* 是 Development Studio Client 的安装目录）中找到此 jar 文件。

   注: 有关 servlet 和 JSP，参见 SV001585 项目中的 `GetItems.java` 和 `ViewItems.jsp` 以查看主 iSeries Toolbox for Java JDBC 和 SQL 相关部件。在 Web 透视图的 “导航器” 视图中，可以通过展开 **SV001585 > source** 来查找 `GetItems.java`，可以通过展开 **SV001585 > webApplication** 来查找 `ViewItems.jsp`。

   .
4. 编写 GetItems servlet，它使用 iSeries Toolbox for Java JDBC 和 SQL 来从实现以下功能的 iSeries INVENTORY 数据库中检索服装商品:
   a. 将包含 SQL 查询结果的 ResultSet bean 放置到会话中
   b. 将请求重定向至 `ViewItems.jsp`

   `GetItems.jsp` 的代码样本如下:

```
public void init() {
   .
   .
   .
   // Load the IBM Toolbox for Java JDBC driver.
   DriverManager.registerDriver(new com.ibm.as400.access.AS400JDBCDriver());
   // Note that we have retrieved the as400 name, userid, and password from
   // web.xml file using and xml parser.
   as400conn =
    DriverManager.getConnection(
     "jdbc:as400://" + as400 + ";naming=sql;errors=full",
     userid,
     password);

   dmd = as400conn.getMetaData();
   .
   .
   .
}

public void service(HttpServletRequest request, HttpServletResponse response){
   .
   .
   .
   Statement select =
    as400conn.createStatement(
     ResultSet.TYPE_SCROLL_INSENSITIVE,
     ResultSet.CONCUR_READ_ONLY);
   ResultSet rs =
    select.executeQuery(
     "SELECT PRODNO, PRODNAME, RPRICE, IMAGE FROM "
      + retailLibrary
      + dmd.getCatalogSeparator()
      + inventoryFile);

 HttpSession session = request.getSession(true);
   session.setAttribute("resultset", rs);

   response.sendRedirect("/ViewItems.jsp");

   .
   .
   .
}
```

5. 编写 ViewItems JSP 文件，它根据前一步骤中所获得的 ResultSet bean 检索服装商品，以便用表格形式显示服装商品。JSP 文件还应为每种商品包括一个表单，可使用该表单来选择尺码和数量，然后将商品加入购物篮。可以使用 iSeries Web 开发工具中的“页面设计器”来编写 JSP。更具体地说，可以在“设计”视图中展开该页面并在“源”视图中添加适当的代码。ViewItems.jsp 的代码样本如下:

```
<!--Getting the ResultSet Object from the session--><%
   int columnCount = 0;
   ResultSet rs = (ResultSet)session.getAttribute("resultset");
   if(rs !=null)
%>
<%
  {
  rs.beforeFirst();
            ResultSetMetaData rsmd = rs.getMetaData ();
            columnCount = rsmd.getColumnCount ();

%>
<TABLE border="1">
<TBODY>
<TR>
  <TD>Product ID</TD>
```

```
  <TD>Name</TD>
  <TD width="551">Price</TD>
  <TD colspan="2"></TD>
</TR>
<%while (rs.next ()){
<TR>
<!--Creating a form for this row (or this item)-->
<FORM name="myform" action="/SV001585/AddtoBasket"
onsubmit="return errorChecking(this);">
<!--Getting each column data from this row of ResultSet object-->
<!--Process data is a user defined method to modify the data for display if needed-->
          <%
                for (int i = 1; i <= columnCount; ++i){
                    String value = rs.getString(i);
                    if (rs.wasNull ())
                        value = "<null>";
                    else{
                     if(i==1)
                      prodID=value;
                     value = processData(i,value);

                   }
     %>
    <TD><%=value%></TD>
      <%
                }
   %>

<!--Creating quantity input field and size drop down menue-->
<!--Note that we are using product id as the name of the field-->
         <TD width="290">Quantity
   <INPUT size="5" type="text" name='<%=prodID+"Q"%>' ><BR>
     Size <SELECT name='<%=prodID+"S"%>'>
       <OPTION value="s" selected>Small</OPTION>
       <OPTION value="m" selected>Medium</OPTION>
       <OPTION value="l" selected>Large</OPTION>
       <OPTION value="XL" selected>Extra Large</OPTION>
       <OPTION value="XXL" selected>Extra Extra Large</OPTION>
       </SELECT>
</TD>
<TD><INPUT type="image"   name="submit"  src="images/Add_to_basket.gif"></TD>
</FORM>
</TR>
<%
}
%>
</TBODY>
</TABLE>
<%
}
```

6. 将 iSeries Java 开发工具 iseriesut.jar 文件导入 Web 项目的 *lib* 文件夹中。可以在 *x*:\wdsc\wssd\plugins\com.ibm.etools.iseries.toolbox\runtime（其中 *x* 是产品的安装目录）中找到此 JAR 文件。参见 SV001585 项目中的 AddtoBasket.java 和 ViewBasket.jsp 来查看实现。在 Web 透视图的“导航器”视图中，可以通过展开 **SV001585 > source** 来查找 AddtoBasket.java，可以通过展开 **SV001585 > webApplication** 来查找 ViewBasket.jsp。
7. 从 iSeries Java 开发工具中使用 RecordIOManager bean 来编写由**加入购物篮**按钮调用的 AddtoBasket servlet，该 servlet 通过减去由客户请求的数量并在会话中将商品加入 Basket Java bean 来更新 iSeries INVENTORY 数据库。AddtoBasket.jsp 代码样本如下:

```
public class AddtoBasket extends HttpServlet {

 //Inner class of AddtoBasket
 public class MyRecordIOManager extends RecordIOManager {
  .
  .
  .
  public MyRecordIOManager(
     String hostInfo1,
     String hostInfo2,
     String hostInfo3,
     String file,
     String lib)throws Exception{
   super(hostInfo1, hostInfo2,hostInfo3,file,lib);
   setFileAccessType(RecordIOManager.FILEACCESS_KEYED);
   setCommitLockLevel(RecordIOManager.COMMITLOCKLEVEL_ALL);
   //journal has the same name as the database file
   setJournal(file);
   //journal is in the same library as the database file
   setJournalLibrary(lib);
  }
  .
  .
  .
  public synchronized String updateDBFile(
   String id,
   String size,
   String quantity
   ) {
   .
   .
   .
   //opening the file
   try {
    if (openFile()) {
     record = readRecord(key);
     quantityAvailable = ((BigDecimal)
      record.getValueAt(0,sizeColumn)).intValue();
     totalQuantityAvailable = ((BigDecimal)
      record.getValueAt(0, 8)).intValue();
     if (quantityRequested <= quantityAvailable) {
      newQuantity =
         new BigDecimal(quantityAvailable - quantityRequested);
      totalNewQuantity =
         new BigDecimal
       (totalQuantityAvailable - quantityRequested);
      record.setValueAt(newQuantity, 0, sizeColumn);
      record.setValueAt(totalNewQuantity, 0, 8);
      // Note that we update the record but we don't commit
      // in case the customer decides to
      // empty the basket in which
      // case we call the rollBack method
      updateRecord(record);
      status = success;
   } else {
      status = notEnough;
     }
    } else
     status = accessError;
   } catch (Exception e) {
  e.printStackTrace();
    status = accessError;
   }

   //closing the file and adding
   try {
    closeFile();
```

```
  } catch (Exception e) {
   //in case of error rollback
   try {
    rollback();
   } catch (Exception e1) {
    e1.printStackTrace();
   }
   status = accessError;
  }

  return status;
 }
}

//init method of AddtoBasket servlet
public void init() {
 hostInfo = GetItems.getHostInfo();

}

public void doGet(HttpServletRequest req, HttpServletResponse res) {
 .
 .
 .
 Basket basket = (Basket) session.getAttribute("basket");
 MyRecordIOManager recIO = (MyRecordIOManager) session.getAttribute("recIO");
 if (basket == null) {
  basket = new Basket();
  session.setAttribute("basket", basket);
 }

 if(recIO == null){
  if (recIO == null) {
  try {
   recIO =
    new MyRecordIOManager(
     hostInfo[0],
     hostInfo[1],
     hostInfo[2],
     GetItems.getInventoryFile(),
     GetItems.getRetailLibrary());
  } catch (Exception e) {
   try {
    res.sendRedirect("errorPage.html");
    return;
   } catch (Exception e1) {
    e1.printStackTrace();
   }
  }
 }
 id = req.getParameter("id");
 size = req.getParameter(id + "S");
 quantity = req.getParameter(id + "Q");

 status = recIO.updateDBFile(id, size, quantity);
 session.setAttribute("recIO", recIO);

 if (status.equals("SUCCESS")) {
  basket.addItem(id, quantity, size);
  try {
   res.sendRedirect("ViewBasket.jsp");
   return;
  } catch (Exception e) {
 e.printStackTrace();
  }
  } else {
  if (status.equals("NOT_ENOUGH")) {
```

```
    try {
     res.sendRedirect("insufficient.html");
     return;
    } catch (Exception e) {
  e.printStackTrace();
    }
   } else
    if (status.equals("ACCESS_ERROR")) {
     try {
      res.sendRedirect("errorPage.html");
      return;
     } catch (Exception e) {
  e.printStackTrace();
     }
    }

  }

 }
```

8. 编写显示购物篮物品的 ViewBasket JSP 文件。
9. 为输入付款信息编写 customerinfo.html 表单，其中包括调用 ViewForConfirm.jsp 的**继续**按钮。
10. 编写显示整个购物篮的物品以及总余额的 ViewForConfirm.jsp。可以用与开发 ViewBasket.jsp 相同的方法来开发 ViewForConfirm.jsp，外加已计算的运费和将订单放在 iSeries ORDERS 数据库中的**确认**按钮。
11. 使用 iSeries Java 开发工具的 “iSeries 程序调用” 向导来创建 PLACEORD.java bean，该 bean 访问 RETAILSTOR 库中的 PLACEODR 服务程序。该向导创建一些 bean，供 Java 应用程序或 “Web 服务” 向导用来访问 iSeries ILE 程序。
    a. 要打开该向导，右键单击 “导航器” 视图中的 SV001585 并选择**新建 > 其它**。
    b. 在**新建**窗口中，单击 **iSeries > Java > 程序调用 Bean**。
    c. 在 “程序调用” 向导中，输入有关 iSeries ILE 程序名、库、程序类型及输入和输出参数的信息。
    d. 您可以在该向导的最后一个窗口中为 Java 应用程序和／或 Web 服务创建 bean。在此项目中，只需要为 Java 应用程序创建一个 bean。

    注: PLACEODR 服务程序采用结构数组并将每个元素都放置到 ORDERS 数据库的一个记录中，该程序生成一个订单号作为每个数组的输出。
12. 编写由 ViewForConfirm.jsp **确认**按钮调用的 PlaceOrder servlet。
    - 该 servlet 使用由 “iSeries 程序调用” 向导生成的 bean 来访问 iSeries 服务器并将订单放置在 RETAILSTOR 库的 ORDERS 数据库中。
    - 订单即购物篮中的商品，是作为结构数组发送至 ILE 程序的。
    - 此数组的每个结构都是购物篮中的一种商品。
    - bean 调用的 PLACEORD RPG 服务程序将订单号作为输出参数返回并将它放置到会话中。

    以下代码段显示 PlaceOrder servlet 是如何使用 PLACEORD bean 的:

```
.
 .
 .
public void init() throws ServletException {
 hostInfo = GetItems.getHostInfo();
 super.init();
```

```
 try {
  /* creating an instance of the PLACEORD bean created
     by iSeries Program Call Bean wizard */
  orderBean = new PLACEORD();
  orderBean.setConnectionData(hostInfo[0], hostInfo[1], hostInfo[2]);
 } catch (Exception e) {
 e.printStackTrace();
 }
}

.
.
.

public void doPost(HttpServletRequest request, HttpServletResponse response)
{  ...

 PLACEORD.Orditems_Struct inputStruct = null;

 // retrieving the order items from the basket
 Basket basket = (Basket) request.getSession().getAttribute("basket");
 AddtoBasket.MyRecordIOManager recIO = (AddtoBasket.MyRecordIOManager)
        request.getSession().getAttribute("recIO");
 if (basket == null || basket.size() == 0 || recIO == null) {
  try {
   response.sendRedirect("errorPage.html");
  } catch (IOException e) {
 e.printStackTrace();
  }
  } else {
  items = basket.elements();
  // setting array of structure elements
  while (items.hasMoreElements()) {
   item = (String[]) items.nextElement();
   inputStruct = orderBean.getOrdItemAr(j);
   inputStruct.setItemNo(new BigDecimal(item[0]));
   inputStruct.setQuantity(new BigDecimal(item[1]));
   inputStruct.setSizeOrd(item[2]);
   j = j + 1;
  }
  // setting the rest of the array elements to dummy values
  for (int i = j - 1; i < 100; i++) {
   inputStruct = orderBean.getOrdItemAr(i);
   inputStruct.setItemNo(new BigDecimal(0));
   inputStruct.setQuantity(new BigDecimal(0));
   inputStruct.setSizeOrd("s");
  }

  // setting the other two input parameters of the bean
  orderBean.setNumOfItems(new BigDecimal(j));
  orderBean.setBalance((BigDecimal) request.getSession().getAttribute("balance"));
  try {
   // invoking the iSeries program
   orderBean.invoke();

   // retrieving the order number from PLACEORD bean
   orderNumber = (orderBean.getRetCode()).toString();
   request.getSession().setAttribute("orderNumber", orderNumber);
   basket.empty();
   // commit this order now
   recIO.commit();
   response.sendRedirect("orderNumber.jsp");
   return;
  } catch (Exception e) {
   response.sendRedirect("errorPage.html");
 e.printStackTrace();
```

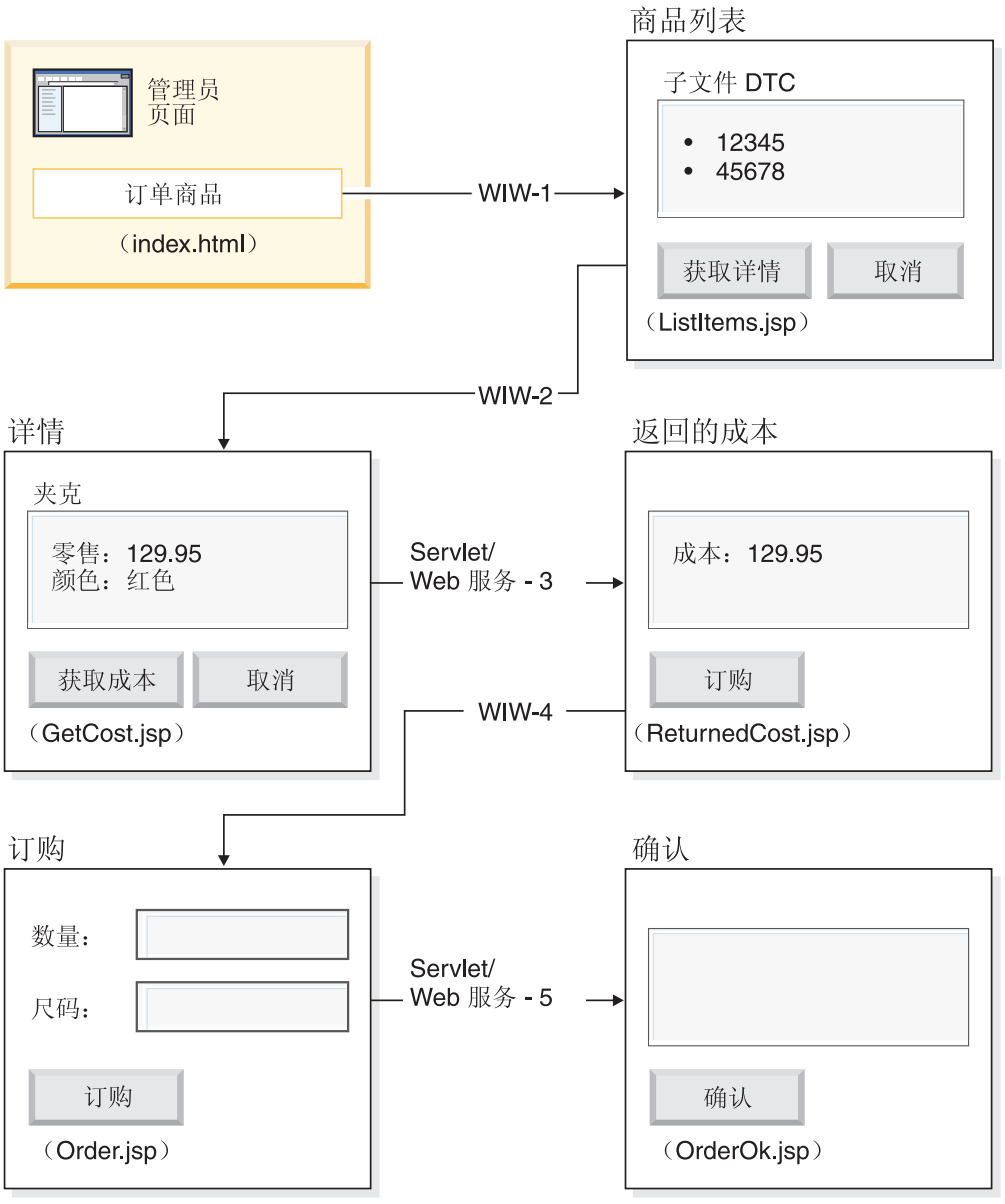
```
    }

  }
}
```

13. 编写 OrderNumber servlet，该 servlet 检索商品号并对客户显示它，且伴随有一条确认消息。如果客户尚未将任何商品加入购物篮，则确保返回错误页面。

第 7 章 高级模块 2: 创建使用 SV000514 和 SV001586 Web 服务的 Web 项目（SV000618）

简介

此项目演示可以如何使用 RPG 编程知识来为 iSeries Web 服务和 RPG 程序创建 Web 客户机。在此项目中您扮演管理员的角色，逐步完成一系列 Web 页面以确定库存数量并从批发商处为零售店订购其它存货。输入商品号，查看商品的详细信息，订购数量和尺码并接受确认。



（WIW = Web 交互向导）

因为这是一个高级模块，所以指示信息不会指导您逐步执行创建项目的每个步骤，但是给出了用来创建这种项目的开发步骤的概要步骤。此项目使用下列 Development Studio Client 组件:

- iSeries Web 开发工具 – 用来借助“页面设计器”创建 Web 页面，并将“Web 交互作用”向导中的输出与各种“Web 组件”合并
- 远程系统资源管理器 – 用来创建返回商品信息的 TNLSTITM RPG 服务程序
- Web 服务向导 – 用来生成查找商品价格和订购商品的 servlet 代理代码
- iSeries Java 开发工具 – 用来创建必需的 servlet
- WebSphere 测试环境 – 用来在通过 WebSphere Application Server 部署至 iSeries 服务器之前验证应用程序

在开始之前

只有在符合下列先决条件的情况下才能完成练习。第 7 页的第 3 章，『运行方案』对先决条件作了更详细的讨论。

- 对 iSeries 服务器具有 TCP/IP 访问权。
- 已使用命令 `STRTCPSVR *ALL` 启动了 iSeries 服务器。
- 已使用命令 `STRTCPSVR *WEBFACING` 启动了 WebFacing 服务器。
- 已将 WHOLESALE 和 RETAILSTOR 库复原至 iSeries 服务器。
- WHOLESALE、RETAILSTOR 和 QGPL 库都在库列表中。
- 您已完成了第 7 页的第 3 章，『运行方案』中的所有任务（可选的 WebSphere Application Server 任务除外，这些任务对于在工作台中测试应用程序不是必需的）。

创建 Web 页面、servlet、JSP 和 RPG 代码

要构造 SV000618 的组件:

1. 创建 Web 项目来保存将创建的所有文件。
2. 编写列示 iSeries 库存数据库中的商品的 ListItems JSP 文件。可以使用 iSeries Web 开发工具中的“页面设计器”来编写该 servlet。更具体地说，可以在“设计”视图中展开该页面并在“源”视图中添加适当的代码。还需要插入子文件“设计时控件”（DTC）来与 TNLSTITM RPG 服务程序交互，以用数据库记录填充子文件。可以在 DTC 控制设置中指定服务程序。

下一步，需要使用“Web 交互作用”向导来创建输入页面:

- 将 `ListItems.jsp` 指定为列示库存商品的输出页面，这将确保“Web 交互作用”向导创建 `ListItems.wit` 文件。
- 确保不要在“Web 交互作用”向导中指定任何程序调用，因为子文件 DTC 自动调用 TNLSTITM RPG 服务程序。该向导还生成充当调用 `ListItems.jsp` 页面的链接的 ListItemsWitServlet。
- 要复查由“Web 交互作用”向导生成的 `ListItemsWit.wit` 文件:
  1. 展开 **SV001618** 并双击 **ListItems.wit** 以显示文件的交互向导。
  2. 继续在向导中单击**下一步**来复查为交互作用指定的值。

下一步，需要使用“页面设计器”编写 GetCost JSP 输出页面，该页面从 ListItems.jsp 输入页面中获取输入。当用户单击 ListItems.jsp 页面上的商品时，GetCost.jsp 页面显示有关该商品的详细信息。

在创建了 GetCost.jsp 页面之后，使用“Web 交互作用”向导来创建 ListItems.jsp（选择为输入页面）和 GetCost.jsp（选择为输出页面）之间的 WitOrder 交互作用。

- 在向导的“程序调用”页面上，指定从 TNLSTITM RPG 服务程序对 GetDetail 过程和参数进行调用。
- 在该过程中，合并子文件 DTC API 以确定已选择了哪个子文件记录。该过程使用此信息来从 INVENTORY 数据库中检索所选记录并显示详细信息（包括 GetCost.jsp 上所选商品的图像）。
- 要复查由“Web 交互作用”向导生成的 WitOrderWit.wit 文件:
  1. 展开 **SV001618** 并双击 **WitOrder.wit** 以显示文件的交互向导。
  2. 继续在向导中单击**下一步**来复查为交互作用指定的值。

  注意，将 *flow* 参数指定为输出页面上的流量控制器。这样可使该参数的值确保显示正确的 JSP 文件。

下一步，需要从项目 SV000514 中导入“Web 服务定义语言”（WSDL）文件，这样管理员可以通过按**获取成本**按钮来从批发商处检索商品的当前成本。

- **获取成本**按钮从项目 SV000514 中调用 QryProdCostServlet.jsp 和相应的 Web 服务。
- 使用“Web 服务”向导和已导入的 WSDL 文件来生成调用 Web 服务所需的 Java 代理代码。
- QryProdCostServlet.jsp 采用 GetCost.jsp 页面中的输入，使用 Java 代理代码来调用 SV000514 Web 服务以查找所选商品的成本，并将该成本显示在称为 ReturnedCost.jsp 的页面中。
- 要查看 QryProdCostServicesProxy.java 代码和 QryProdCostServlet.java:
  1. 展开 **SV001618 > source > proxy > SOAP**。
  2. 双击 **QryProdCostServicesProxy.java**。
  3. 对于 QryProdCostServlet.java，在 **SV001618 > source** 中双击 **QryProdCostServlet.java**，并注意它是如何实例化 Java 代理代码的。

下一步，需要使用“Web 交互作用”向导来链接 ReturnCost.jsp（作为输入）和 Order.jsp（作为输出），这样，管理员可以单击**订购**按钮来从批发商处订购所选商品。

- 借助此交互作用，因为两个页面的链接足以显示正确的信息，因此不需要使用程序调用。
- 要查看 WitPlaceOrder.wit:
  1. 展开 **SV001618**。
  2. 双击 **WitPlaceOrder.wit** 以打开交互作用。
  3. 继续在向导中单击**下一步**来复查指定的值。

下一步，使用 SV001586 Web 服务以便管理员可以指定所订购商品的尺码和数量。

- 将 SV001586 WSDL 文件导入此项目，生成调用 Web 服务的 Java 代理代码，并编写在用户按 Order.jsp 中的**订购**按钮时调用的 OrderSupplyServlet。

- servlet 收集 Order.jsp 中的信息，调用 Web 服务 Java 代理代码（该代理代码调用 SV001586 Web 服务并订购商品）。
- 如果订购成功，则该 servlet 显示 OrderOK.jsp；如果订购不成功，则显示错误页面。
- 要了解如何实例化已生成的 Web 服务代理以及如何调用它来使用 SV001586 Web 服务:
  1. 展开 **SV000618 > source**。
  2. 双击 **OrderSupplyServlet.java** 并检查物品。

在部署到 WebSphere Application Server 之前

在将您的应用程序部署到 WebSphere Application Server 之前，需要更改 SV000514 和 SV001586 Java 类文件中的特定 URL，这样才能使应用程序正确运行。

对于 SV000514:

- 在 Web 服务代理类 QryProdCostServicesProxy 中，定义了一个变量来包含要调用的 Web 服务的 URL。
- 当第一次创建代理时，此 URL 被设置为 http://localhost:9080/SV000514/servlet/rpcrouter。
- 在变量设置为此值的情况下，调用 IDE 中项目 SV000514 中的 Web 服务，如第 14 页的『在工作台中运行应用程序』中所演示的那样。
- 在部署此应用程序之前，需要更改此 URL 值以指向在 iSeries IFS 目录中部署了 EAR 文件 SVWholeSale.ear 的位置。

对于项目 SV001586:

- 在 Web 服务代理类 OrderSupplyServicesProxy 中，定义了一个变量来包含要调用的 Web 服务的 URL。
- 当第一次创建代理时，此 URL 被设置为 http://localhost:9080/SV001586/servlet/rpcrouter。
- 在变量设置为此值的情况下，调用 IDE 中项目 SV001586 中的 Web 服务，如第 14 页的『在工作台中运行应用程序』中所演示的那样。
- 在部署此应用程序之前，需要更改此 URL 值以指向在 iSeries IFS 目录中部署了 SVWholeSaleEAR.ear 文件的位置。

第 8 章 声明

Note to U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

本信息是为在美国提供的产品和服务编写的。IBM 可能在其它国家或地区不提供本文档中讨论的产品、服务或功能特性。有关您当前所在区域的产品和服务的信息，请向您当地的 IBM 代表咨询。任何对 IBM 产品、程序或服务的引用并非意在明示或暗示只能使用 IBM 的产品、程序或服务。只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务，都可以代替 IBM 产品、程序或服务。但是，评估和验证任何非 IBM 产品、程序或服务，则由用户自行负责。

IBM 公司可能已拥有或正在申请与本文档内容有关的各项专利。提供本文档并未授予用户使用这些专利的任何许可证。您可以用书面方式将许可证查询寄往:

IBM Director of Licensing IBM Corporation North Castle Drive Armonk, NY 10504-1785 U.S.A.

有关双字节（DBCS）信息的许可证查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往:

IBM World Trade Asia Corporation Licensing 2-31 Roppongi 3-chome, Minato-ku Tokyo 106, Japan

本条款不适用英国或任何这样的条款与当地法律不一致的国家或地区: 国际商业机器公司以“按现状”的基础提供本出版物，不附有任何形式的（无论是明示的，还是默示的）保证，包括（但不限于）对非侵权性、适销性和适用于某特定用途的默示保证。某些国家或地区在某些交易中不允许免除明示或默示的保证，因此本条款可能不适用于您。

本信息中可能包含技术方面不够准确的地方或印刷错误。此处的信息将定期更改；这些更改将编入本资料的新版本中。IBM 可以随时对本资料中描述的产品和／或程序进行改进和／或更改，而不另行通知。

本信息中对非 IBM Web 站点的任何引用都只是为了方便起见才提供的，不以任何方式充当对那些 Web 站点的保证。那些 Web 站点中的资料不是 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

本程序的被许可方如果要了解有关程序的信息以达到如下目的: （i）允许在独立创建的程序和其它程序（包括本程序）之间进行信息交换，以及（ii）允许对已经交换的信息进行相互使用，请与下列地址联系:

Lab Director IBM Canada Ltd. Laboratory 8200 Warden Avenue Markham, Ontario, Canada L6G 1C7

只要遵守适当的条件和条款，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本资料中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际程序许可证协议或任何同等协议中的条款提供。

涉及非 IBM 产品的信息可从这些产品的供应商、其出版说明或其它可公开获得的资料中获取。IBM 没有对这些产品进行测试，也无法确认其性能的精确性、兼容性或任何其它关于非 IBM 产品的声明。有关非 IBM 产品性能的问题应当向这些产品的供应商提出。

本资料中包含用于日常业务运作的数据和报表的示例。为了尽可能完整地说明这些数据和报表，这些示例中可能包括了个人、公司、品牌和产品的名称。所有这些名称都是虚构的，如与实际商业企业所用的名称和地址有雷同，纯属巧合。

版权许可:

本信息包括源语言形式的样本应用程序，这些样本说明不同操作平台上的编程方法。如果是为按照在编写样本程序的操作平台上的应用程序编程接口（API）进行应用程序的开发、使用、经销或分发为目的，您可以任何形式对这些样本程序进行复制、修改、分发，而无须向 IBM 付费。这些示例并未在所有条件下作全面测试。因此，IBM 不能担保或暗示这些程序的可靠性、可维护性或功能。用户如果是为了按照 IBM 应用程序编程接口开发、使用、经销或分发应用程序，则可以任何形式复制、修改和分发这些样本程序，而无须向 IBM 付费。

凡这些样本程序的每份拷贝或其任何部分或任何衍生产品，都必须包括如下版权声明:

(C)（贵公司的名称）（年）。此部分代码是根据 IBM 公司的样本程序衍生出来的。(C) Copyright IBM Corp. 1992, 2003. All rights reserved.

编程接口信息

编程接口信息用来帮助您使用此程序来创建应用程序软件。

通用编程接口允许您编写获取此程序工具的服务的应用程序软件。

然而，本信息还可能包含诊断、修改和调整信息。诊断、修改和调整信息用来帮助您调试应用程序软件。

警告: 切勿使用此诊断、修改和调整信息作为编程接口，因为它随更改而变化。

商标和服务标记

- 400
- AFP
- AIX
- AIX windows
- APPN
- Application System/400
- AS/400
- CUA
- DB2
- DB2 Extenders
- DB2 Universal Database
- @server
- GDDM
- IBM
- OS/390
- OS/400
- POWER2
- PowerPC
- PROFS
- RPG/400
- RS/6000

- AS/400e
- BookManager
- C Set ++
- C/400
- CICS
- CICS/400
- CICS/ESA
- COBOL/2
- COBOL/400
- Common User Access
- IBMLink
- Integrated Language Environment
- iSeries
- Language Environment
- MQSeries
- Network Station
- Open Class
- Operating System/2
- Operating System/400
- OS/2
- S/390
- SAA
- SQL/400
- System/36
- System/38
- VisualAge
- VTAM
- WebSphere

InstallShield 是 InstallShield Corporation 的商标。

Intel 和 Pentium 是 Intel Corporation 在美国和／或其它国家或地区的商标。

Java 和所有基于 Java 的商标和徽标是 Sun Microsystems, Inc. 在美国和／或其它国家或地区的商标或注册商标。

Rational 是国际商业机器公司和 Rational Software Corporation 在美国和／或其它国家或地区的商标。

Lotus、Lotus Notes 和 Domino 是 Lotus Development Corporation 在美国和／或其它国家或地区的商标。

ActiveX、Microsoft、SourceSafe、Visual C++、Visual SourceSafe、Windows、Windows NT、Win32、Win32s 和 Windows 徽标是 Microsoft Corporation 在美国和／或其它国家或地区的商标。

Netscape Navigator 是 Netscape Communications Corporation 的商标。

UNIX 是 The Open Group 的注册商标。

其它公司、产品和服务名称可能是其它公司的商标或服务标记。

程序号:　5724-A81

中国印刷