Importing Web Service in CAF and Developing Web Dynpro Application

Applies to:

SAP Net Weaver CE 7.1.1 EHP1 (SP0, SP1, SP2 sand SP3).

Summary

This article explains importing web service in Composite Application Framework (CAF) and developing a Web Dynpro application using Enterprise Java Bean Model.

Author: Sunil Kumar S S

Company: SAP Labs India Pvt. Ltd.

Created on: 04 May 2009

Author Bio

Sunil Kumar S S works as a Quality Engineer within the Web Dynpro Tools team. He has been associated with SAP Labs since July 2007. His areas of expertise include Web Dynpro Java, Data Dictionary, Web Services, CAF and NWDI.

Table of Contents

Prerequisite

- SAP NetWeaver Developer Studio 7.1

Introduction

External Services

You can use external services to communicate with external systems and applications independent of implementation language, technology, or platform. This ensures that the pre-existing functions of these systems and applications can be reused in CAF.

In CAF, you call back-end functions with the following external services:

- Remote function calls (RFCs)
- Web services

You cannot modify external services. You only define dependencies from the application to the external service, and then expose or consume the external service from your application service. External services are always stateless, meaning a session opens and closes within the call.

Web Services

You can use Web services to enable systems to exchange data and start actions of any type. The WSDL (Web Service Description Language) files are imported and retrieved from a directory or UDDI server. You can find a Web service definition either by specifying the URL of its WSDL file, or by browsing the Services Registry.

Remote Function Calls

You can use the RFCs to communicate with ABAP-based SAP systems. RFCs are interfaces that enable external communication with the function modules of SAP systems.

You can import the metadata (including methods and input and output parameters) of RFCs into CAF and subsequently make them available to other CAF services.

More information: Importing RFC Modules

Enterprise Java Bean (EJB) Model

The Web Dynpro model importer supports the Common Model Interface (CMI). CMI is a layer of indirection between a framework like Web Dynpro Java, which uses a model, and a framework which implements a model, in this case the Enterprise Java Bean model. Both the using and the implementing framework are independent. Nevertheless they can exchange data via data binding, for instance.

Regarding the model, the layer grants a framework like Web Dynpro access to the business logic. It exposes business data and metadata via the CMI to the framework that sits on top of the CMI. For typed models, the structure is at least partially known at design time, and design time meta data is available. You can bind context nodes to model classes at design time and program in a type safe manner against these models. There is no urgent need for CMI metadata at runtime. For the generic models it is typical that the structure is unknown or at least not stable at design time. The CMI metadata must be available at runtime. There is only a dynamic binding between context node and model class at runtime, and no generation step is needed. The Enterprise Java Bean model allows mixtures between these two model types.

Modeling External Services

Creating Composite Application Development Component

Prerequisites

1. Launch the SAP NetWeaver Developer Studio.
2. Configure Application Server (AS) Java: Windows -> Preference -> SAP AS JAVA.
3. Choose Window -> Open Perspective -> Other.
4. Select Composite Application from the list.
5. Choose OK.

Procedure

1. Choose File -> New ->Project…
2. Select Development Infrastructure -> Development Component and choose Next.
3. Select Composite Application and click next
4. Select the software component for the new DC to be created in and Click next.

   The default software component is Local Development -> My Components.

5. Fill the required data as follows:

   Vendor **:** demo.sap.com

   Name **:** Enter the name of the DC.

   Caption **:** Enter a single line caption.

   Language **:** Choose the language. (***Default value:*** American English)

   Domain **:** Choose the domain. (***Default value*** **:** Basis)

   Support Component **:** BC

6. Choose Next, then choose Finish. The application you have created is added to the Composite Application Explorer.

Importing Web Services

When designing your composite applications with the Composite Application Framework (CAF), you can easily consume existing enterprise services, and map them to customizable application services that fit your particular needs. In addition, CAF allows you to edit the services in an entirely model-driven manner, which ensures flexibility, reusability, ease of development, and increased development productivity.You can browse the enterprise services that are offered in the Enterprise Service (ES) Workplace of the SAP Developer Network.

Procedure

1. In the Composite Applications Explorer of SAP NetWeaver Developer Studio, choose the CAF application

   In which you want to import a Web service.

2. Click the external node with the secondary mouse button and choose Import Web Service.

3. To import a WSDL definition, select Remote Location / File System – imports a Web service from your local file system or from another location.

4. In the URL field, enter the URL address of a Web service given below and Choose Next.

http://pwdf6237.wdf.sap.corp:50000/ProjectDataServiceBeanService/ProjectDataServiceBean?wsdl

5. Choose “Create New” to create a service group and Click Finish.

The imported Web service appears under the external package node in Composite Application Explorer.

You can now create mappings for the Web service.

Mapping Service Operations

Once you have imported an external service in your composite application, you need to map the external service operations to application service operations. This way you use external services without implementing code. When the external service you have imported has a huge number of operations and attributes you can create automatic identical mappings to an application service.

The CAF application uses an application service operation to send an input message. The input message attributes are mapped to the remote service operation input attributes (input mapping). The remote service processes the input attributes and generates output attributes. The output message attributes are mapped to the application service operation output attributes (output mapping).

Creating Automatic Mappings for Application Services

1. Click on the external services you have imported with the secondary mouse button and choose Default mappings.

2. Select all the attributes of the external services to be automatically mapped.

3. Choose the target application service for your mapping. Select the options, *Create methods in new Application Service* and Click OK.

Generated EJB Implementation Class for Application service

1. Double Click on the Application service that you have created under modeled node.

2. Open the Implementation tab and click on the EJB Implementation Class hyperlink.

3. Save, generate, build and deploy your application.

Create a Web Dynpro Application consuming Application service created by CAF

Creating a Web Dynpro Project

To manage local development objects of an application, you first need an appropriate project in the Developer Studio. Once you have set up the initial project framework, you can create the specific components of the Web application, define them, and implement the necessary source code. There is a wizard which generates an initial structure for your new Web Dynpro project. The Web Dynpro Explorer provides a logical view of the structure of a Web Dynpro project then. With this project view, you can access all Web Dynpro units. In addition, each Web Dynpro project provides access to local dictionaries, with the help of which you can create user-defined dictionary types. Additionally, you can add required projects resources, such as separate Java help classes, to the Web Dynpro project through the resources node.

Procedure

1. Select Development Infrastructure -> Development Component and choose Next.

2. Select Web Dynpro and Click next.

3. Select the software component for the new DC to be created in and Click next.

   The default software component is Local Development -> My Components.

4. In the window that appears, enter a name for the project and specify the settings for Project contents and Project language.
5. Finishing the wizard you are asked to open the perspective, based on the type of the project to be created is associated with the Web Dynpro perspective.

Creating the Application Unit

The project structure you have created does not yet include all the elements that enable you to define the specific functions of the whole Web Dynpro application. These additional elements are encapsulated in a Web Dynpro component that can contain any number of windows and views for visual representation and their corresponding controllers. Moreover, you also need an object that can be used to address and display the Web Dynpro component in the Web browser. This is why you create the logical unit application as an additional object. As an independent program unit it connects to an URL that can be accessed from the user via a window within the Web Dynpro component.

Procedure

1. Expand the Web Dynpro node in the structure of the Demo project.
2. To start the relevant wizard, open the context menu of the Applications node, and choose Create Application.
3. Enter a name for the application unit, such as unit *DemoApp*, and specify the package name (example: com.sap.demoapp) in which the generated Java classes will be stored. Choose Next.
4. Assign a Web Dynpro component to which the application is linked. Since a Web Dynpro component has not yet been created, the wizard creates a component at this point. Accept *Create a new component* and choose Next again.
5. In the next screen you specify some general values for the component. Enter the name *DemoComp* for your Web Dynpro component and specify the package name *com.sap.demoapp.comp*. Accept the other suggested settings and choose Finish.

Result

The application wizard performs several generation routines. Once it has finished, it adds the new application object to the structure. Additionally, it inserts a substructure for the new Web Dynpro component. Furthermore the initial *DemoCompView* is automatically created for you; you find this view under the Views node in your Web Dynpro project.

Furthermore, a window unit with name <myproject>Window, in this example with name *DemoWindow*, has been generated; including the needed plug for application starting purposes.

Importing Enterprise Java Bean (EJB) Model

The Web Dynpro model importer supports the Common Model Interface (CMI). CMI is a layer of indirection between a framework like Web Dynpro Java, which uses a model, and a framework which implements a model, in this case the Enterprise Java Bean model.

Procedure

1. To start the relevant wizard, open the context menu of the Models node, and choose *Create Model*.

   In the model import wizard, choose the Enterprise Java Bean Model option and Click next.

2. Enter a name for the model, such as unit *Demomodel*, and specify the package name (example:

   com.sap.demomodel).

3. Click on Add Dependency, select “caf/runtime/ear” and “<cafproject>/ear”, Click Next, Finish.
4. Select “<cafproject>/ear” and click Next.
5. Select the Enterprise Java Bean Class and Click Next.
6. If proposed model class name clashes with another name then rename the model class Else Click Next.
7. If there are unresolved relations then resolve them by selecting a valid Target Model Class and Click Next.
8. Click Finish. Model is created and visible in the WD explorer.

Creating a User Interface to access data from web service

Apply Service Controller Template

Creates a controller that wraps a service. This service is either a Remote Function Call or a Web service. The wizard then creates the required contexts and an optional method for triggering the call.

Procedure

1. Add the created model to the list of Used Models node under Component with the secondary mouse

   button, choose the created model and click Ok.

2. Select the Component Controller in Web dynpro explorer with the secondary mouse button, choose

   Template -> Apply

3. Select Service Controller and Click Next.

4. Select “*Request_<ApplicationServiceName><ServiceLocal>_getAllEmployees”* Model Class and Click next.

5. Select all the nodes, attributes under
“*Request_<ApplicationServiceName><ServiceLocal>_getAllEmployees”* and Click next.

6. Enable Generate method for model execution and Click on Finish.

View - Context Mapping

1. Double click om the view in Web Dynpro explorer.

2. Go to Context tab, Select Context and with the secondary mouse button, Choose New > Node > Mapping.

3. Click Next, Choose Component Controller in popup Required Controllers and Click OK.

4. Select “*Request_<ApplicationServiceName ><ServiceLocal>_ getAllEmployees”* and click Next.

5. Select all the nodes, attributes under
“Request_<ApplicationServiceName><ServiceLocal>_getAllEmployees” and Click Finish.

Apply Action Button Template

The wizard for the Action Button template allows you to easily create a button, an action, an event handler, and the code for calling a method and triggering an event.

Procedure

1. Select the View in Web dynpro explorer with the secondary mouse button, choose Template -> Apply
2. Choose the option *ActionButton*. Enter a name for the template or use the default.
3. Choose *Next*. In the next window, you use the default for action, and event. For the label, Enter *GetAllEmployees*.
4. Choose *Next* to proceed to the wizard window for defining the methods and plugs.
5. Select the check box Call Method and for controller drop down field select the View Controller.

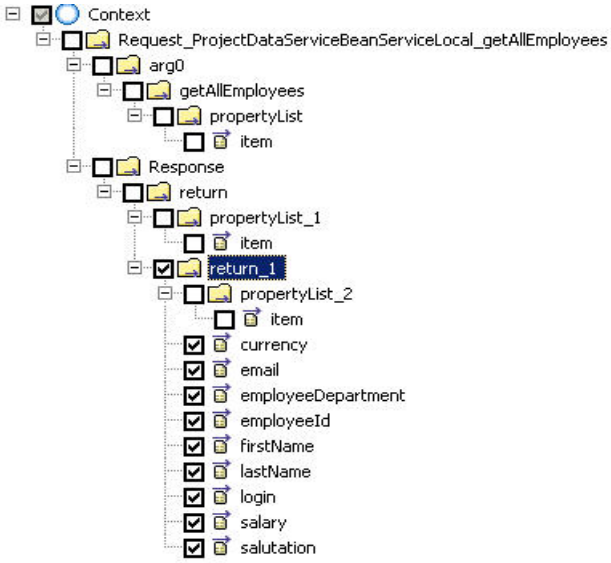
   Corresponding method is also filled in Method drop down field.
6. Close the wizard by choosing *Finish* and save your changes.

Apply Table Template

When you define this table layout, you can access the controller context data directly.

Procedure

1. Select the View in Web dynpro explorer with the secondary mouse button, choose Template -> Apply
2. Choose the option Form. Enter a name for the layout template or use the default.
3. Choose Next. You see a window in which you can choose the context attributes you require for the form.
4. Select the nodes and attributes displayed in the image below.



5. Close the wizard by choosing Finish and then choose Save All to save your changes.

Build and Deploy DCs

Build

1. On the web dynpro project with the secondary mouse button choose Development Component > Build.

2. Select all the DCs and Click OK. Make sure that Fore Build Check box is active.

Deploy

1. On the web dynpro project with the secondary mouse button choose Deploy.

2. Enter the J2ee Engine Username and password and click OK.

3. Click OK on Deploy wizard once deployment is successful.

Run Application

To deploy the logical Application unit which is also displayed in the Web Dynpro Explorer view, you select Deploy new Archive and Run. As you can see, you would also trigger the execution process for the WD application.

Output


GetAllEmployees

login	currency	salutation	employeeId	email	employeeDepartment	firstName	salary	lastName
springs	USD	Ms.	1	sally.spring@itelo.info	SALESOP	Sally	55,549	Spring
summers	USD	Ms.	2	susan.summer@itelo.info	MARKETING	Susan	76,239	Summer
fallf	USD	Mr.	3	franco.fall@itelo.info	MARKETING	Franco	85,962	Fall
winterw	USD	Mr.	4	walter.winter@itelo.info	MARKETING	Walter	68,597	Winter
hicksm	USD	Ms.	5	maria.hicks@itelo.info	GLOBALIT	Maria	49,876	Hicks

Related Content

For more information, visit the User Interface Technology

For more information, visit the Composite Application Framework

For more information, visit the Web Dynpro Java homepage.

Copyright