



IBM Software Group

IBM Rational Application Developer V7.5

Java EE 5 tools



@business on demand.

© 2008 IBM Corporation
Updated April 21, 2015

This presentation provides an overview of Java™ EE 5 development tools that are available in Rational® Application Developer V7.5.

Agenda

- Overview of Java EE 5
- Java EE 5 development tools
 - ▶ Enterprise projects
 - ▶ Annotations
 - ▶ Web technologies



The first section of this presentation includes an overview of the Java EE 5 specification, which includes many important technology updates like EJB 3.0 and the Java Persistence API. The rest of the presentation discusses the tools that the Rational Application Developer workbench provides for building, compiling, and testing Java EE 5 applications. This includes creating projects, working with annotations, and using new Web development tools. Other tools for Web services, EJB, and JPA development are covered in separate modules.

Overview of Java EE 5

- Java EE 5 focuses on reducing complexity, using intelligent defaults to simplify the programming model
 - ▶ Less boilerplate code, optional deployment descriptors
- Enterprise JavaBeans (EJB) 3.0
- Java Persistence API (JPA)
- Web services (JAX-WS 2.0)
- Web application technologies
 - ▶ JSP 2.1
 - ▶ JSF 1.2
 - ▶ Servlet 2.5



The main goal of Java EE 5 is to simplify the programming model. The new specification aims to make it as easy as possible to implement simple things, while keeping complex things possible. In many cases this is accomplished by utilizing contextually appropriate default values, allowing you to override the defaults when needed. When it is possible for the container to figure something out, it will do so, rather than requiring a developer to provide unnecessary information. With this simplification, regular Java developers should be more able to make the transition to developing enterprise Java applications. The EJB specification has been heavily revised, and is now a plain-old Java object based programming model. This approach extends even to data persistence, with the introduction of JPA, the Java Persistence API. Java EE 5 also incorporates the latest Java Web services standards, such as JAX-WS, the Java API for XML Web services. Web application technologies, such as servlets, JavaServer Pages, and JavaServer Faces have also been revised in this specification.

Java EE 5 development tools

- Java EE projects
 - ▶ Creating projects
 - ▶ Working with facets
 - ▶ Application packaging
- Annotations capabilities
- Web technologies
 - ▶ Creating Web pages
 - ▶ JavaServer Faces
- EJB 3.0, JPA, Web services – covered in other modules



The rest of this presentation focuses on the tools in the Rational Application Developer V7.5 workbench that support Java EE 5 development. The first section describes how to create projects, how to work with project facets, and changes in the Java EE 5 application packaging model. The Java EE 5 programming model is heavily annotations-based, and tools for doing annotation-based development are covered in the next section of the presentation. Finally, the presentation covers new tools for Web development, including a graphical Web page designer and enhanced tools for developing JavaServer Faces. There are many other specialized tools in the workbench for working with EJB 3.0 artifacts, JPA constructs, and doing Web services development. These tools are covered in separate modules.

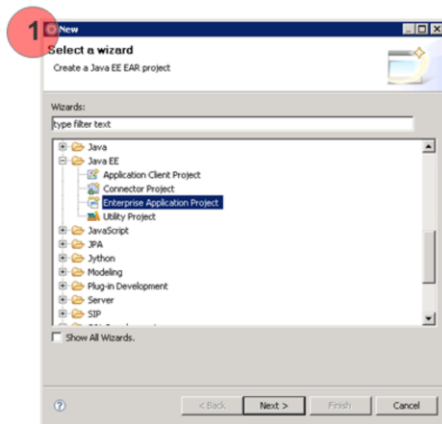
Section

Java EE projects

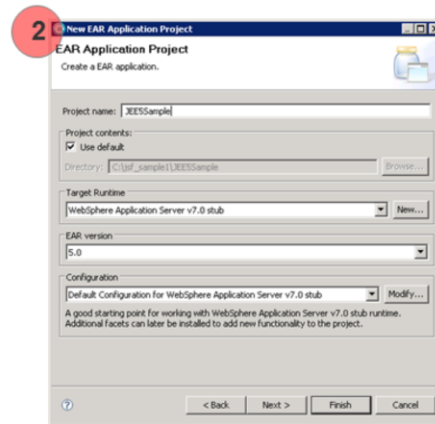


This section describes how to create Java EE projects, how to manipulate project facets, and discusses changes in application packaging that are a part of the Java EE 5 specification.

Creating an enterprise project



Start the EAR Application Project wizard by selecting **File > New > Project > Java EE > Enterprise Application Project**

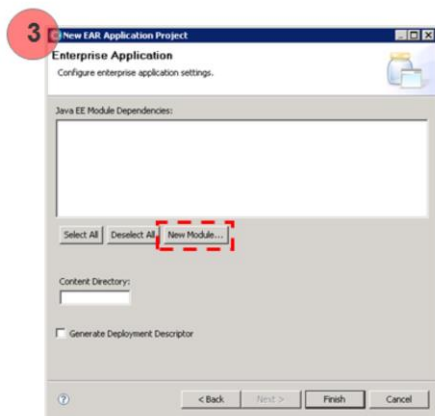


Choose a target runtime and EAR version – the default for WebSphere® Application Server V7 is Java EE 5

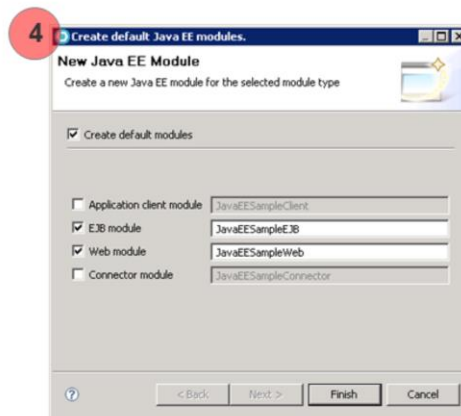


There is a wizard to walk you through creating a Java EE project. First, launch the EAR Application Project wizard using the product menus. On the second panel of the wizard, provide a name for the project and select other runtime configuration options. The target runtime that you associate with the project is the runtime that you are going to use to test your application. If you choose WebSphere Application Server V7, then the default EAR version is 5.0 for Java EE 5. You can select a different Java EE level for the project using the dropdown menu. The configuration profile that you choose determines which facets are associated with your project. Either choose a configuration profile from the dropdown menu or use the Modify button to manually configure the facets for your project.

Creating an enterprise project (continued)



Optionally, create new modules to include in the enterprise application

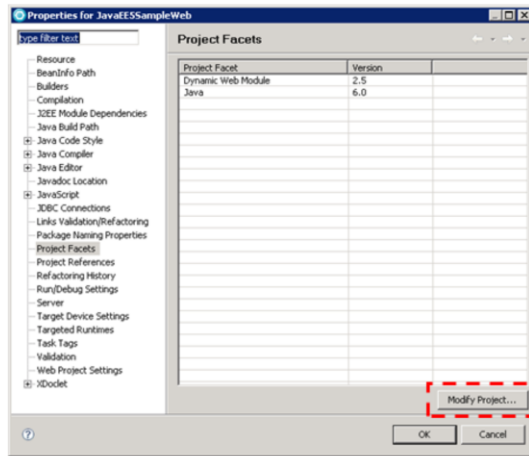


Standard module types are available – projects are created in the workspace for the new modules



After choosing the runtime and workspace configuration options for your project, you have the option of creating modules for your enterprise. If you do not want to create any modules, you can exit the wizard by clicking Finish on the Enterprise Application settings panel – this creates an empty EAR project that you can later customize with application modules. Alternatively, if you want to create application modules at project creation time, click the New Module button to choose which modules to create. On the new Java EE module page, you have the option of creating an application client, EJB, Web, or connector module. Choose the options you want and click Finish to create the enterprise application.

Displaying project facets



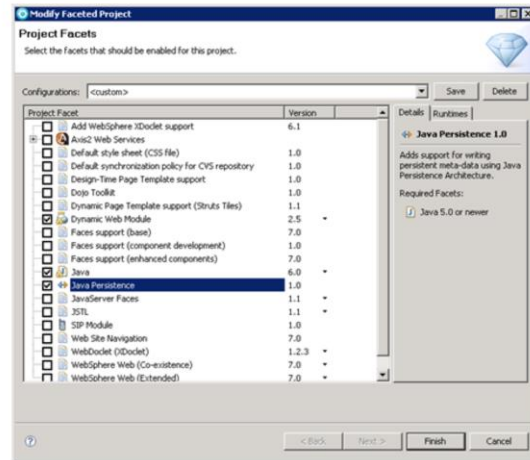
- The features available in a project are defined through **Project Facets**
- Display the configured facets by right-clicking on a project and selecting **Properties > Project Facets**
- Facet names and versions are provided
- To add new facets to your project or change the facet versions, click the **Modify Project...** button



Java EE projects typically have facets, which describe the functions that are contained in the project. When you add a facet to a project, the project is configured to perform a certain task, fulfill certain requirements, or have certain characteristics. For example, the EAR facet sets up a project to function as an enterprise application by adding necessary metadata and setting up the classpath for a project. Using product tools, you can easily add EJB 3.0 or JPA facets to a plain Java project, which converts the project to a Java EE project.

Changing project facets

- Configure facets on the **Modify Faceted Project** panel
 - ▶ Add or remove facets
 - ▶ Change the version of a facet
- **Example:**
 - ▶ To use the Java Persistence API in a Web project, select the check box next to Java Persistence
- **Example:**
 - ▶ To change the targeted JDK level for a project, click the arrow in the Version column next to Java and choose the appropriate JDK level



By modifying the project facets, you control the technologies that are available to the application that you are developing in that project. You can add and remove facets to your project and change the version of a facet. For example, to use the Java Persistence API in a Web project, select the check box next to Java Persistence to add that facet to the project. If you want to change the target JDK level for a project, click the arrow in the version column next to Java and choose the appropriate JDK level.

Application packaging

- Java EE 5 provides simplified packaging rules for enterprise applications:
 - ▶ Web applications use .WAR files
 - ▶ Resource adapters use .RAR files
 - ▶ Enterprise applications use .EAR files
 - ▶ The lib directory contains shared .JAR files
 - ▶ A .Jar file with main-class implies an application client
 - ▶ A .Jar file with @stateless annotation implies an EJB application
 - ▶ Many simple applications no longer require deployment descriptors, including
 - EJB applications (.JAR files)
 - Web applications that use JSP technology only
 - Application clients
 - Enterprise applications (.EAR files)



Java EE 5 provides a simplified packaging model for enterprise applications. In most cases, deployment descriptors are optional, so the specification relies on file name extensions and package contents to identify application components. For example, a WAR file is a Web application, a RAR file is a resource adapter, and an EAR file is an enterprise application. Since JAR file packages can contain different types of application components, the packaging model relies on the contents of the archive to identify the package type. For example, a JAR file with a main-class is an application client, and a JAR file with EJB annotations like the @Stateless annotation is an EJB module. Most applications no longer require deployment descriptors; include EJB applications, enterprise applications, and some Web applications. If a Web application contains technologies other than JavaServer Pages – for example, if the application contains Servlets – then a deployment descriptor is still required. You can still include a deployment descriptor in your application package, even if it is not required. In that case, the information in the deployment descriptor overrides information in the annotations in the application content.

Java EE 5 annotations overview

- Java EE 5 annotations let you embed resources, dependencies, services, and life-cycle notifications in your source code
 - ▶ Replace descriptors for most purposes
 - ▶ Reduce the number of artifacts you have to maintain
 - ▶ Remove the need for marker interfaces (like `java.rmi.Remote`)
 - ▶ Allow application settings to be visible in the component they affect



Java EE 5 supports the injection of annotations into your source code, so that you can embed resources, dependencies, services, and life-cycle notifications in your source code, without having to maintain these artifacts elsewhere. An annotation is a modifier or Metadata tag that provides additional data to Java classes, interfaces, constructors, methods, fields, parameters, and local variables. Annotations replace boilerplate code, common code that is required by certain applications. For example, an annotation can replace the paired interface and implementation required for a Web service. Annotations can also replace additional files that programs require, which are maintained separately. By using an annotation, this separate file is no longer required. For example, annotations can replace the need for a separately maintained deployment descriptor for JavaBeans.

Java EE 5 annotation examples

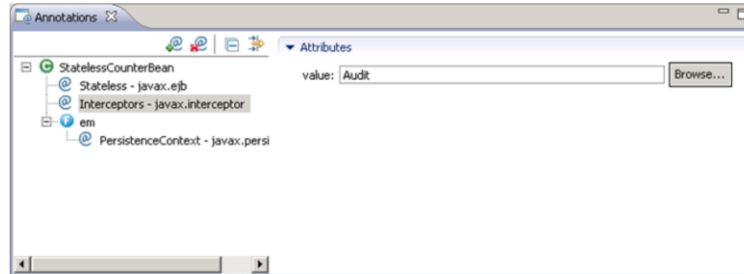
- Java EE 5 provides annotations for these tasks, among others:
 - ▶ Developing Enterprise JavaBeans applications
 - ▶ Defining and using Web services
 - ▶ Mapping Java technology classes to databases
 - ▶ Specifying external dependencies
 - ▶ Specifying deployment information, including security attributes



Java EE 5 defines several annotations that can be injected into your source code. To declare an annotation, you precede the keyword with an "@" sign (@). There are special annotations defined for doing EJB development, Web services development, and using the Java Persistence API to map Java objects to databases. Annotations also allow you to inject resources directly into your application. For example, rather than having to do a complex lookup and cast an EJB as an appropriate object type to be able to access its data in an application, you can use a simplified annotation-based programming model to create an instance of an EJB in your application. In Java EE 5, you can create an instance of an EJB in your application using the @EJB annotation, the name of the EJB you want to use, and the name of the variable that will contain the EJB. The line "@EJB ShoppingCart myCart", for example, injects a callable instance of the EJB ShoppingCart into a Java program.

Annotations view

- Will display in your workspace in the Java EE Perspective
 - ▶ If needed, open the Annotations view explicitly under **Window > Show View > Other > Annotations > Annotations**
- Annotations view provides
 - ▶ Tree view of the annotations in a class
 - ▶ Ability to add, edit, or delete annotations
 - ▶ Menus for defining annotation attributes



The Annotations view provides a way for you to create, edit, browse, and generally keep track of the annotations that you use in your applications. This view detects annotation types from the metadata in the annotation tag implementation class to provide rich editing capability, including the ability to indicate what attributes can be defined for an annotation and which attributes are required and to provide default validation and user assistance for each annotation. This view also displays in an easy-to-navigate tree structure all of your annotations in your Java classes. You can add and remove annotations using the toolbar icons above the tree. You can filter the tree by typing a filter value in the type filter text field. The view displays implied attributes, default values for attributes that are not required, and annotation values that are being overridden by deployment descriptors.

Section

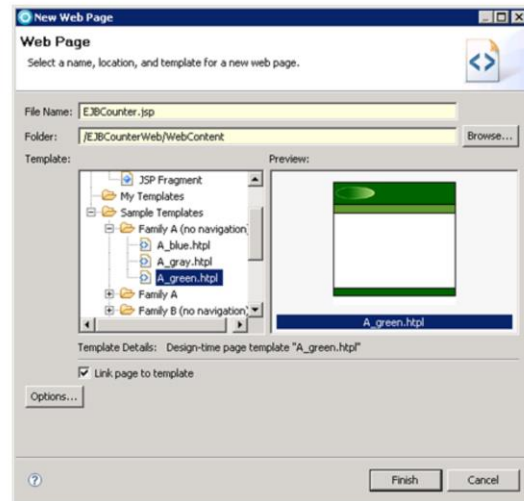
Web technologies



The last section of this presentation describes new tools in Rational Application Developer for developing Web applications.

Creating Web pages

- Create a Web page by right-clicking on a project in the Project Explorer and selecting **New > Web Page**
- Pages can be created based on existing templates
 - ▶ Basic templates for HTML/XHTML, JSP, JSP fragments
 - ▶ Graphical templates with optional navigation components



The development workbench contains a simple wizard for creating a new Web page. It allows you to create artifacts for HTML, XHTML, JSP, and JSP fragments. Open the new Web page panel, provide a name for the page, and indicate what type of page you are creating. You can also choose from one of several pre-defined page templates. Many of the templates offer optional navigation components.

Web page designer

Use the icons to adjust the split orientation

Split panes allow viewing the source and the design panel at the same time

Choose a tab to change between the split view, design only, source only, or a preview of the page

Drag and drop items from the Palette to add them to the page designer

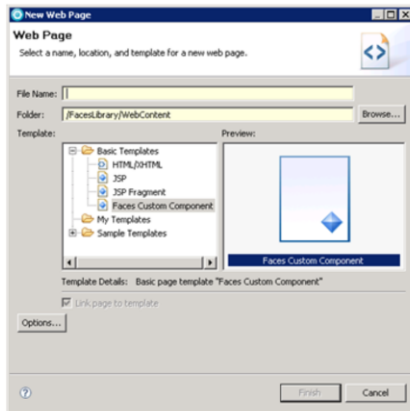
Items are inserted in a flow layout – drag by the upper left corner to use absolute positioning

Apply page templates and configure site navigation

Java EE 5 tools © 2008 IBM Corporation

The Web page designer is a multi-tabbed editor that makes it easy to edit HTML, XHTML, page template files, JavaServer Pages, Faces JavaServer Pages, and embedded JavaScript code. By clicking on a Page Designer tab, you can display multiple representations of each page: Design, Source, Split, and Preview. The Design page is the WYSIWYG environment that enables you to create and work with a file while viewing its elements on the page. For example, you can see the graphics that you have inserted into the file and continually check the visual presentation of the Web page as you design it. You can drag page components from the Palette to the Design panel. The Source page enables you to view and work with a file's source code directly. The Outline and Properties views both have features that supplement the Source page. They combine the Source page and either the Design page or the Preview page in a split screen view. Changes that you make in one part of the split screen can immediately be seen in the other part of the split screen. You can split the screen horizontally or vertically. The Preview page shows you how the current page is likely to look when viewed in an external Web browser. To preview any dynamic content (such as JSP tags), you use the Run on Server option from the page's pop-up menu in the Enterprise Explorer view.

JavaServer Faces custom components



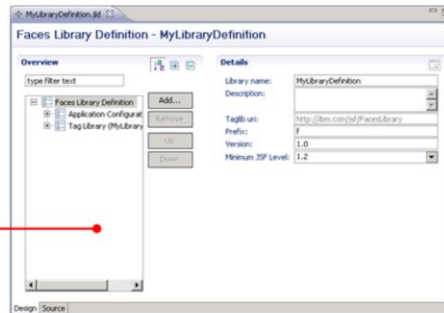
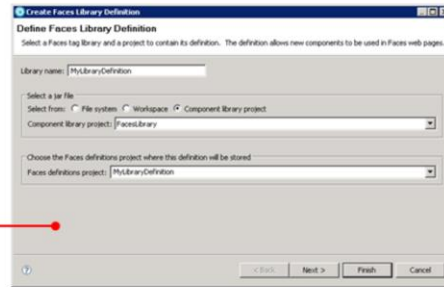
- Build libraries of customized JSF components
 - ▶ Combine existing JSF components to build a customized library
 - ▶ New components can be made available in the designer palette, used like any other JSF control
- Start with a component library project
 - ▶ **New > Project > Web > Faces Component Library Project**
- Generate components using the standard Web page wizard
 - ▶ Edit components in the standard Web page editor
 - ▶ Tool will automatically generate JSF classes and configuration files



A custom component library contains new and modified JSF components that you can use in your Web applications and distribute the libraries to your development team. A custom component library is a Faces-enabled Web project with the Faces component development facet selected. Each custom component is created in a separate JSP. After one or more JavaServer Pages are created, the project is built and a new custom tag library is generated automatically. The custom JSF components can be added to Web pages.

Creating a JSF library

- Creating a library definition allows you to:
 - ▶ Add custom components to the palette
 - ▶ Share them with other developers
- Right-click a Component Library project and choose **New > Faces Library Definition**
 - ▶ Can also directly create Faces Library Definition projects for vendor libraries
- The library definition editor allows you to customize the library configuration



A Faces Library Definition contains project resources needed for a component library and the metadata necessary for the interpretation of JSF tags. To configure the Faces Library Definition, double-click your Faces Library Definition to open the library definition file in the editor. Your library definition file has a JLD extension. Once you have configured your library definition, save the file. The definition is updated. To update the file, click Update Library Definition then click Perform update. The update process finds new tags and attributes in the tag library and adds them to the library definition.

Section

Summary



This section provides a summary of this presentation.

Summary

- Rational Application Developer provides rich tools for Java EE 5 development
 - ▶ Specialized projects for enterprise applications
 - ▶ Annotations tools to simplify working with annotations
 - ▶ Drag and drop Web development tools



Rational Application Developer V7.5 provides rich tools for Java EE 5 application development. There are specialized tools available for creating and working with enterprise application projects. The workbench includes an annotations view that allows you to visually browse through and edit the annotations in your application. Web application development tools have been enhanced with a drag-and-drop page designer tool and the ability to create customized JSF libraries.

Trademarks, copyrights, and disclaimers

IBM, the IBM logo, ibm.com, and the following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

Rational WebSphere

If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of other IBM trademarks is available on the Web at "Copyright and trademark information" at <http://www.ibm.com/legal/copytrade.shtml>

Rational is a trademark of International Business Machines Corporation and Rational Software Corporation in the United States, Other Countries, or both.

EJB, Enterprise JavaBeans, Java, JavaScript, JavaServer, JDK, JSP, and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

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

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

© Copyright International Business Machines Corporation 2008. All rights reserved.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.

