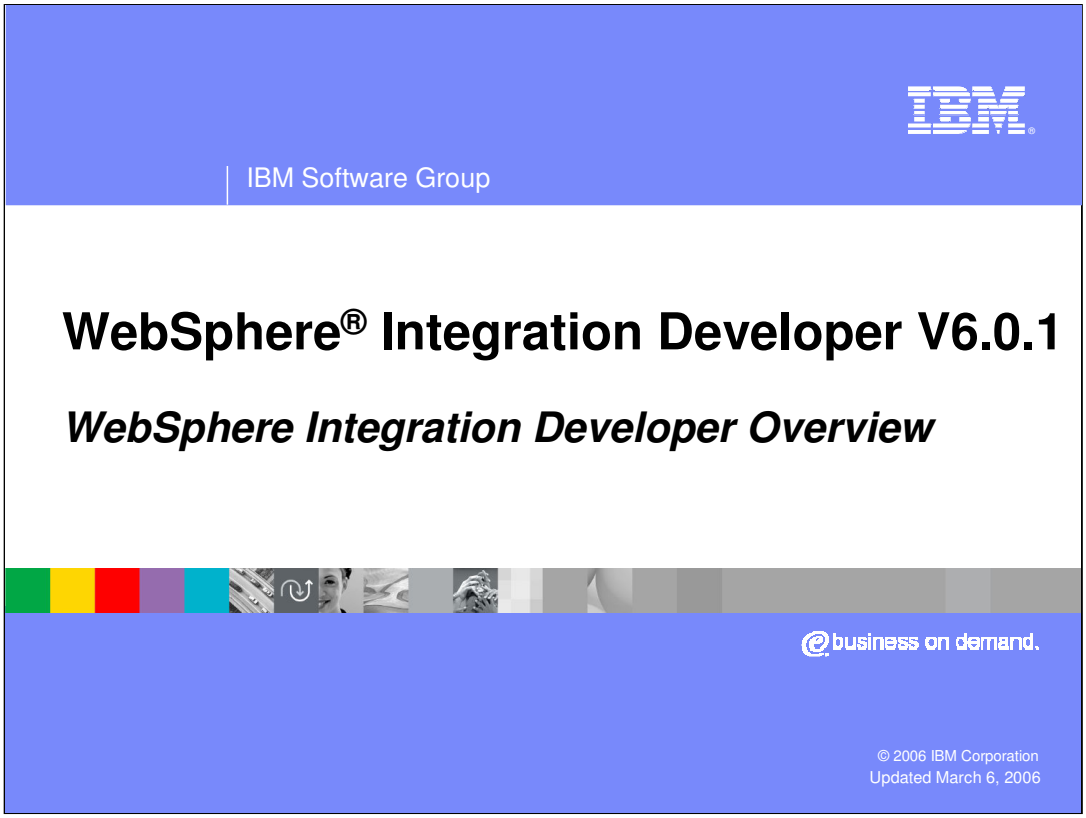


This presentation will provide an Overview of WebSphere Integration Developer V6.0.1.

Goals

- Introduce WebSphere Integration Developer V6.0.1 product and features
- Introduce editors and views associated with integration development

© 2006 IBM Corporation

The goals of this presentation are to introduce the WebSphere Integration Developer V6.0.1 product and features and several of the important editors and views associated with integration development.

Agenda

- Overview
- Concepts
- Integration Developer Basics
- Summary and References

© 2006 IBM Corporation

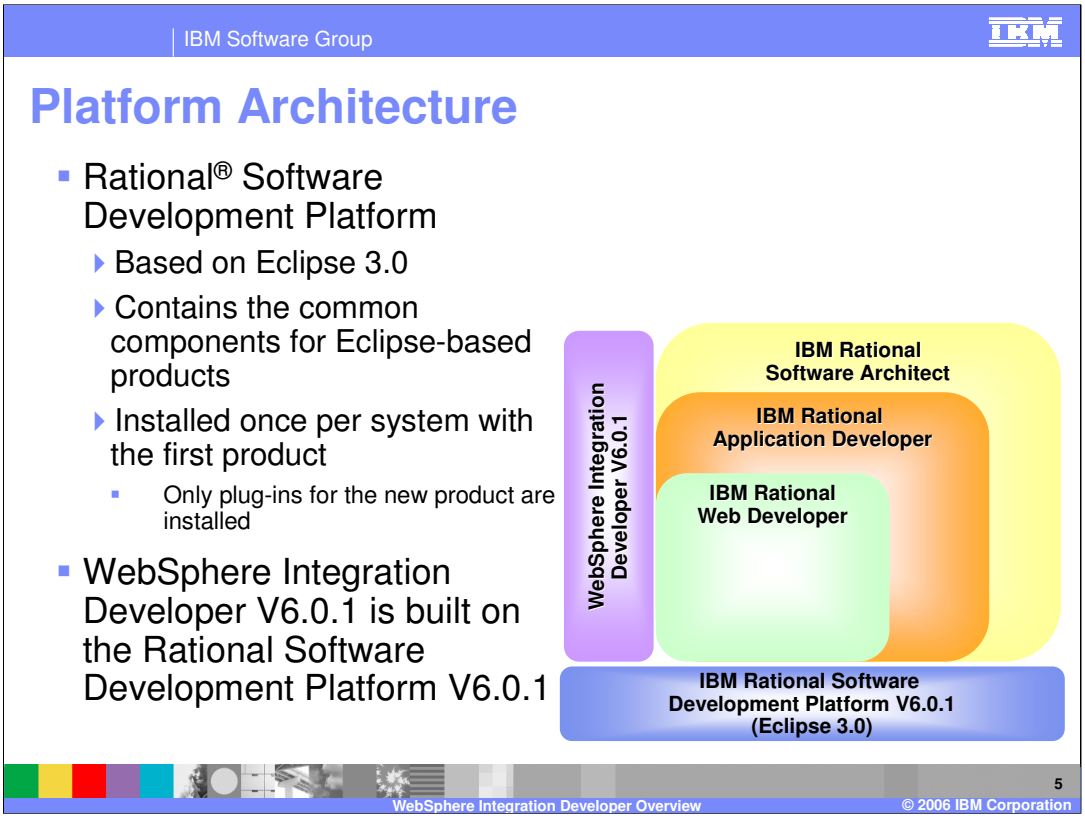
The agenda for this presentation begins with an overview of WebSphere Integration Developer. Then some background concepts and basics of the user interface will be presented.

Product Description

- Development environment for building integrated applications based on a service oriented architecture (SOA)
- Authoring tool for WebSphere Process Server V6.0.1
- Authoring tool for WebSphere Enterprise Service Bus V6.0.1
- Application development is based upon the service component architecture (SCA)
- Tools are aimed at helping decouple business logic and mediation logic from implementation details

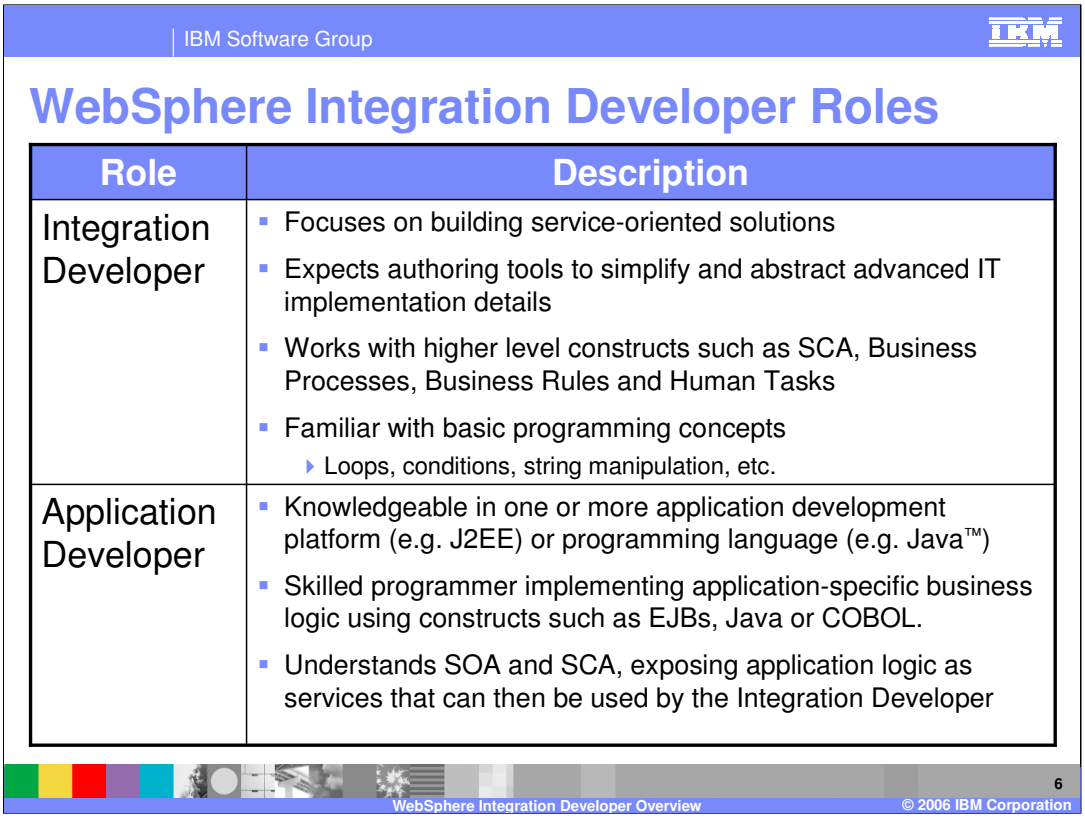
© 2006 IBM Corporation

WebSphere Integration Developer V6.0.1 is the development environment for building integrated business applications targeted for WebSphere Process Server V6.0.1 and WebSphere Enterprise Service Bus V6.0.1. WebSphere Process Server is the next generation business process integration server that provides an application framework for building and running applications based upon a service oriented architecture (SOA). SOA support in WebSphere Process Server is based on a new programming model referred to as Service Component Architecture (SCA). WebSphere Enterprise Service Bus is also an SOA based runtime utilizing SCA and provides a mechanism to allow loose coupling of service requestors and service providers, which is referred to as mediation. One of the primary purposes of WebSphere Integration Developer is to provide the appropriate tools to easily build and test SCA-based applications targeted for WebSphere Process Server and WebSphere Enterprise Service Bus. Both SCA and the tools support for SCA are aimed at helping developers decouple business logic and mediation logic from implementation details.



As this graphic shows, WebSphere Integration Developer V6.0.1 is built on the Rational Software Development Platform V6.0.1. Rational Software Development Platform is based on Eclipse 3.0 technology, and each IBM product built on this platform will co-exist and share plug-ins with other Rational Software Development Platform based products. Rational Software Development Platform is installed once per system with the first product that is installed. As other products built on this platform are installed on the system, only the necessary plug-ins are installed.

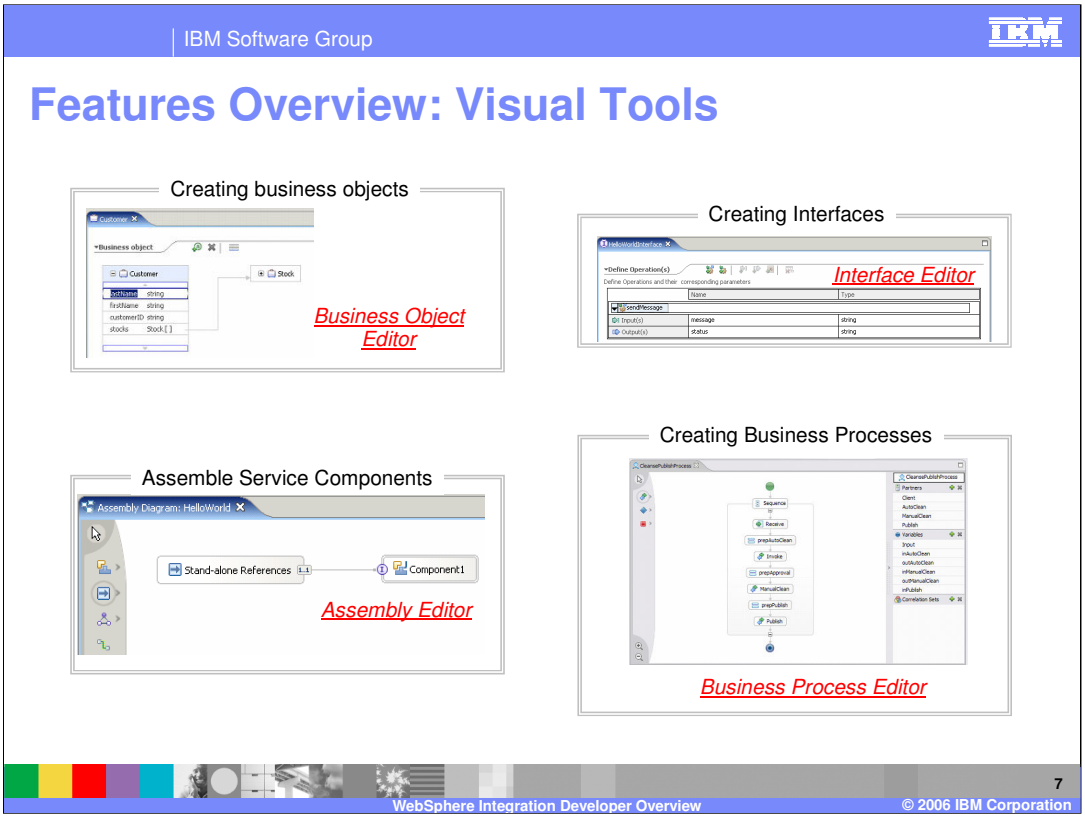
Refer to the WebSphere Integration Developer installation guide for more information on WebSphere Integration Developer coexistence with other Rational Software Development Platform based products. This document can be accessed from the installation launchpad for WebSphere Integration Developer.



A description of the two primary user roles associated with WebSphere Integration Developer is provided here.

The Integration Developer is the primary user role for WebSphere Integration Developer. This user type is focused on building service oriented solutions and expects the authoring tools to simplify much of the advanced implementation details associated with building complex business applications and workflows. Although the Integration Developer is typically familiar with basic programming concepts such as loops, conditions, and string manipulations, this user might not be an expert in any particular implementation platform or programming language. WebSphere Integration Developer provides the Integration Developer with a development environment for defining Business Processes, Business Rules, Human Tasks and other high level constructs without having an extensive knowledge of the underlying implementation technologies. These then become building blocks which the Integration Developer uses to compose a complex business application using SCA.

Another important user role is the Application Developer. Unlike the Integration Developer, the Application Developer is typically knowledgeable in one or more programming languages or application development platforms. The Application Developer is knowledgeable about Service Oriented Architectures and specifically SCA. Because the application developer has development platform and programming language knowledge they can build those portions of the application that might otherwise be difficult to define using the higher level tools. Using Enterprise Java Beans, Java, COBOL or other implementation technologies, the Application Developer develops application function and



The next three slides provide an overview of the primary features and associated visual tools found in WebSphere Integration Developer. The remainder of this presentation will cover some of the general features, but others are discussed in the module specific to the particular feature.

The following features are listed on this slide:

Business Object Editor: The business object editor is used to build and edit Business Objects. This editor is described in the Business Object presentation.

Interface Editor: The interface editor is used to build WSDL port-type interfaces which are used to define SCA service components. This editor will be described later in this presentation.

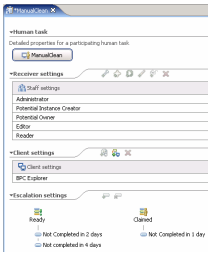
Assembly Editor: The assembly editor is the primary editor for building and assembling SCA applications. This editor is described in the SCA presentation.

Business Process Editor: The business process editor allows developers to visually create and manipulate business processes. This editor will be discussed in the Business Process Choreography presentation.

Features Overview: Visual Tools (cont.)

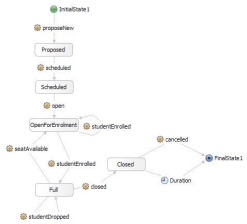
Defining human tasks

Human Task Editor

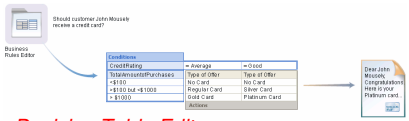


Creating a business state machine

Business State Machine Editor

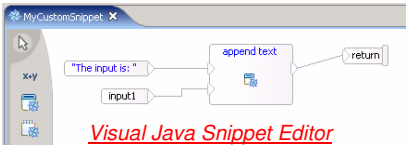


Defining business rules



Decision Table Editor
Ruleset Editor
Rule Group Editor

Creating custom visual snippets



Visual Java Snippet Editor

© 2006 IBM Corporation

The following features are listed on this slide:

Human Task Editor: The Human Task editor allows developers to visually compose services that interact with human participants. This editor is described in the Human Task Manager presentation.

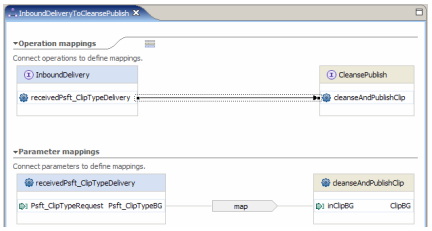
Business State Machine Editor: Enables developers to visually build business processes that are defined using a state machine model. This editor is described in the Business State Machine presentation.

Business rules editors: There are a number of editors associated with building business rules, including the Decision Table Editor, Ruleset Editor, and Rule Group editor.

Visual Java Snippet Editor: This editor is used to compose custom snippets visually. This editor will be described later in this presentation.

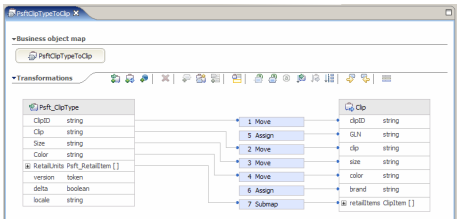
Features Overview: Visual Tools (cont.)

Defining Interface Maps



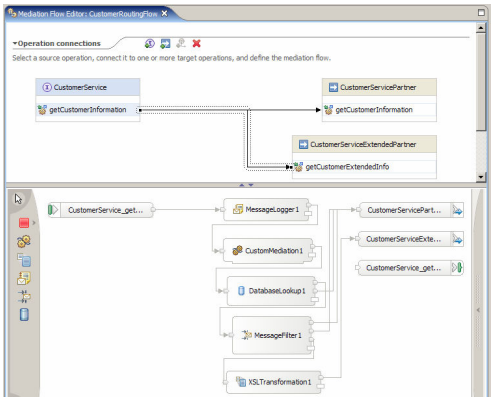
Interface Mapping Editor

Defining Business Object Maps



Business Object Mapping Editor

Defining ESB Mediation Flows



Mediation Flow Editor

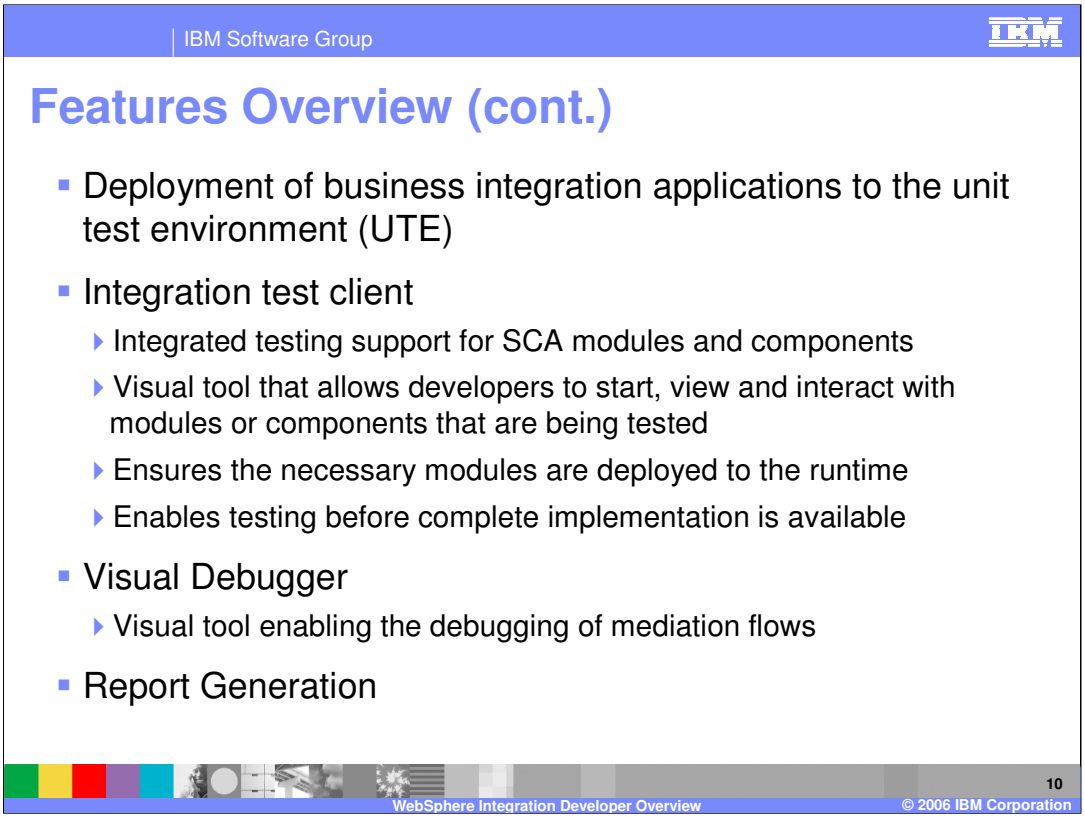
© 2006 IBM Corporation

The following features are listed on this slide:

Interface Mapping Editor: The Interface Mapping editor allows developers to map operations and parameters between semantically similar but syntactically different interfaces. This editor is described in the Interface and Data Mapping presentation.

Business Object Mapping Editor: The Business Object Mapping editor allows developers to define the mapping of data between business objects. This editor is described in the Interface and Data Mapping presentation.

Mediation Flow Editor: The Mediation Flow Editor enables developers to define the logic used in Enterprise Service Bus mediation flows. This editor is described in the Enterprise Service Bus presentation.



Listed here are several other key WebSphere Integration Developer features.

Like other IBM integrated development environments such as IBM Rational Application Developer, WebSphere Integration Developer provides an integrated unit test environment for deploying and testing your business applications during the development process. As part of the installation process for WebSphere Integration Developer you will have the option to silently install the WebSphere Process Server and the WebSphere Enterprise Service Bus unit test environments. If however you would like to have WebSphere Integration Developer use a stand-alone WebSphere Process Server or WebSphere Enterprise Service Bus test server that you have already configured, you can configure the tools to use that test environment during your development process.

A valuable testing option available for developers is the integration test client. This feature is offered for unit testing SCA modules and components and provides visual tools that allow developers to start, view and interact with resources being tested. The integration test client takes care of deploying the necessary modules to the runtime, and even allows testing before all components in the application are fully implemented through the use of emulators.

For the debugging of mediation flows used in the Enterprise Service Bus, there is a visual debugging tool. It enables the developer to step through a mediation flow and examine or modify the service message object that represents the message passing through the flow.

The report generation feature allows you to create a summary of the resources in your project by right clicking on the appropriate project in the Business Integration view and selecting the “Generate Documentation” context menu option. The report generation process lets you customize the types of resources in your module or library that are documented in the final report. The report provides information on resources such as Business Objects, Interfaces, Human Tasks, and Business Processes that are included in

Getting Started …

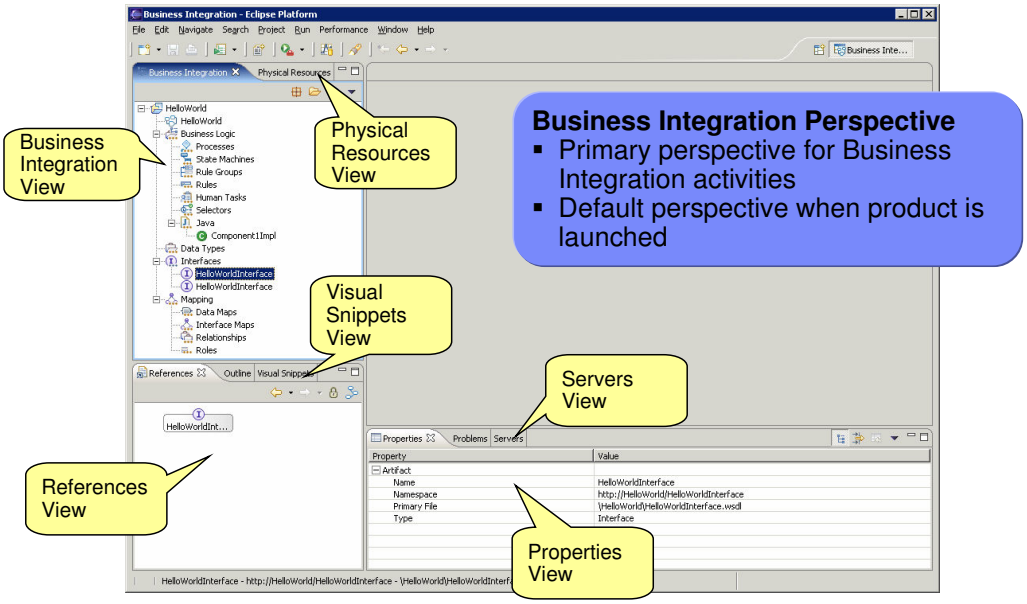
- Welcome screen provides links to
  - Product Overview, Cheat Sheets, Tutorials, Samples, What's New, Migration information, Web Resources
- Progressive Disclosure
  - Hides product features based on developer role
  - Features are enabled when first accessed or through Preferences
  - Product features are only loaded when enabled

© 2006 IBM Corporation

One of the first things you will notice if you have not used a product based upon Eclipse 3.0, is the welcome screen that is shown when you start a new workspace. From this screen you can access information about the product overview, cheat sheets, tutorials, samples, new features, migration information, and web resources. If you close the Welcome screen, you can always open it again by selecting Help > Welcome from the menu bar.

There is another new feature added in Eclipse 3.0 that is important to be aware of. This feature is referred to as progressive disclosure, though you may also hear it associated with the term "Capabilities". Progressive disclosure is a way to hide certain product features based upon the developers role. For example, if a user is not working on Web services development, it is possible to hide the associated wizards and tools by disabling the Web services development capabilities. Capabilities that are not enabled can be enabled either when the feature is first enabled, and you can find hidden features by selecting the "Show all wizards" check box, or by enabling the appropriate capability through the Preferences menu. Capabilities can be found by selecting Window > Preferences, expand Workbench, select Capabilities, and check the appropriate capabilities that are needed. Note that capabilities are associated with a given workspace. It is important to be aware of progressive disclosure (capabilities) because it might be necessary for you to turn on certain capabilities in order to make sure that the features you typically use during your development activities are visible to you.

Business Integration Perspective



© 2006 IBM Corporation

The primary perspective in WebSphere Integration Developer is the Business Integration perspective. The majority of your business integration work is done from this perspective and it is the default perspective when the product is launched. The screen capture on this slide shows the perspective and highlights several of the important views. Each of these views will be discussed later in this presentation.

Section

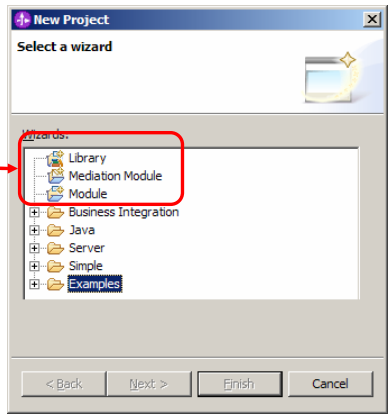
Concepts

© 2006 IBM Corporation

This section will provide a description of some important WebSphere Integration Developer concepts.

Integration Developer Concepts: Projects

- Business Integration project types
  - Library
  - Module
  - Mediation Module
- Module
  - Contains business integration logic
- Mediation Module
  - Contains service request mediation logic
- Library
  - Contains artifacts shared by multiple modules

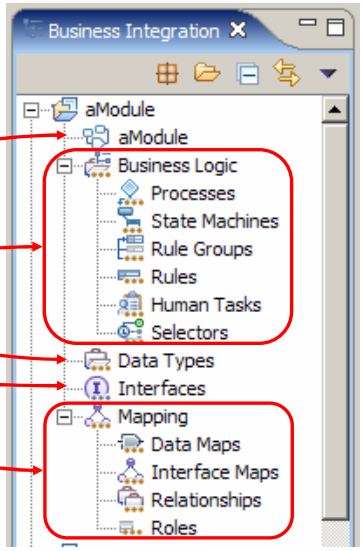


© 2006 IBM Corporation

WebSphere Integration Developer provides three types of business integration projects, namely the Library, the Module and the Mediation Module. The module is the project that contains the business logic that defines the business integration application. The mediation module is the project that contains the mediation flow logic used to loosely couple service requestors with service providers. The library project contains the definition of artifacts that are to be shared by multiple modules or mediation modules. The following slides will look at each of these more closely.

Module Project

- Business Integration project type for developing SCA based applications
- Contains the:
  - SCA wiring diagram
  - Various forms of business logic
  - Business Object definitions
  - Interface definitions
  - Various mapping artifacts

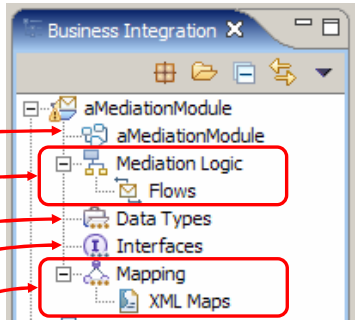


© 2006 IBM Corporation

A module is a business integration project type for developing SCA based applications and is where the business logic is defined. At the highest level, the business logic for the module is defined by the SCA wiring diagram which defines how the SCA components relate to each other and what imports and exports exist for interaction with services defined outside of this module. SCA components contain business logic which is defined using various high level constructs such as Business Processes, Business State Machines, Business Rules and Rule Groups, Human Tasks and Selectors. The business object definitions define the data types that are used by the business logic within the module. Likewise, the interface definitions describe the interfaces supported and used by the SCA components within the module. The mapping artifacts include interface and data maps and relationships and roles.

Mediation Module Project

- Business Integration project type for developing mediations between service requestors and service providers
- Contains the:
  - SCA wiring diagram
  - Mediation logic (mediation flows)
  - Business Object definitions
  - Interface definitions
  - Mapping definitions (XML maps)

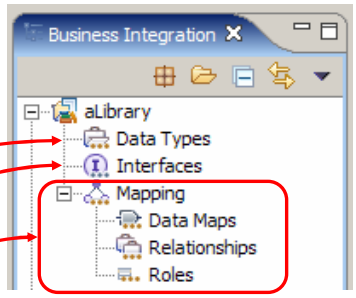


© 2006 IBM Corporation

A mediation module is a business integration project type for developing SCA based mediations that provide a loose coupling between service requestors and service providers. It is where the mediation logic is defined. At the highest level, the definition of the mediation starts in the SCA wiring diagram which defines how the mediation will be called by the service requestor and how it will call out to service providers and make use of custom mediation logic. The mediation logic itself is defined in the Mediation Flow component. The business object definitions define the data types that are used by the mediation. Likewise, the interface definitions describe the interfaces supported and used by the components of the mediation module. The mapping artifacts used with mediation flows are XML maps which describe mapping between XML based representations of the service message flowing through the mediation.

Library Project

- Business Integration project type for artifacts shared between multiple modules and mediation modules
- Contains:
  - Business Object definitions
  - Interface definitions
  - Various mapping artifacts
- Can be added to the dependency list for a module or mediation module from the dependency editor

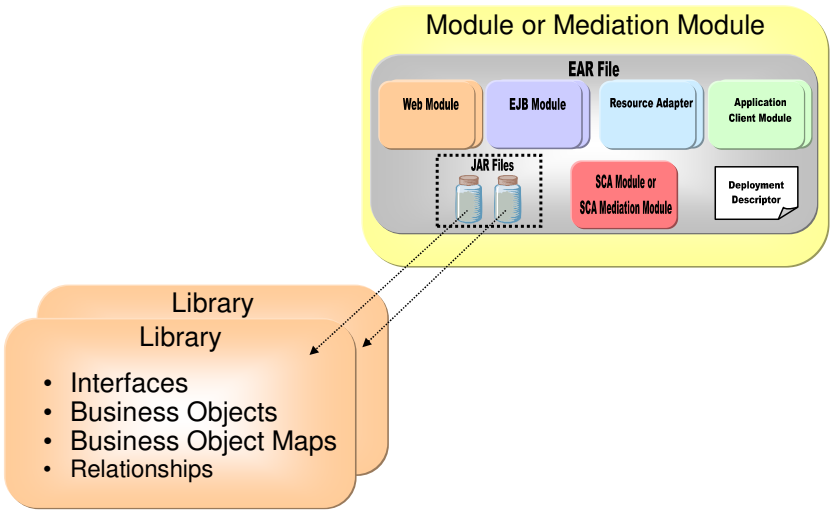


© 2006 IBM Corporation

A Library project is a business integration project type used for defining artifacts that are shared between multiple modules and mediation modules. An important distinction to make concerning the library is that it does not contain any definitions that define business logic. Library projects only contain business objects, interfaces and some mapping artifacts. Nothing that is represented as an SCA component can be included in a Library.

If a module or mediation module is dependent upon a particular Library project, that project must be added to the dependency list for the appropriate module or mediation module using the dependency editor, which is discussed later in this presentation.

Deploying Projects



© 2006 IBM Corporation

WebSphere Integration Developer packages projects so that they can be deployed to a WebSphere Process Server or WebSphere Enterprise Service Bus. The module and the mediation module are the projects which are deployable units whereas the library project is not a deployable unit. Rather, the library is deployed with the modules and mediation modules which are dependent upon the artifacts in the library.

As a deployable unit, the module or mediation module is constructed into a J2EE EAR file and is therefore equivalent in scope to a J2EE application which can be installed into a server. For each module or mediation module to be deployed, WebSphere Integration Developer generates several different J2EE projects which contain generated code and artifacts needed in support of the SCA architecture. These J2EE modules are packaged in JAR files and are include in the deployed EAR file. The artifacts in any dependent libraries are also packaged into JAR files and included in the deployed EAR.

Section

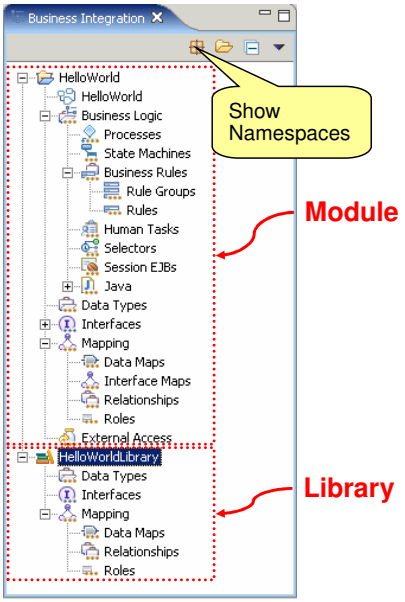
Integration Developer basics

© 2006 IBM Corporation

This section will provide an introduction to the basic WebSphere Integration Developer development environment.

Business Integration View

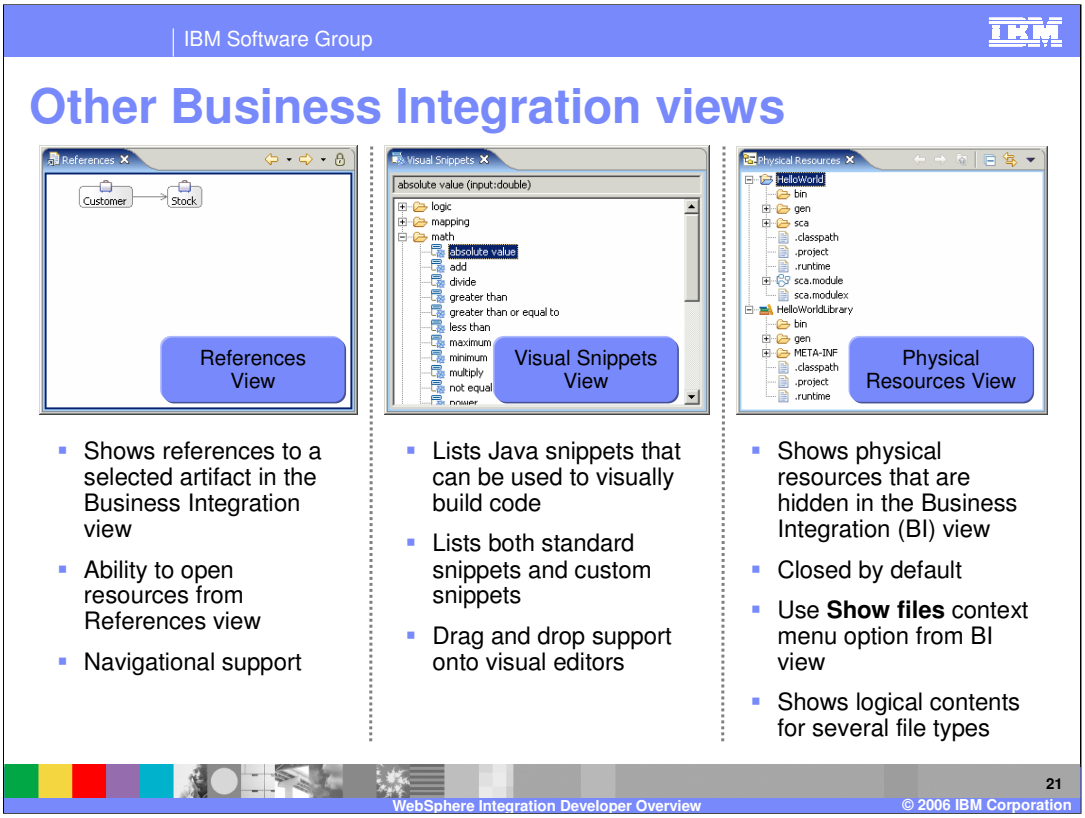
- Primary view for managing and viewing business integration resources
- Provides logical grouping of resources
  - Artifacts not essential for business integration development are not visible
- Logical resources do not necessarily have a one-to-one relationship with a physical resource



© 2006 IBM Corporation

The primary view in the business integration perspective is the business integration view. This view is used to manage and view all business integration resources. The resources shown in this view provide a logical grouping of resources, and hides artifacts that are not essential for business integration development. It is important to note that this view only shows a logical representation of the resources in the workspace that are related to business integration work, and there is not a one-to-one relationship to physical resources on the file system.

From the screen capture shown on this slide, notice that both Module and Library projects are visible from the Business Integration view. A mediation module is not shown here, but they are also visible in the business integration view. Also note that the types of resources that can be added to a Library project is a subset of those that can exist in a Module project as explained in the previous section.



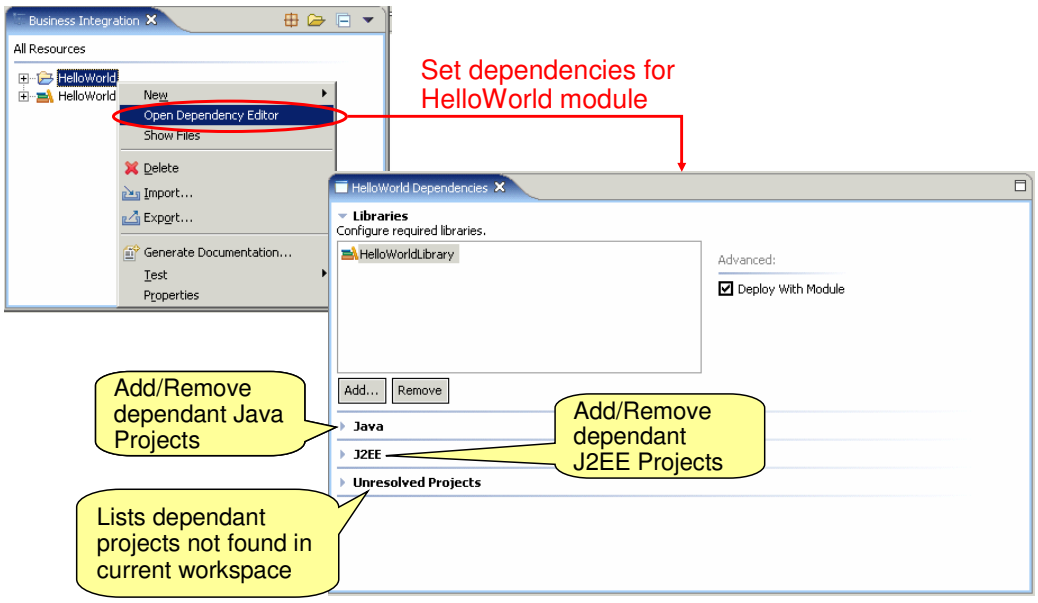
Shown here are several other views to be aware of when working in the Business Integration perspective.

The References view is used in association with the Business Integration (BI) view. The contents of the references view are based on the artifact that is selected in the BI view. For example, if you have selected a business object called Customer underneath the Data Type category in the BI view, the Resources view will show any references the Customer business object has to other business objects in your module or in dependent libraries. From the resources view, you can open the resource and navigate to other artifacts displayed in the view.

The Visual Snippets view lists Java snippets that can be used to visually build code. From this view, both standard snippets that come with the product and custom snippets are shown, and are available for drag and drop support onto various visual code editors.

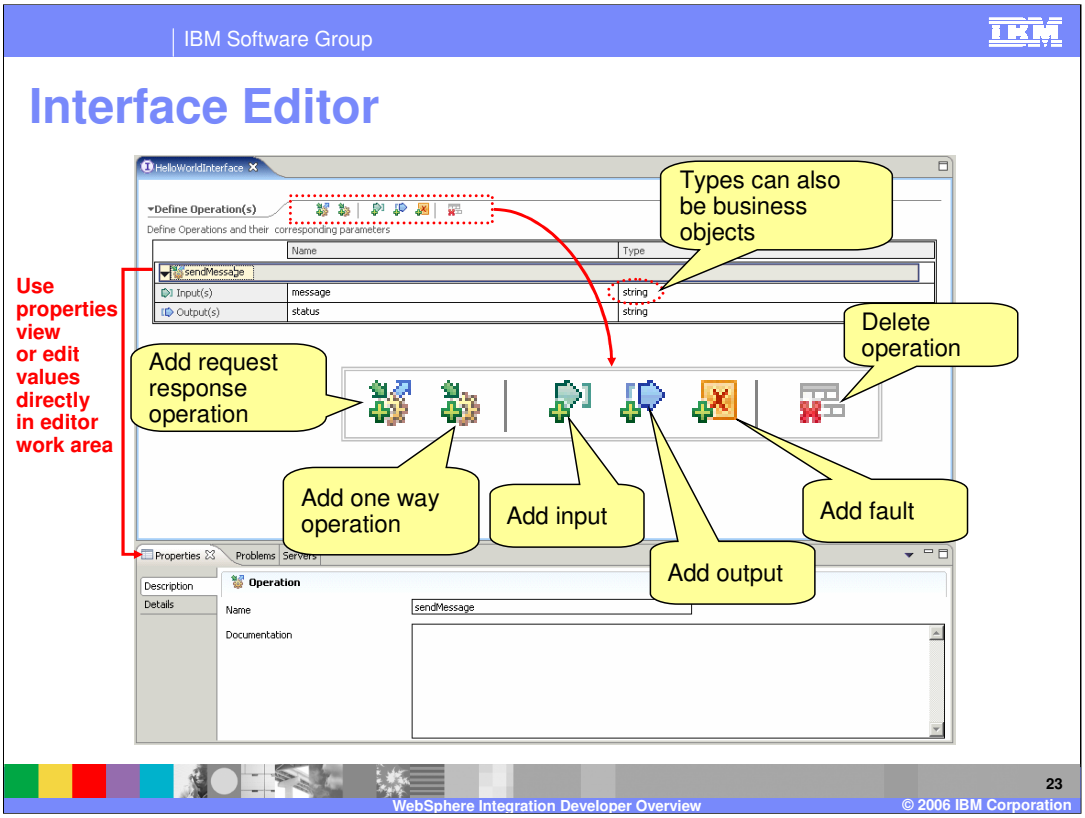
The Physical Resources view by default is not open in the Business Integration perspective. However, this view can be a useful view to have open if you want a view that shows the physical resources that are hidden in the BI view. For example, from this view you can view the individual SCA resources that make up the elements in your assembly diagram. This can be helpful when debugging an application, or to learn more about the artifacts that are created while building an SCA application. To open the Physical Resources view, select Window > Show View > Physical Resources from the menu. Alternatively, you can right click on an item in the BI view and select Show files from the context menu option. This will open the physical resources view and highlight the relevant

Dependency Editor



© 2006 IBM Corporation

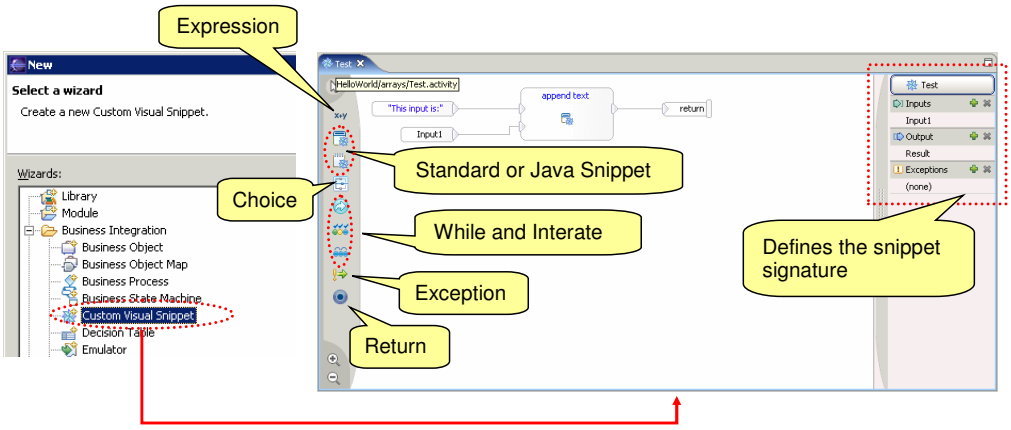
The dependency editor is used to identify projects that a module is dependant upon. For example, a module project might have a dependency upon a library project if the module references resources such as business objects or interfaces that are included in that library. You can set dependencies on module, mediation module or library projects. One difference is that a library project cannot be dependent upon a J2EE project, only other libraries and Java projects. With each dependency that is added to the list, you have the option to deploy that project with the module. For J2EE project types you also have the option to include the project on the build path.



The interface editor is used to view and construct WSDL port type interfaces that you will associate with service components in your SCA module. Like other editors in WebSphere Integration Developer, the interface editor is used in conjunction with the properties view to edit and view properties associated with the interface definition. Interface definitions created with the interface editor can have one or more operations. When the file is saved it is saved in a <INTERFACE_NAME>.wsdl file.

The first thing that must be added to an interface definition is one or more operations. This is done from the toolbar at the top of the interface editor that allows a developer to add either a request/response operation or a one way operation. There are also options on the tool bar that provide the ability to associate inputs, outputs, and faults with the selected operation. Once inputs or outputs are added to an operation, you can use the table in the interface editor or the properties view to set the data type for the parameter. You can click anywhere in the interface editor to view or update basic properties associated with the interface, such as target namespace, in the Properties view.

Visual Snippet Editor

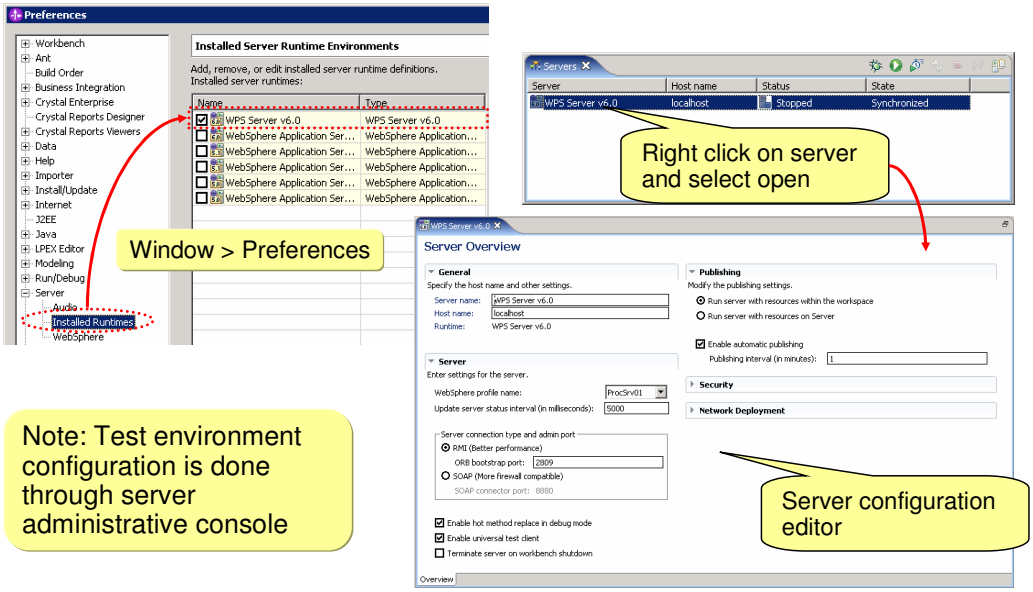


© 2006 IBM Corporation

The visual snippet editor is used to help developers build custom Java code in a visual manner. To create a new custom visual snippet, select File > New > Other and find 'Custom Visual Snippet' under the Business Integration category. When you create a new custom snippet, you have the option to add this snippet to the list shown in the Visual Snippets view. In this way, you can reuse a visual snippet in various visual code editors.

The visual snippet editor is divided into several areas. At the left of the editor is a palette that is used to visually add coding constructs, such as expressions, iterations, exceptions, and choices, to the snippet being built. These constructs are dropped onto the canvas area found in the center of the editor. At the right of the snippet editor is an area called the tray, which lists inputs, output, and exceptions for this particular Java snippet. The snippet editor is also used in conjunction with the Properties view to further develop the snippet.

Unit Test Environment: Configuration

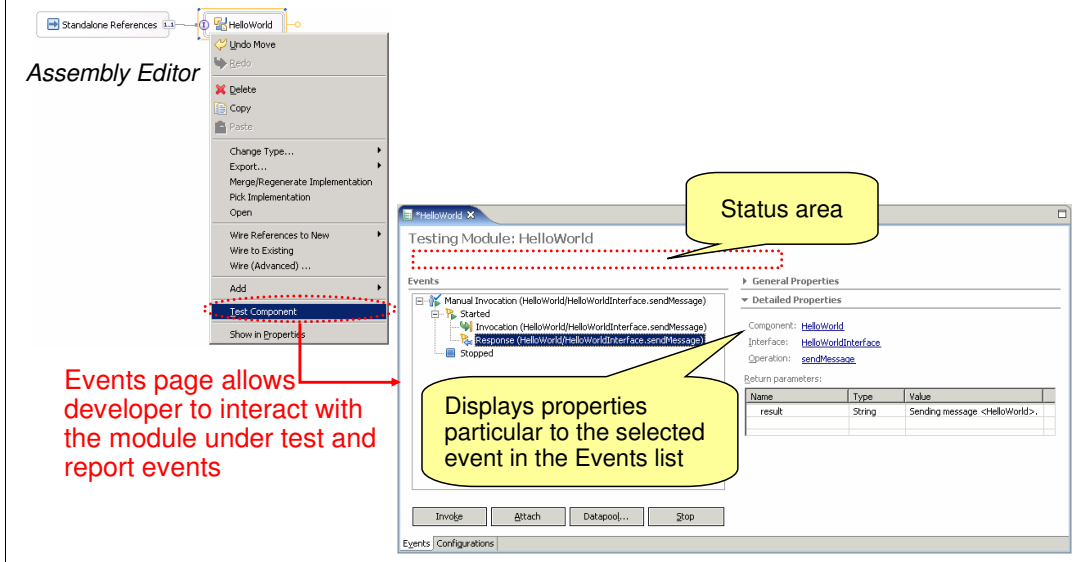


© 2006 IBM Corporation

Typically when you build an application in WebSphere Integration Developer you will at some point want to deploy your application to a WebSphere Process Server or WebSphere Enterprise Service Bus unit test environment. Your server could be the default runtime test environment that you optionally installed as part of the WebSphere Integration Developer installation, or it could be a stand-alone server that you have installed separately and are using for testing purposes.

Your test server is accessed from the Servers view in the Business Integration perspective. It is from this view that you can manage your server and perform tasks such as start and stop the server or add/remove projects from the server. Each server that appears in the servers view has a basic configuration with which it is associated. This configuration is used by WebSphere Integration Developer to hold information that indicates how the tools environment connects to the test server and how applications are published to that test server. However, overall test environment configuration is done through the WebSphere Process Server or WebSphere Enterprise Service Bus administrative console. To launch the administrative console, right click on the server and select 'Run administrative console' from the context menu.

Integration Test Client



© 2006 IBM Corporation

The integration test client is a feature in WebSphere Integration Developer that allows developers to unit test SCA modules and components with a convenient user interface. The primary page in the integration test client interface is the events page. It is from this page that developers interact with the component under test to provide inputs and view the resulting output from a test run. The Configurations page allows developers to customize the test. SCA components that are not implemented can be emulated using the integration test client, thus allowing incremental development and test.

Section

Summary and references

© 2006 IBM Corporation

This section will provide a summary and some references for this overview presentation of WebSphere Integration Developer.

Summary

- WebSphere Integration Developer V6.0.1 is an
  - Authoring tool for building reusable SOA assets targeted for WebSphere Process Server V6.0.1 or WebSphere Enterprise Service Bus V6.0.1
  - Integrated development environment to help developers build composite business applications
- WebSphere Integration Developer V6.0.1 includes an integrated unit test environment
  - WebSphere Process Server V6.0.1
  - WebSphere Enterprise Service Bus V6.0.1
  - Allows testing and debugging of business applications

© 2006 IBM Corporation

WebSphere Integration Developer V6.0.1 is the authoring tool for building reusable SOA assets targeted for WebSphere Process Server V6.0.1 or WebSphere Enterprise Service Bus V6.0.1. Specifically, the tools in WebSphere Integration Developer are designed for building SCA based applications. Included with WebSphere Integration Developer is a WebSphere Process Server and WebSphere Enterprise Service Bus integrated unit test environment that allows developers to easily test and debug business applications built in WebSphere Integration Developer.

References

- Eclipse
  - http://www.eclipse.org/
- Eclipse 3.0 – New Features
  - http://download.eclipse.org/downloads/drops/R-3.0-200406251208/eclipse-news-R3.html
- IBM Rational Application Developer V6 Overview
  - http://www.ibm.com/software/info/education/assistant

© 2006 IBM Corporation

Template Revision: 11/22/2005 12:10 PM

Trademarks, Copyrights, and Disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM	CICS	IMS	MQSeries	Tivoli
IBM(logo)	Cloudscape	Informix	OS/390	WebSphere
e(logo)business	DB2	iSeries	OS/400	xSeries
AIX	DB2 Universal Database	Lotus	pSeries	zSeries

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

Other company, product and 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 and/or changes in the product(s) and/or program(s) 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.

Information is provided "AS IS" without warranty of any kind. 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 (e.g., 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 2005,2006. 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.