SAP NetWeaver
How-To Guide

How To... Call An Enterprise Service From JSF

Applicable Releases:

SAP NetWeaver Composition Environment 7.1

Topic Area:

User Productivity

Development and Composition

Capability:

User Interface Technology

Java

Version 1.0

October 2008

THE BEST-RUN BUSINESSES RUN SAP™

Document History

Document Version	Description
1.00	First official release of this guide

Typographic Conventions

Type Style	Description
Example Text	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Cross-references to other documentation
Example text	Emphasized words or phrases in body text, graphic titles, and table titles
`Example text`	File and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
`Example text`	User entry texts. These are words or characters that you enter in the system exactly as they appear in the documentation.
`<Example text>`	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
`EXAMPLE TEXT`	Keys on the keyboard, for example, `F2` or `ENTER`.

Icons

Icon	Description
⚠	Caution
💡	Note or Important
✿	Example
⬆	Recommendation or Tip

Table of Contents

1. Business Scenario

The following guide will explain you how to create a JSF application that invokes an Enterprise Service using Java EE annotation. It will also explain how to locate and test Enterprise Services by browsing through the “Enterprise Services Index” in the Enterprise Service-Oriented Architecture website.

For the purpose of this tutorial the Enterprise Service to be consumed is the *Read Employee Basic Data* operation, located in the *Manage Employee In* service interface from the *Business Partner Data Management* Process component. This Enterprise Service will allow you to search employee’s personal information by providing the employee ID number.

2. Background Information

Enterprise SOA has been defined by SAP as "an open architecture for adaptive business solutions". It allows you to encapsulate the business logic and expose it as Enterprise Services. Enterprise Services are highly-integrated Web Services combined with business semantics that can be located and invoked by other applications over a network. To learn more about the Enterprise SOA you can visit Enterprise Service-Oriented Architecture website

In addition, this example demonstrates the use of Java EE Annotations. Annotations are Java modifiers, similar to *public* and *private*, that simplify the application development process by allowing developers to specify within the Java class itself how the application component behaves in the container and requests for dependency injection. To learn more about Java EE Annotations you can visit Introduction to Java EE 5 Technology

3. Prerequisites

The following is a list of all you need for developing JSF applications and invoking Enterprise Services.

- AS Java 7.1 (CE 7.1 or NW 7.1)
- NWDS 7.1 (SP3 or higher with latest patch level).

  Note

  While this tutorial is geared towards to the SAP AS Java (the build/deploy steps of the guide), it wouldn’t be hard to replace the build/deploy portions with similar steps for any other Java EE 5 platform
- A registered user on the ES Workplace who can consume the Enterprise Services. To get your user for free, you can visit Enterprise SOA Registration Form

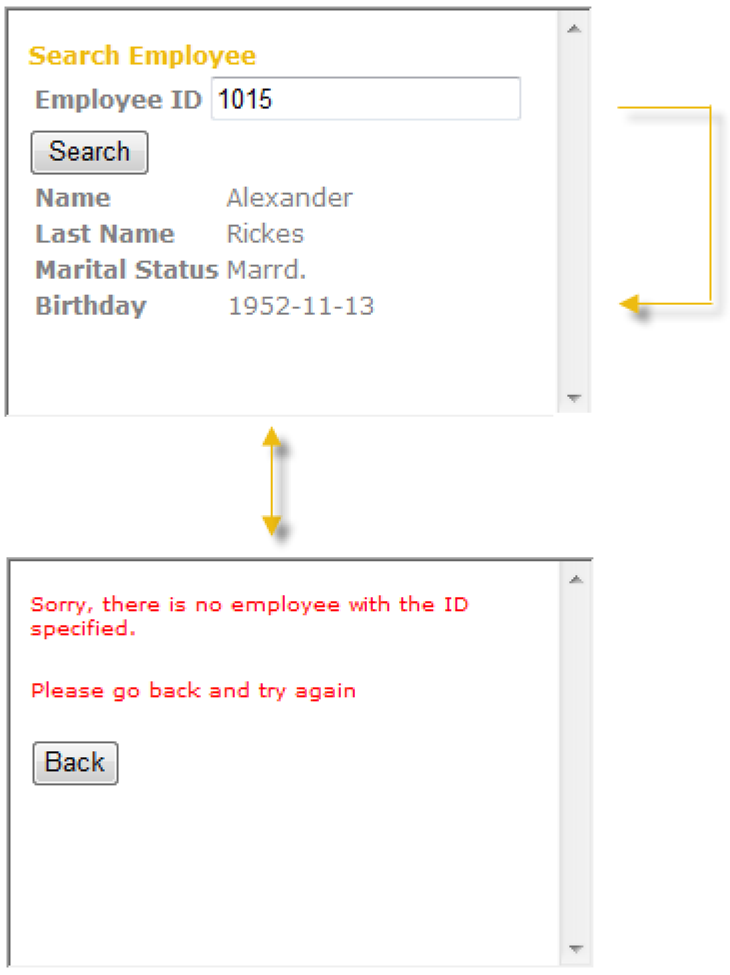
Knowledge

- You have a basic knowledge of Java Enterprise Edition
- You have acquired some basic experience with JSF applications, for example by working through the JSF tutorials (Create a Hello World Application using JavaServer Faces [Extern] and Create Your First JSF Application [Extern])

4. Step-by-Step Procedure

In the following sections, you will create a Web Module Development component and an Enterprise Application needed to deploy the web module. You will get to know how to locate an Enterprise Service in the Enterprise SOA website and how to invoke it from your JSF page.

This Web application will consist of two views. In the first view, the user should be able to enter the employee ID in the input text to find the employee information. If no information is found, an error view will be displayed. From the error view the user should be able to navigate to the first view using the Back button.



Search Employee

Employee ID 1015

Search

Name	Alexander
Last Name	Rickes
Marital Status	Marrd.
Birthday	1952-11-13

Sorry, there is no employee with the ID specified.

Please go back and try again

Back

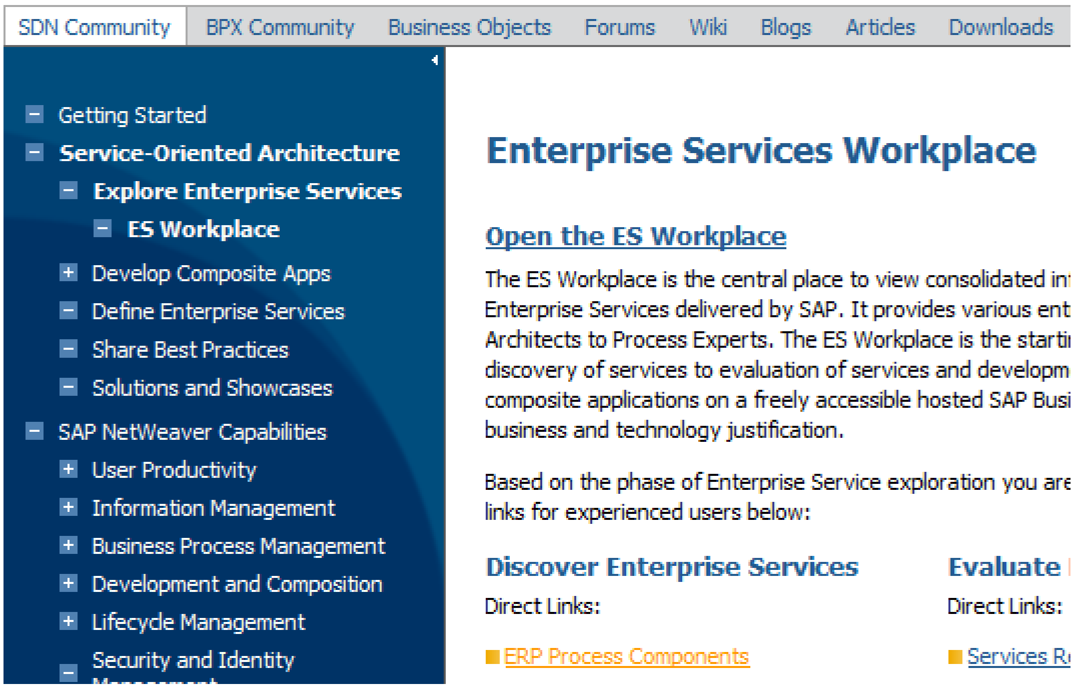
4.1 Tutorial Setup

1. Create a Web Module Development Component named **`esjsf/web`**.
2. Create an Enterprise Application Development Component named **`esjsf/ear`**.

4.2 Locate Enterprise Service

The fastest way to locate your Enterprise Service is to browse through the “Enterprise Services Index”. The navigation hierarchy is *Process Component* → *Service Interface(s)* → *Service Operation(s)*. For more details, check the ES Workplace Handbook to efficiently browse and testdrive Enterprise Services.

1. Go to the www.sdn.sap.com
2. In the left navigation menu select *Service-Oriented Architecture* → *Explore Enterprise Services* → *ES Workplace*
3. In the main page click in the *ERP Process Components* link to get the list of the process components found in the ERP system



SDN Community | BPX Community | Business Objects | Forums | Wiki | Blogs | Articles | Downloads

- Getting Started
- **Service-Oriented Architecture**
  - **Explore Enterprise Services**
    - **ES Workplace**
  - Develop Composite Apps
  - Define Enterprise Services
  - Share Best Practices
  - Solutions and Showcases
- SAP NetWeaver Capabilities
  - User Productivity
  - Information Management
  - Business Process Management
  - Development and Composition
  - Lifecycle Management
  - Security and Identity

Enterprise Services Workplace

Open the ES Workplace

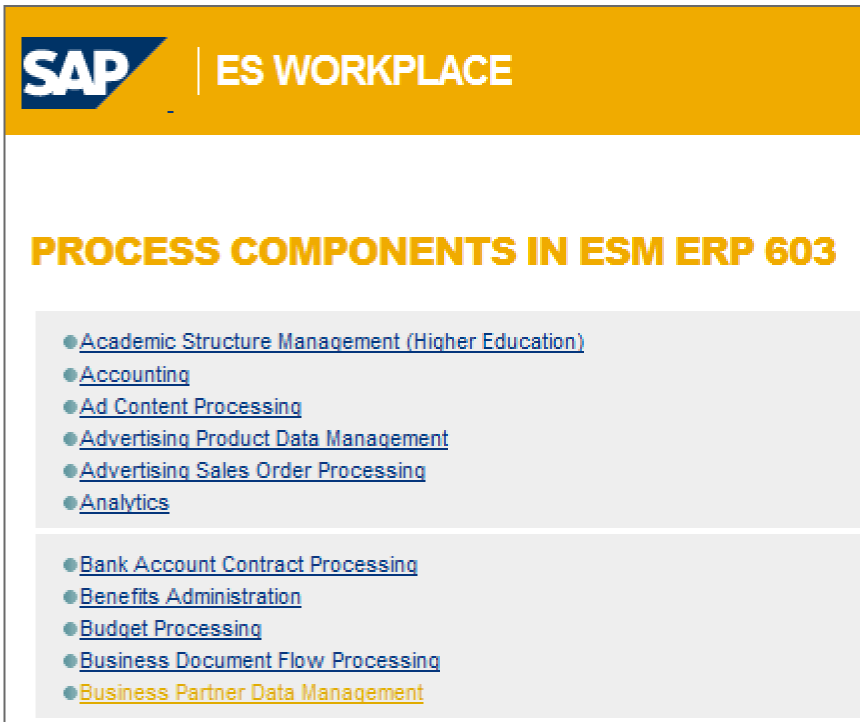
The ES Workplace is the central place to view consolidated in
Enterprise Services delivered by SAP. It provides various ent
Architects to Process Experts. The ES Workplace is the starti
discovery of services to evaluation of services and developm
composite applications on a freely accessible hosted SAP Busi
business and technology justification.

Based on the phase of Enterprise Service exploration you are
links for experienced users below:

Discover Enterprise Services
Direct Links:
■ ERP Process Components

Evaluate
Direct Links:
■ Services R

4. Select the *Business Partner Data Management* process component.



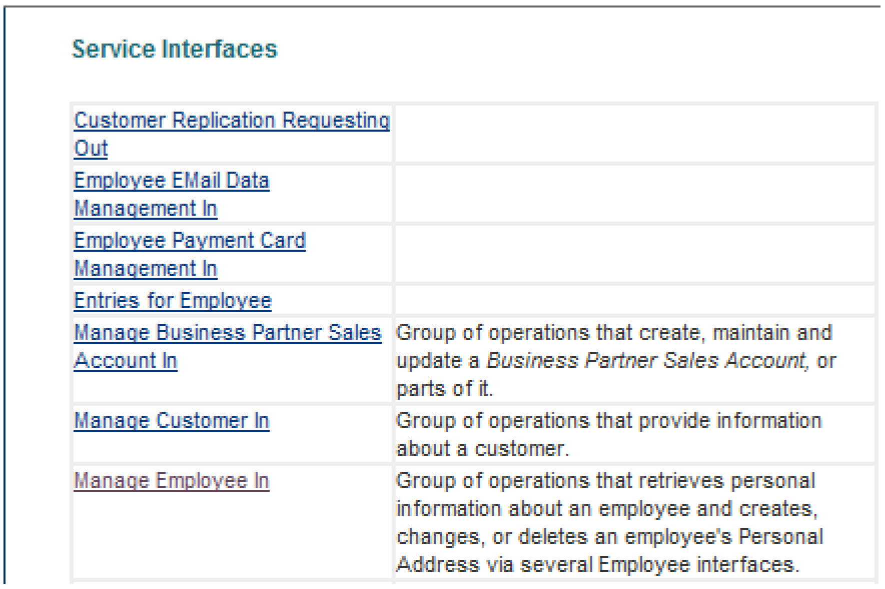
ES WORKPLACE

PROCESS COMPONENTS IN ESM ERP 603

- Academic Structure Management (Higher Education)
- Accounting
- Ad Content Processing
- Advertising Product Data Management
- Advertising Sales Order Processing
- Analytics

- Bank Account Contract Processing
- Benefits Administration
- Budget Processing
- Business Document Flow Processing
- Business Partner Data Management

5. The selection of *Business Partner Data Management* results in a list of enterprise service interfaces supporting this process component. Select the *Manage Employee In* service interface.

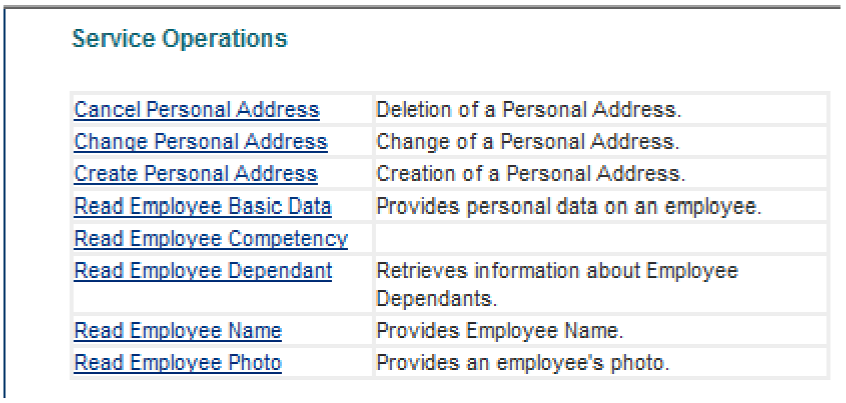


Service Interfaces

Customer Replication Requesting Out	
Employee EMail Data Management In	
Employee Payment Card Management In	
Entries for Employee	
Manage Business Partner Sales Account In	Group of operations that create, maintain and update a *Business Partner Sales Account*, or parts of it.
Manage Customer In	Group of operations that provide information about a customer.
Manage Employee In	Group of operations that retrieves personal information about an employee and creates, changes, or deletes an employee's Personal Address via several Employee interfaces.

6. This modeled interface comprises several Service Operations (Enterprise Services): Cancel Personal Address, Change Personal Address, Create Personal Address, Read Employee Basic Data and so on.

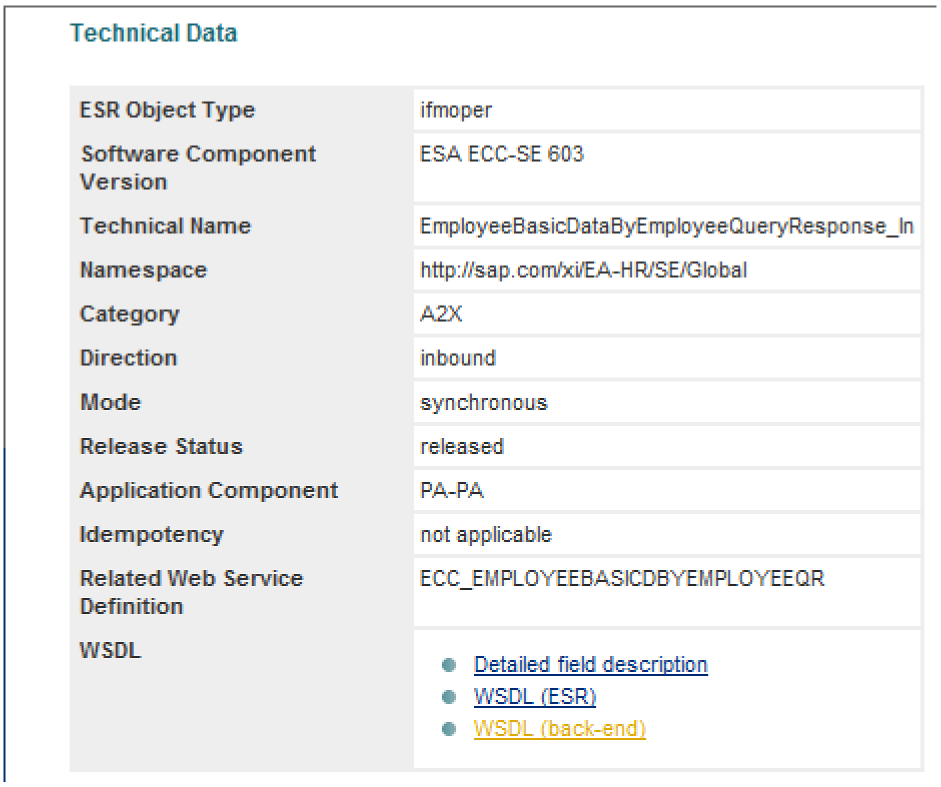
7. Select the *Read Employee Basic Data* Enterprise Service



Service Operations

Cancel Personal Address	Deletion of a Personal Address.
Change Personal Address	Change of a Personal Address.
Create Personal Address	Creation of a Personal Address.
Read Employee Basic Data	Provides personal data on an employee.
Read Employee Competency	
Read Employee Dependant	Retrieves information about Employee Dependants.
Read Employee Name	Provides Employee Name.
Read Employee Photo	Provides an employee's photo.

8. The Enterprise Service is by default documented with a definition, Technical Data, Business Context, etc. In the *Technical Data* section, select the *WSDL (backend)* link, which points to the related backend system and shows the WSDL for this enterprise service.



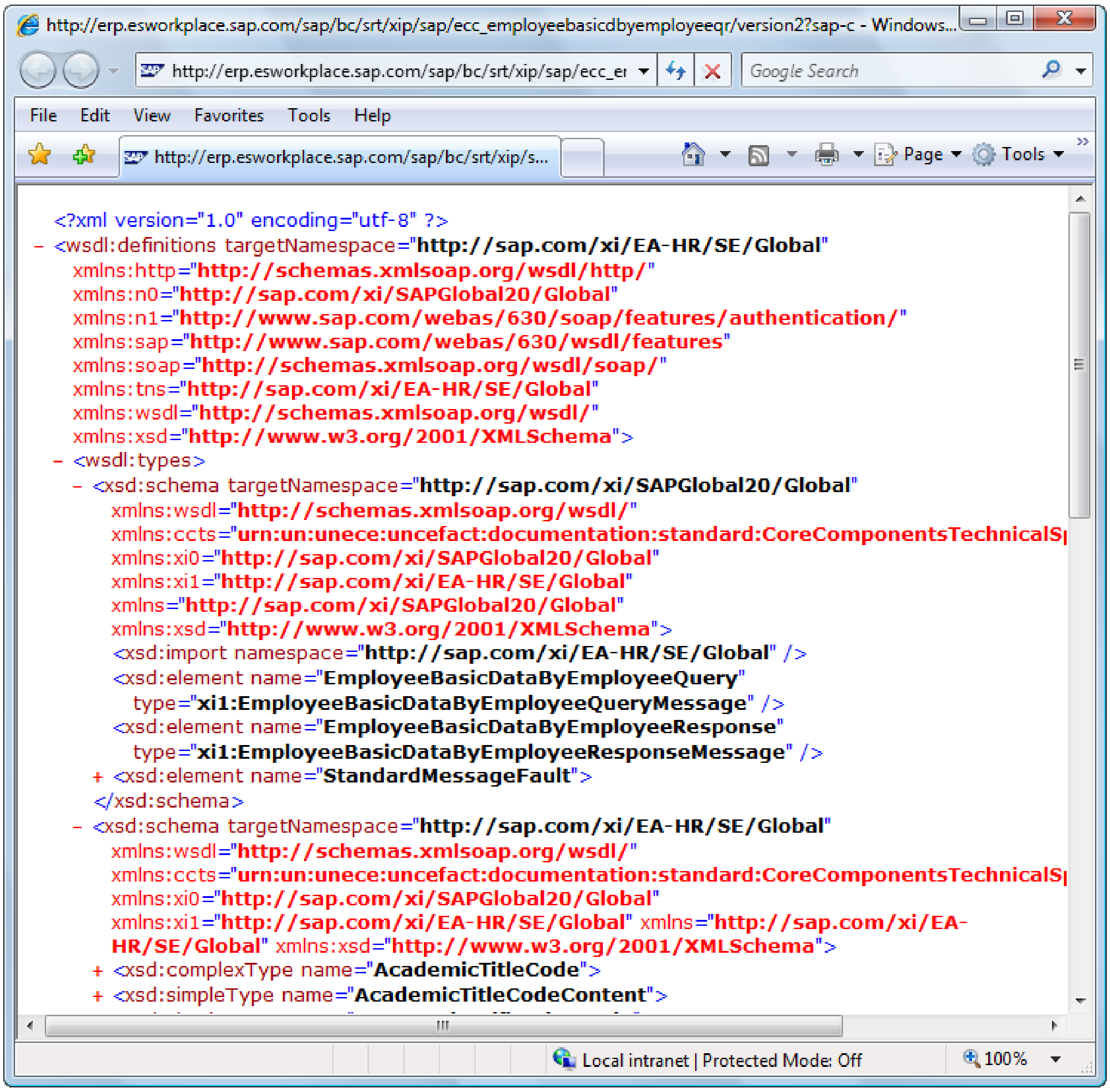
Technical Data

ESR Object Type	ifmoper
Software Component Version	ESA ECC-SE 603
Technical Name	EmployeeBasicDataByEmployeeQueryResponse_In
Namespace	http://sap.com/xi/EA-HR/SE/Global
Category	A2X
Direction	inbound
Mode	synchronous
Release Status	released
Application Component	PA-PA
Idempotency	not applicable
Related Web Service Definition	ECC_EMPLOYEEBASICDBYEMPLOYEEQR
WSDL	● Detailed field description ● WSDL (ESR) ● WSDL (back-end)

9. You will be prompted for authentication, enter your username and password which you obtained at the Enterprise SOA Registration Form. The next screen would bring up the WSDL as an XML file

    Note

    The WSDL descriptor describes the services in a language-independent manner. The WSDL file does not specify what the service does. It specifies only the parameter and return types


http://erp.esworkplace.sap.com/sap/bc/srt/xip/sap/ecc_employeebasicdbyemployeeqr/version2?sap-c - Windows...
http://erp.esworkplace.sap.com/sap/bc/srt/xip/sap/ecc_er
Google Search
File
Edit
View
Favorites
Tools
Help
http://erp.esworkplace.sap.com/sap/bc/srt/xip/s...
Page
Tools
Local intranet | Protected Mode: Off
100%

```
  <?xml version="1.0" encoding="utf-8" ?>
- <wsdl:definitions targetNamespace="http://sap.com/xi/EA-HR/SE/Global"
    xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
    xmlns:n0="http://sap.com/xi/SAPGlobal20/Global"
    xmlns:n1="http://www.sap.com/webas/630/soap/features/authentication/"
    xmlns:sap="http://www.sap.com/webas/630/wsdl/features"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:tns="http://sap.com/xi/EA-HR/SE/Global"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  - <wsdl:types>
    - <xsd:schema targetNamespace="http://sap.com/xi/SAPGlobal20/Global"
        xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
        xmlns:ccts="urn:un:unece:uncefact:documentation:standard:CoreComponentsTechnicalS
        xmlns:xi0="http://sap.com/xi/SAPGlobal20/Global"
        xmlns:xi1="http://sap.com/xi/EA-HR/SE/Global"
        xmlns="http://sap.com/xi/SAPGlobal20/Global"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <xsd:import namespace="http://sap.com/xi/EA-HR/SE/Global" />
        <xsd:element name="EmployeeBasicDataByEmployeeQuery"
          type="xi1:EmployeeBasicDataByEmployeeQueryMessage" />
        <xsd:element name="EmployeeBasicDataByEmployeeResponse"
          type="xi1:EmployeeBasicDataByEmployeeResponseMessage" />
      + <xsd:element name="StandardMessageFault">
      </xsd:schema>
    - <xsd:schema targetNamespace="http://sap.com/xi/EA-HR/SE/Global"
        xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
        xmlns:ccts="urn:un:unece:uncefact:documentation:standard:CoreComponentsTechnicalS
        xmlns:xi0="http://sap.com/xi/SAPGlobal20/Global"
        xmlns:xi1="http://sap.com/xi/EA-HR/SE/Global" xmlns="http://sap.com/xi/EA-
        HR/SE/Global" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      + <xsd:complexType name="AcademicTitleCode">
      + <xsd:simpleType name="AcademicTitleCodeContent">
```

10. The WSDL describes an EmployeeBasicDataByEmployeeQueryResponse_In operation name as follows:

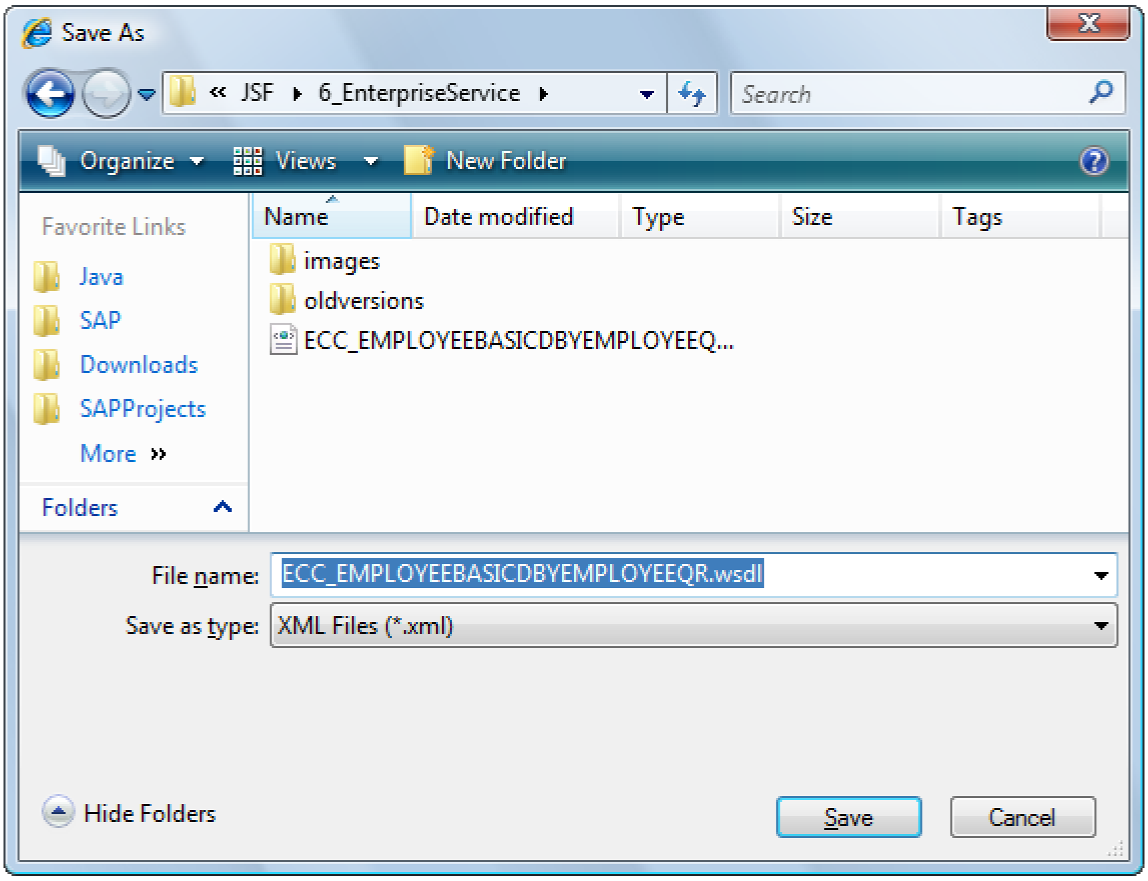
```
<wsdl:operation name="EmployeeBasicDataByEmployeeQueryResponse_In">
   <wsdl:input message="tns:EmployeeBasicDataByEmployeeQuery" />
   <wsdl:output message="tns:EmployeeBasicDataByEmployeeResponse" />
   <wsdl:fault name="exception00" message="tns:exception00" />
</wsdl:operation>
…
<wsdl:message name="EmployeeBasicDataByEmployeeQuery">
   <wsdl:part name="parameters"
      element="n0:EmployeeBasicDataByEmployeeQuery" />
```

```
</wsdl:message>
<wsdl:message name="EmployeeBasicDataByEmployeeResponse">
   <wsdl:part name="parameters"
      element="n0:EmployeeBasicDataByEmployeeResponse" />
</wsdl:message>
```

11. It also defines the *EmployeeBasicDataByEmployeeQuery* and *EmployeeBasicDataByEmployeeResponse* types as follows:

```
<xsd:element name="EmployeeBasicDataByEmployeeQuery"
   type="xi1:EmployeeBasicDataByEmployeeQueryMessage" />
 <xsd:element name="EmployeeBasicDataByEmployeeResponse"
   type="xi1:EmployeeBasicDataByEmployeeResponseMessage" />
```

12. With this information you will be able to use the enterprise service from your Application
13. Save the WSDL by using the *Save as* option of your browser. Note that the file extension has to be changed from XML to WSDL.

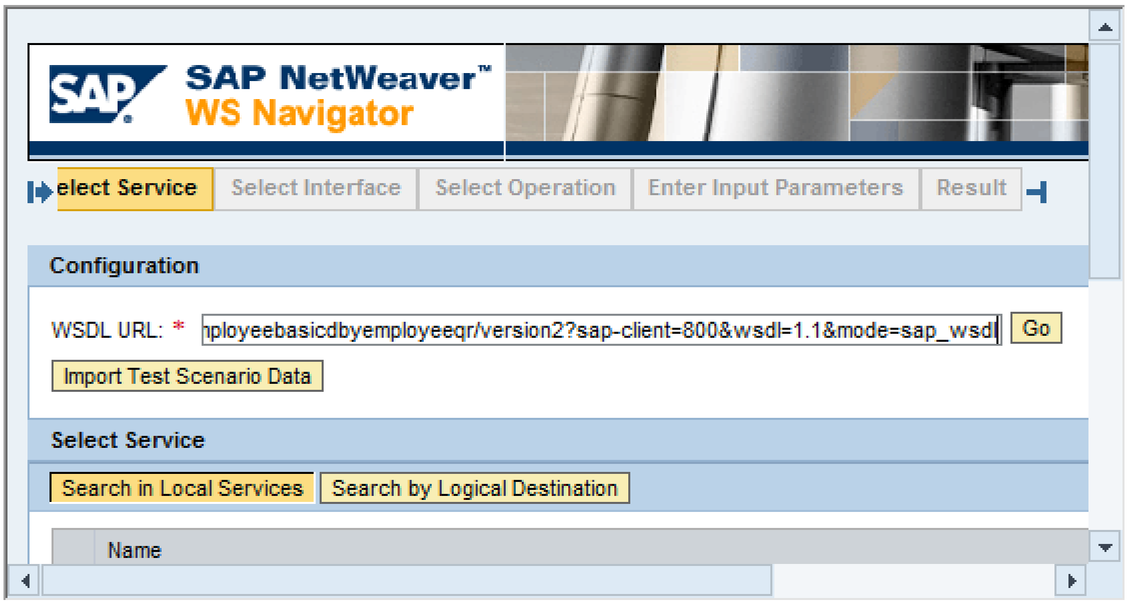


File name: ECC_EMPLOYEEBASICDBYEMPLOYEEQR.wsdl
Save as type: XML Files (*.xml)

4.3 Test the Enterprise Service in Web Services Navigator (Optional)

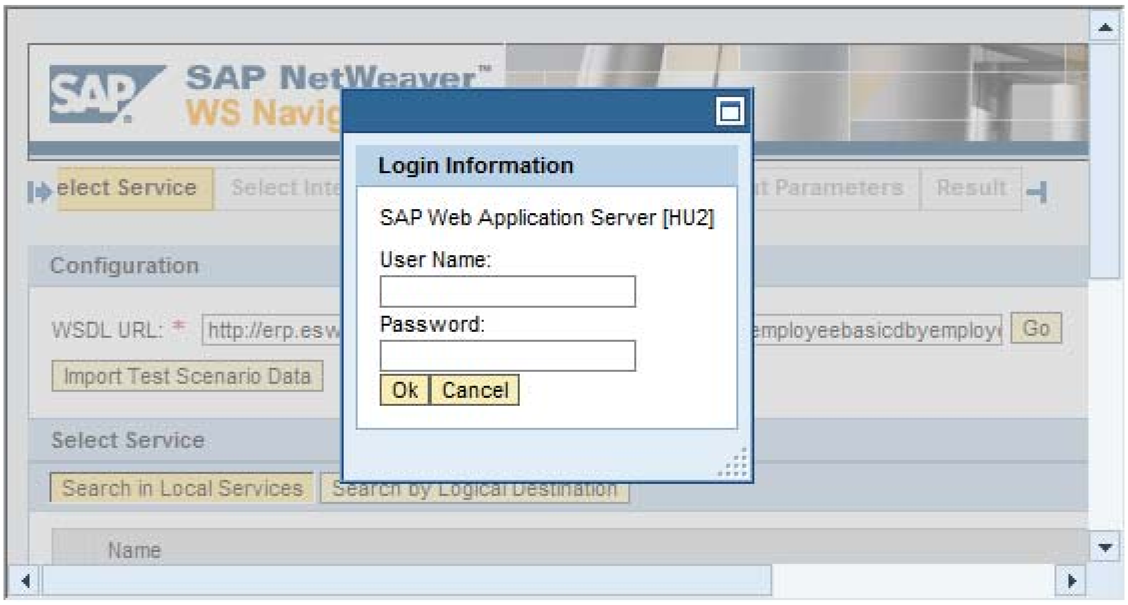
The Web Services Navigator (WS Navigator) allows you to test an enterprise service without having a client for it

1. To start the WS Navigator, in a Web browser, enter the following address: http://sr.esworkplace.sap.com/wsnavigator
2. The WS Navigator opens. In the WSDL URL field, enter the following address and push the *Go* button: http://erp.esworkplace.sap.com/sap/bc/srt/xip/sap/ecc_employeebasicdbyemployeeqr/version2?sap-client=800&wsdl=1.1&mode=sap_wsdl

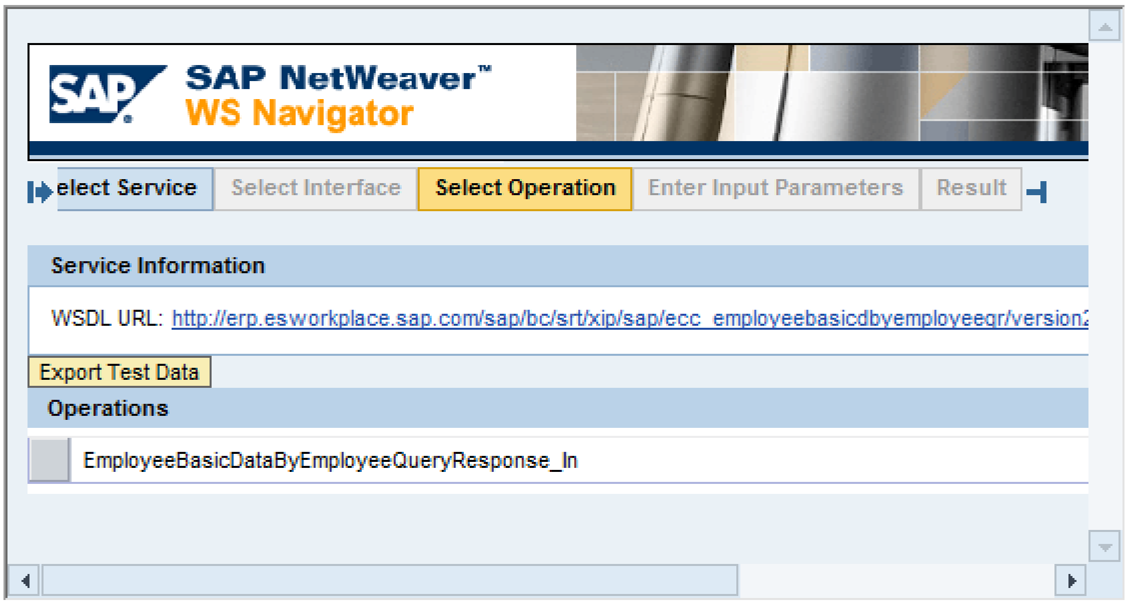


WSDL URL: * nployeebasicdbyemployeeqr/version2?sap-client=800&wsdl=1.1&mode=sap_wsdl

3. To execute the Web service operations, enter your user name and password, which you obtained at the Enterprise SOA Registration Form



4. In the Select Operation step, choose EmployeeBasicDataByEmployeeQueryResponse_In



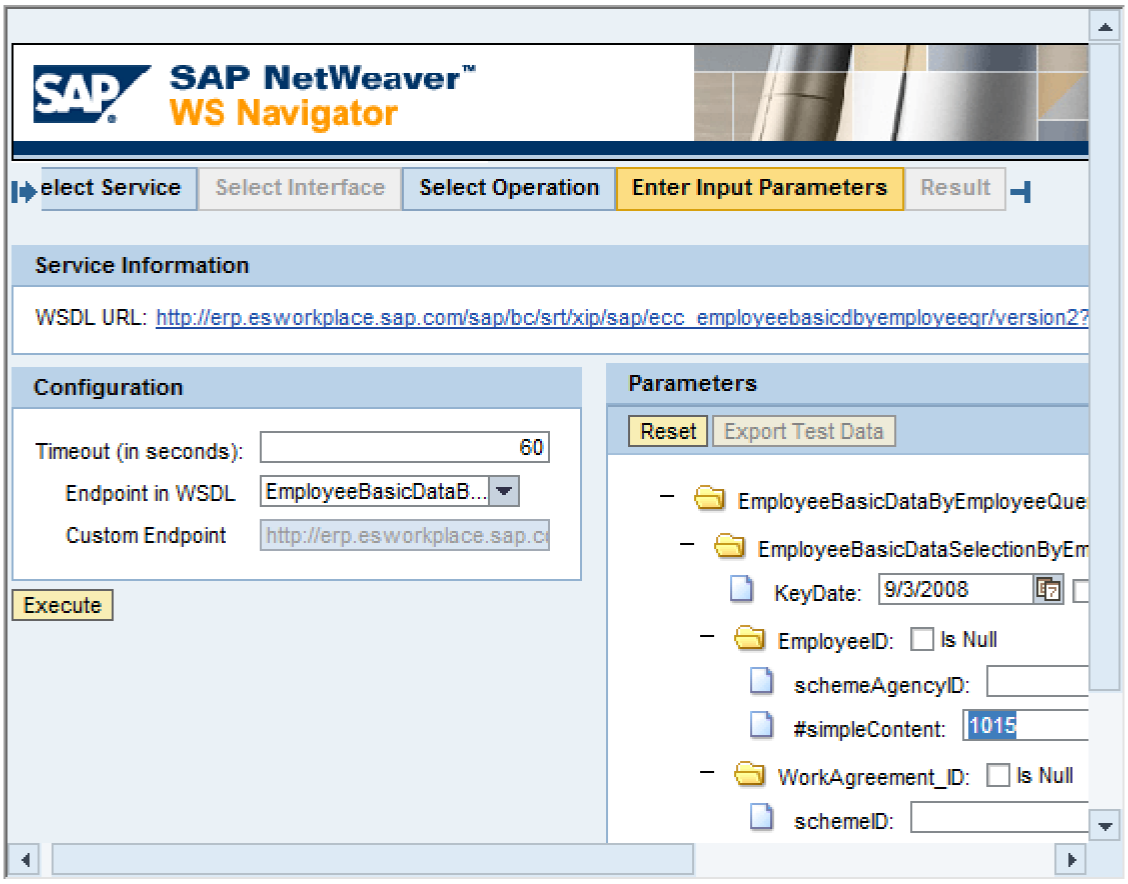
Service Information

WSDL URL: http://erp.esworkplace.sap.com/sap/bc/srt/xip/sap/ecc_employeebasicdbyemployeeqr/version2

Operations

EmployeeBasicDataByEmployeeQueryResponse_In

5. In the *Enter Input Parameters* step, in the *Parameters* area enter the *EmployeeID* and push the *Execute* button



Service Information

WSDL URL: http://erp.esworkplace.sap.com/sap/bc/srt/xip/sap/ecc_employeebasicdbyemployeeqr/version2?

Configuration

Timeout (in seconds): 60

Endpoint in WSDL EmployeeBasicDataB...

Custom Endpoint http://erp.esworkplace.sap.c

Parameters

EmployeeBasicDataByEmployeeQue

EmployeeBasicDataSelectionByEm

KeyDate: 9/3/2008

EmployeeID: Is Null

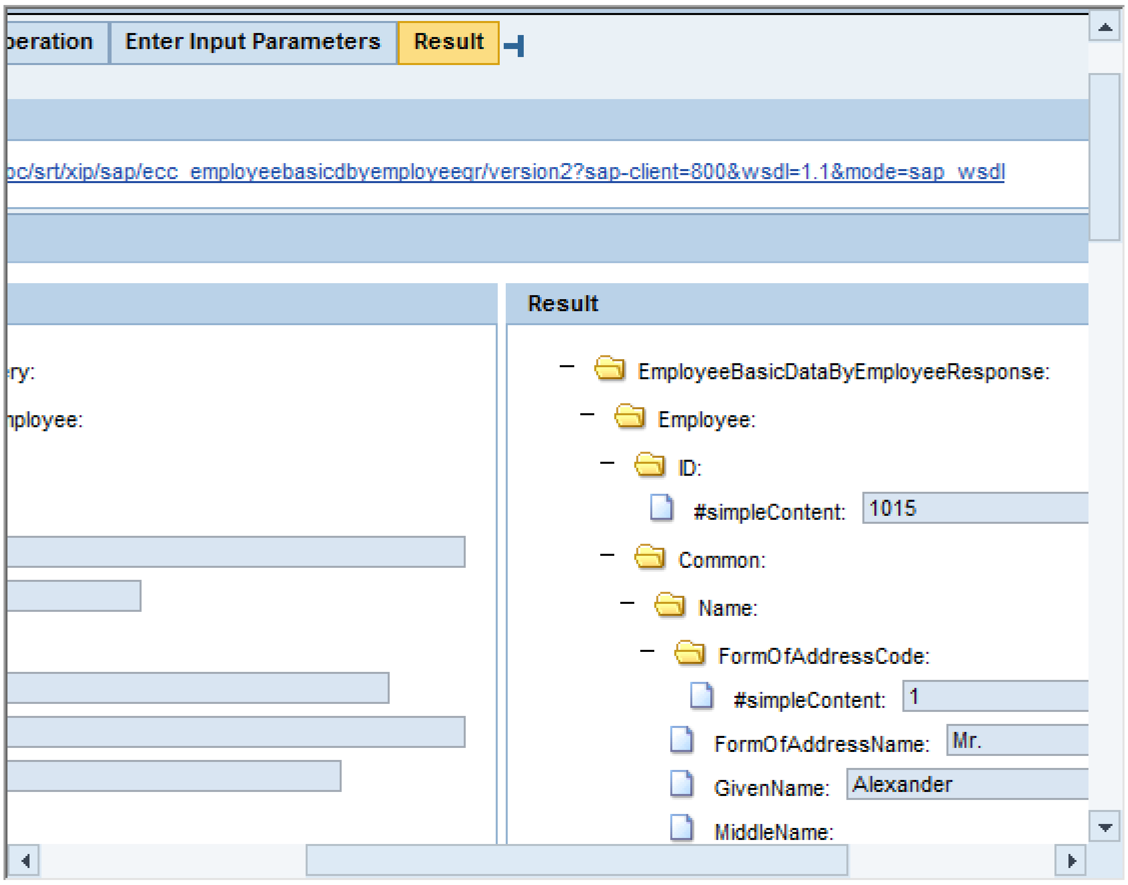
schemeAgencyID:

#simpleContent: 1015

WorkAgreement_ID: Is Null

schemeID:

6. In the *Result* step, the system displays the input parameters and the result of the test



pc/srt/xip/sap/ecc_employeebasicdbyemployeeqr/version2?sap-client=800&wsdl=1.1&mode=sap_wsdl

ry:
nployee:

Result
- EmployeeBasicDataByEmployeeResponse:
- Employee:
- ID:
#simpleContent: 1015
- Common:
- Name:
- FormOfAddressCode:
#simpleContent: 1
FormOfAddressName: Mr.
GivenName: Alexander
MiddleName:

4.4 Import Enterprise Service Definition

You need to import a WSDL document to be able to generate enterprise service proxies.

Important

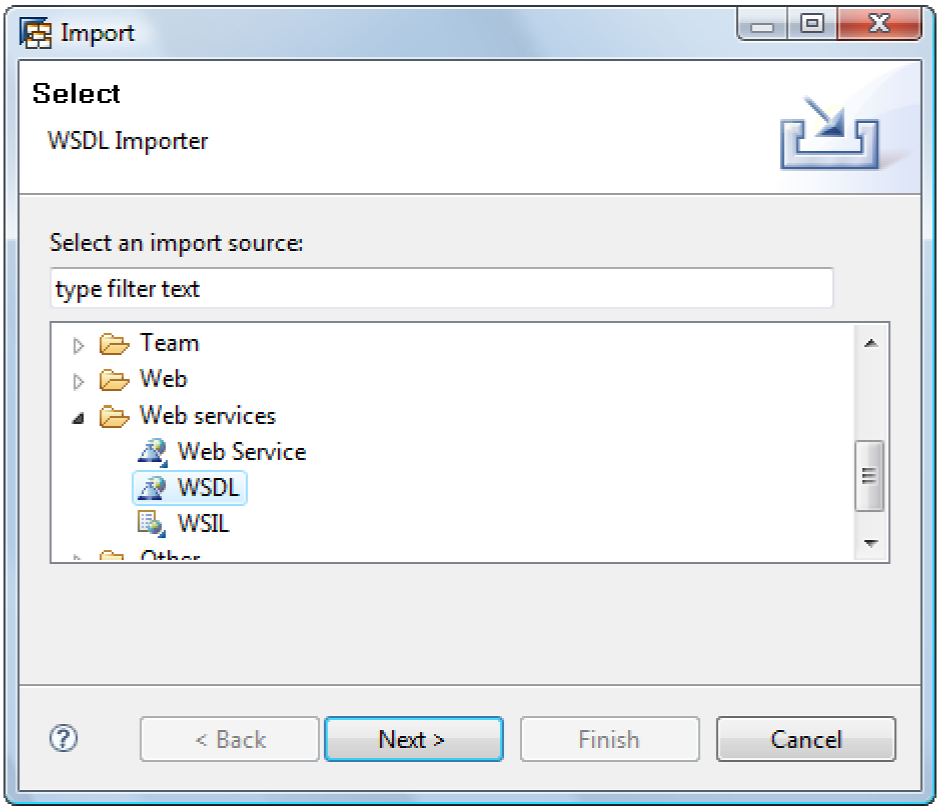
You can import WSDL documents from the following sources:

Enterprise Services Repository (ESR): You import the WSDL document for a Service Interface (SI) modeled in the ESR

File system or a remote location such as a URL

Services Registry (SR): You import the WSDL document of a Web service published in the SR

1. Choose *File* → *Import* → *Web services* → *Import WSDL*.


Import

Select

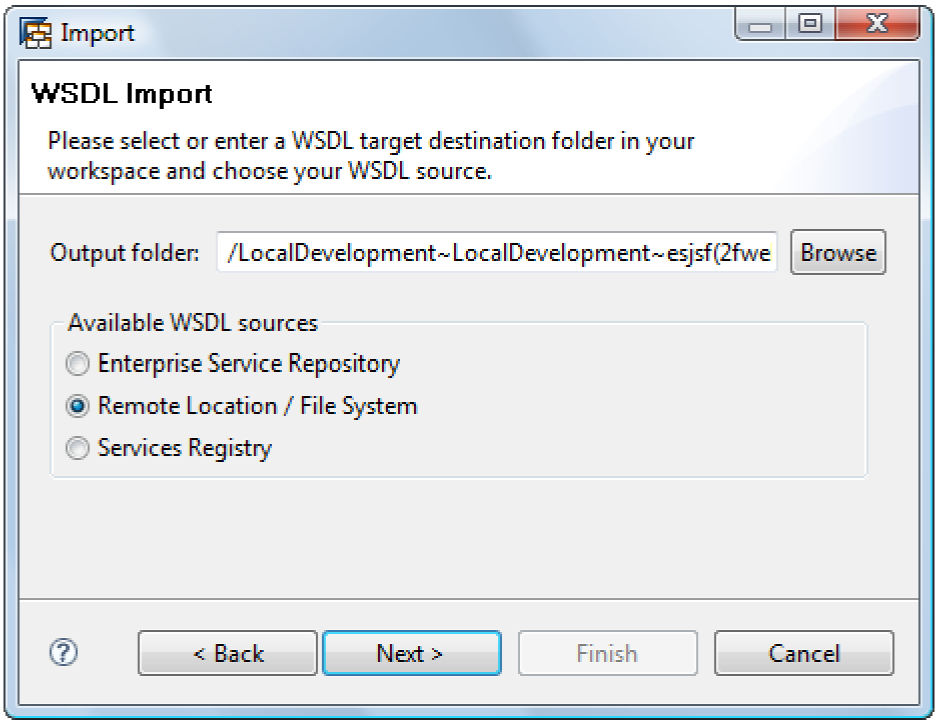
WSDL Importer

Select an import source:

type filter text

- Team
- Web
- Web services
  - Web Service
  - WSDL
  - WSIL
- Other

2. The WSDL Import wizard starts. In Output folder field, browse to the Web module project where you want to import the WSDL document. Under Available wsdl sources choose *Remote Location / File System*, and then choose the *Next* button.


Import

WSDL Import

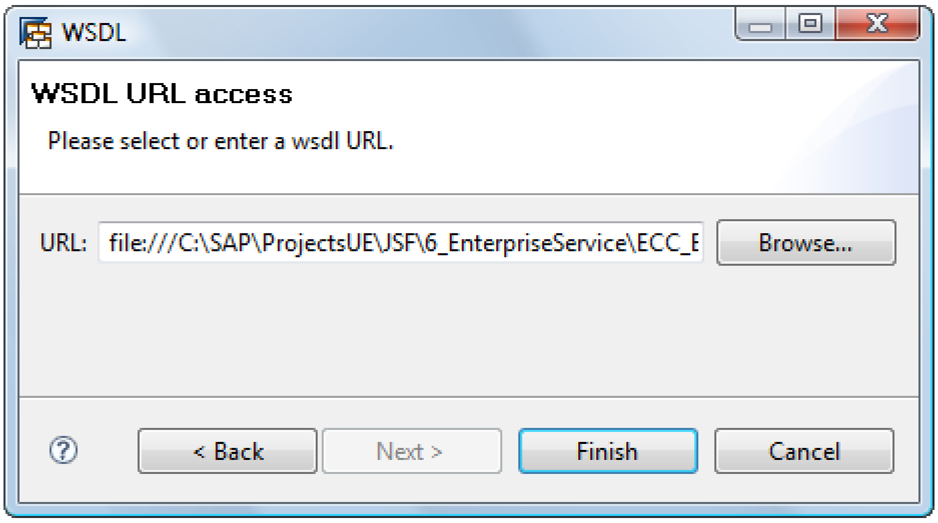
Please select or enter a WSDL target destination folder in your workspace and choose your WSDL source.

Output folder: /LocalDevelopment~LocalDevelopment~esjsf(2fwe

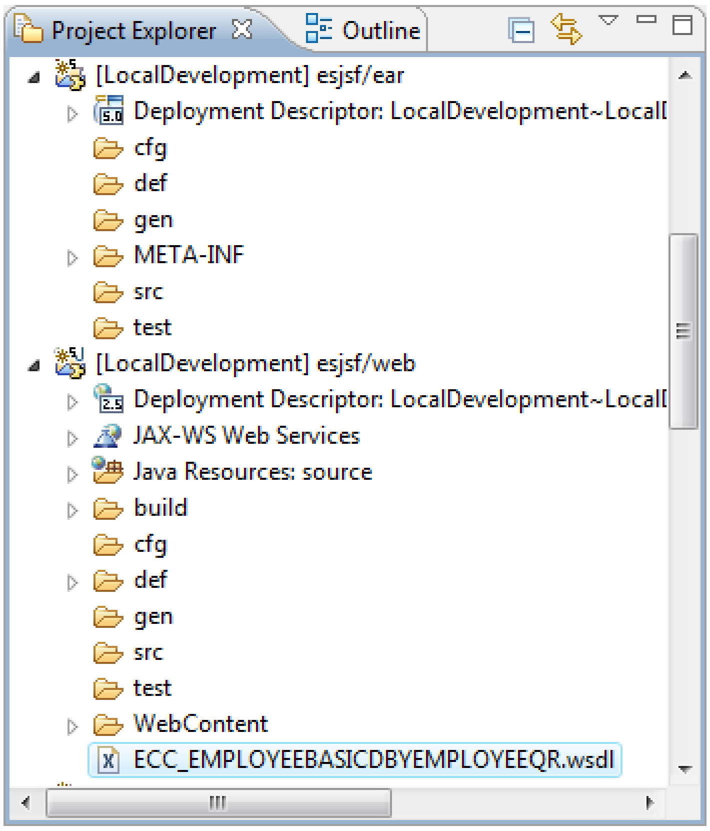
Available WSDL sources

- Enterprise Service Repository
- [✓ Remote Location / File System]
- Services Registry

3. In the *URL* field, provide the path to the WSDL document, and then choose *Finish*


WSDL
WSDL URL access
Please select or enter a wsdl URL.
URL: file:///C:\SAP\ProjectsUE\JSF\6_EnterpriseService\ECC_E
Browse...
< Back
Next >
Finish
Cancel

4. The WSDL document is imported in the SAP NetWeaver Developer Studio. The original filename is preserved. If the WSDL document imports other WSDL documents or schemas, their filenames are preserved as well


Project Explorer
Outline
[LocalDevelopment] esjsf/ear
Deployment Descriptor: LocalDevelopment~LocalI
cfg
def
gen
META-INF
src
test
[LocalDevelopment] esjsf/web
Deployment Descriptor: LocalDevelopment~LocalI
JAX-WS Web Services
Java Resources: source
build
cfg
def
gen
src
test
WebContent
ECC_EMPLOYEEBASICDBYEMPLOYEEQR.wsdl

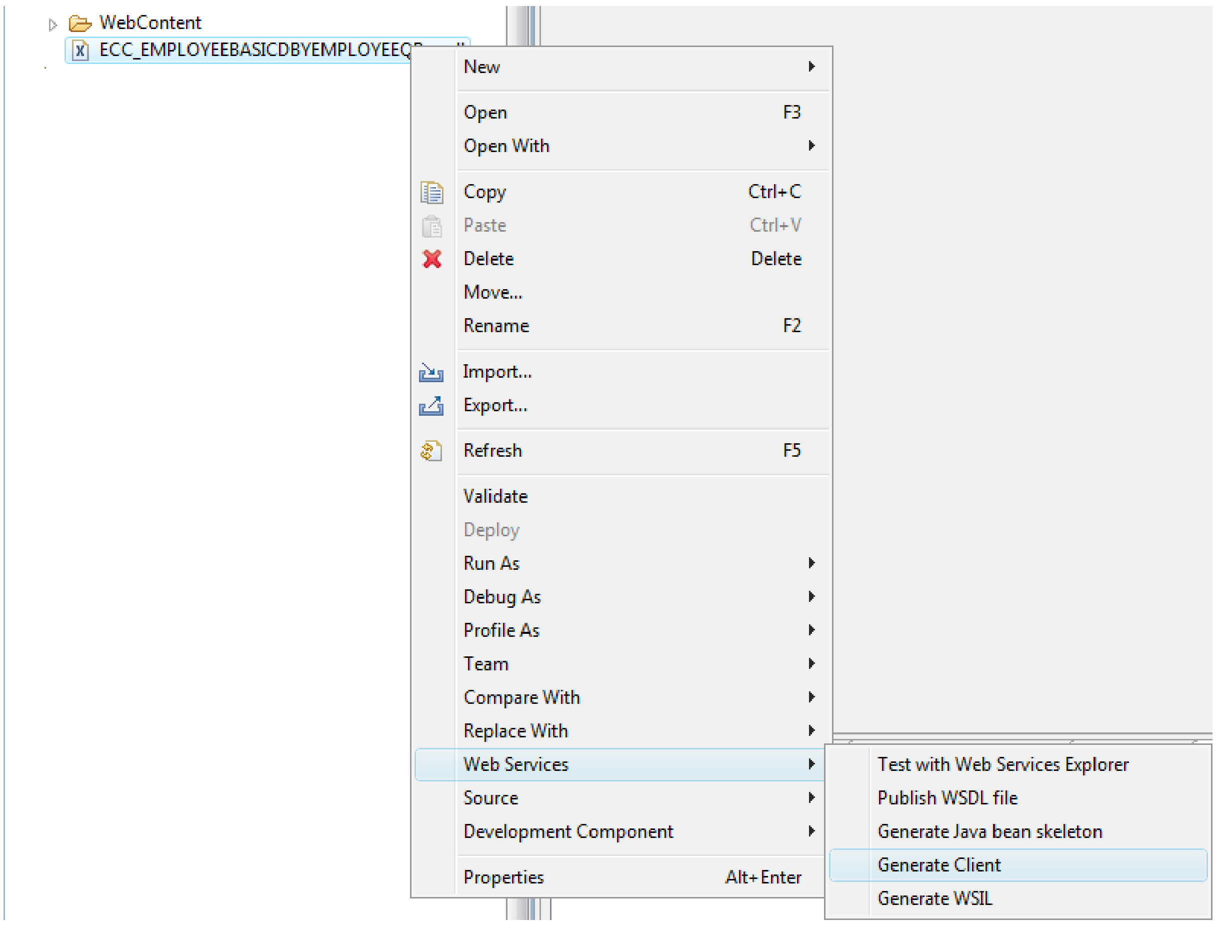
4.5 Generate Enterprise Service Proxy

In this part you will generate a deployable proxy out of the downloaded WSDL.

Note

A deployable proxy is a Web service client, which can and has to be deployed on the application server to be operational. A deployable proxy is consumed from an Enterprise Java Bean, a servlet or a Java class and is created and configured by the responsible managing container of the application server. Deployable proxies are used through the Java EE 5 injection mechanism.

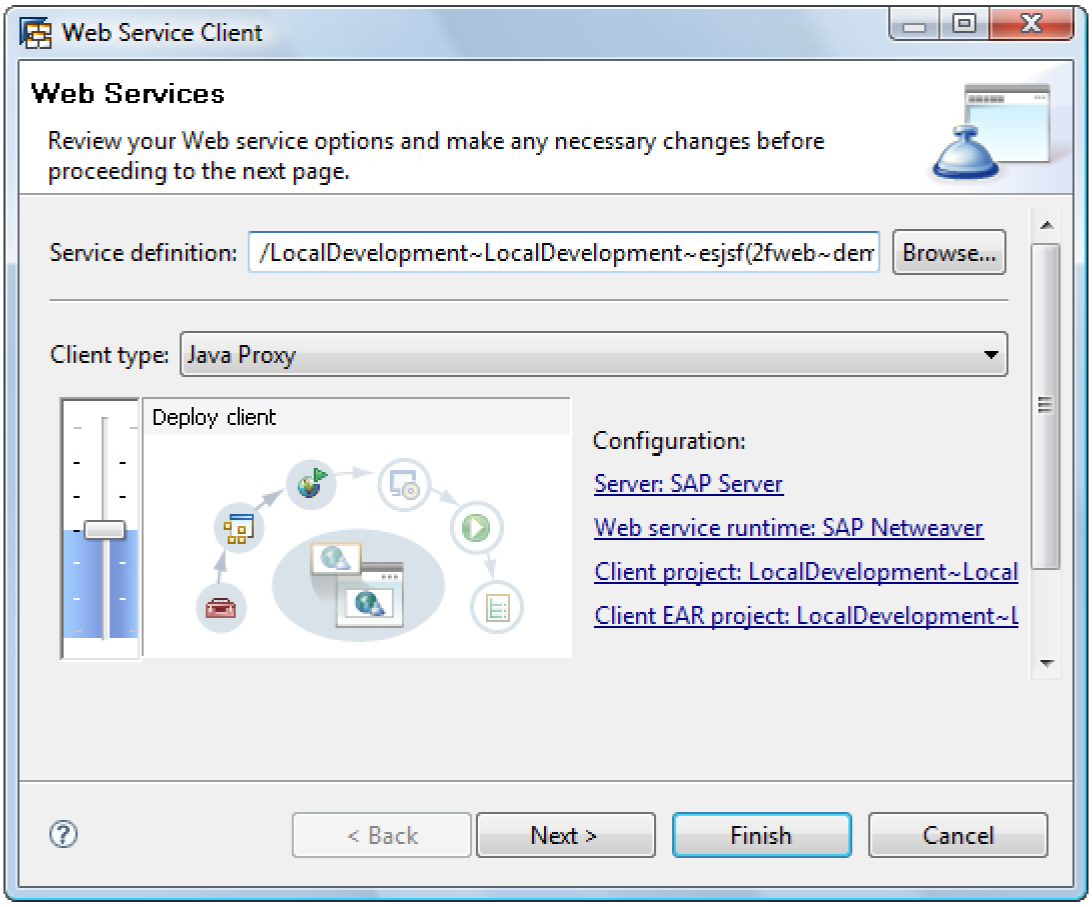
1. In the Project Explorer of the Web Module project, choose the WSDL document, and then from the context menu, choose *Web Services* → *Generate Client*.



2. The Web Services wizard opens. On the *Web Services* screen, move the slider to the *Develop Client* position. In this case, the Web service framework only generates the relevant artifacts. Push the *Next* button.

   Note

   Under *Configuration*, choose Service EAR Project, and make sure that the correct service EAR project is selected in the Service project field



Web Services

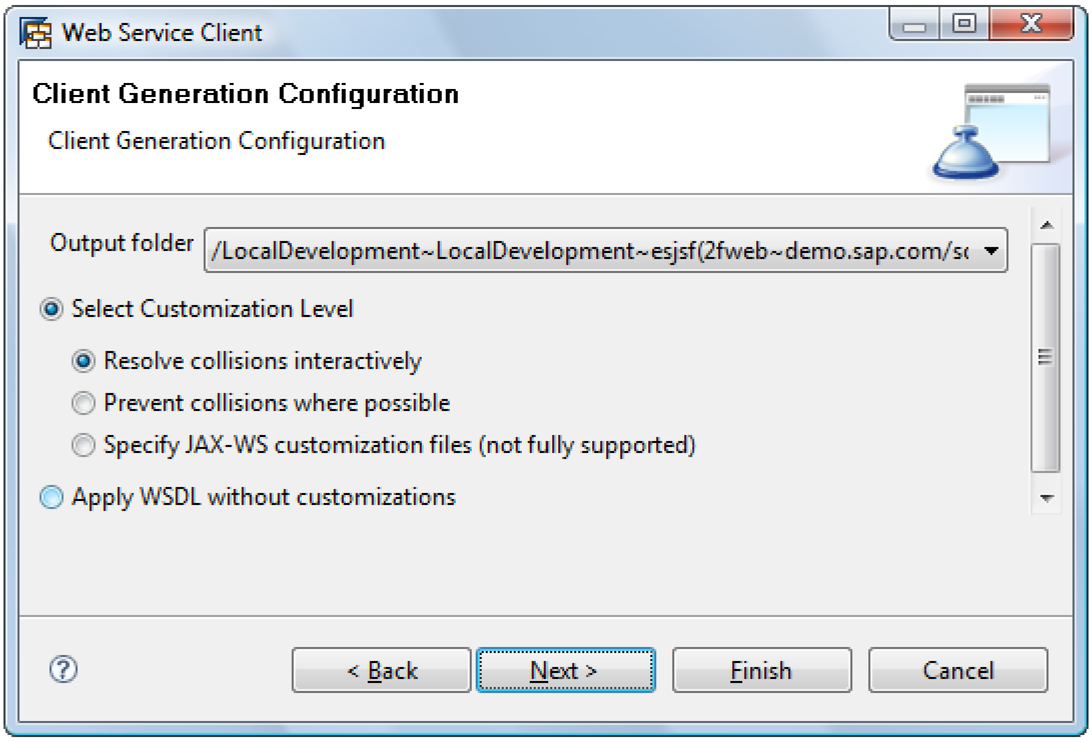
Review your Web service options and make any necessary changes before proceeding to the next page.

Service definition: /LocalDevelopment~LocalDevelopment~esjsf(2fweb~dem

Client type: Java Proxy

Configuration:
Server: SAP Server
Web service runtime: SAP Netweaver
Client project: LocalDevelopment~Local
Client EAR project: LocalDevelopment~L

3. On the *Client Generation Configuration* screen, choose the **`src`** folder of the Web Module project and choose the way you want to resolve the possible collisions in the WSDL to Java mapping during the conversion of the WSDL document to Java artifacts. Push the *Finish* button.



Client Generation Configuration

Client Generation Configuration

Output folder /LocalDevelopment~LocalDevelopment~esjsf(2fweb~demo.sap.com/s

[● Select Customization Level]

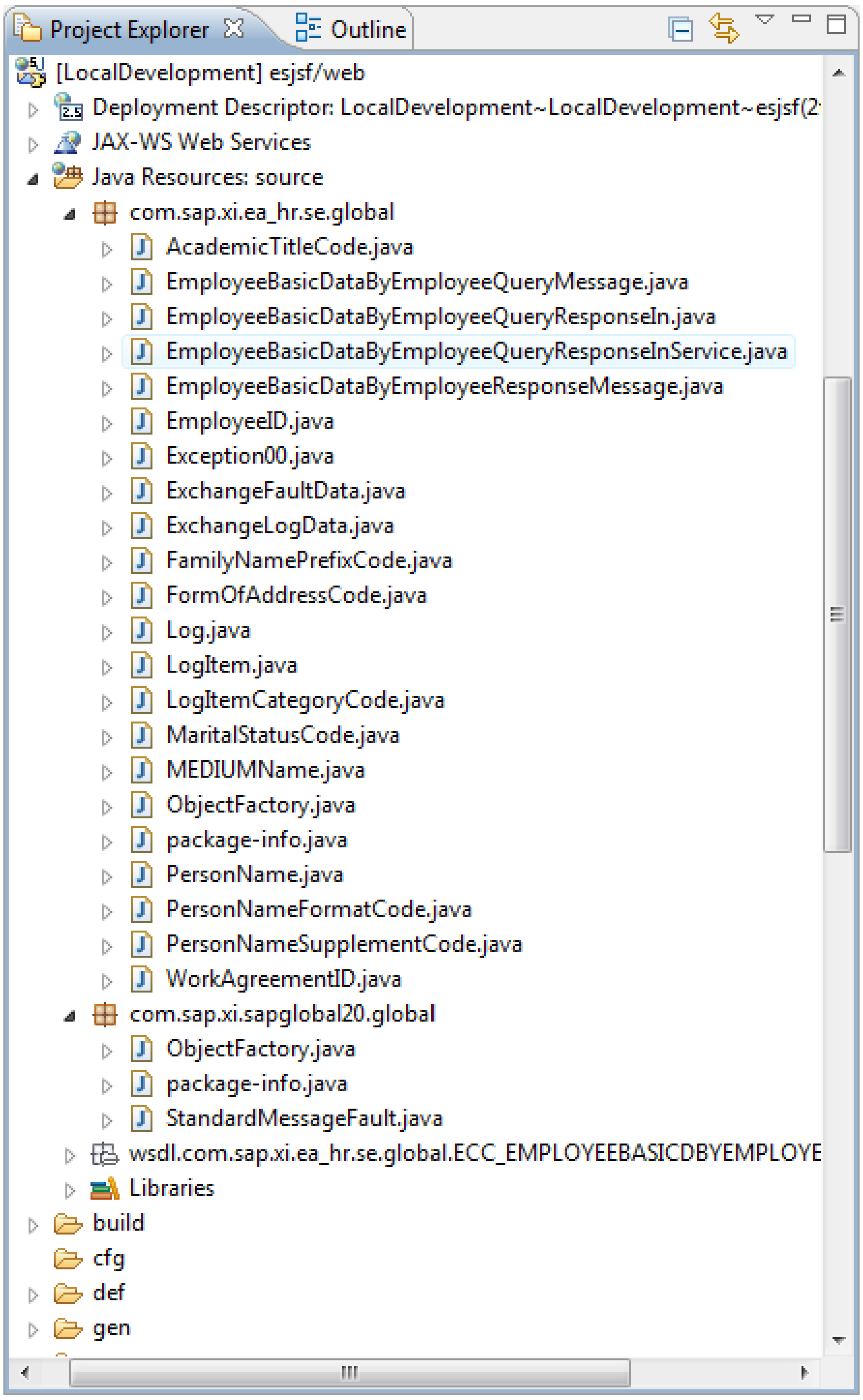
[● Resolve collisions interactively]

○ Prevent collisions where possible

○ Specify JAX-WS customization files (not fully supported)

○ Apply WSDL without customizations

4. The framework generates the service endpoint interface, reference classes and the service class in the output folder and the class path is updated with additional JAR files



```
Project Explorer    Outline
[LocalDevelopment] esjsf/web
  ▷ Deployment Descriptor: LocalDevelopment~LocalDevelopment~esjsf(2
  ▷ JAX-WS Web Services
  ◢ Java Resources: source
    ◢ com.sap.xi.ea_hr.se.global
      ▷ AcademicTitleCode.java
      ▷ EmployeeBasicDataByEmployeeQueryMessage.java
      ▷ EmployeeBasicDataByEmployeeQueryResponseIn.java
      ▷ EmployeeBasicDataByEmployeeQueryResponseInService.java
      ▷ EmployeeBasicDataByEmployeeResponseMessage.java
      ▷ EmployeeID.java
      ▷ Exception00.java
      ▷ ExchangeFaultData.java
      ▷ ExchangeLogData.java
      ▷ FamilyNamePrefixCode.java
      ▷ FormOfAddressCode.java
      ▷ Log.java
      ▷ LogItem.java
      ▷ LogItemCategoryCode.java
      ▷ MaritalStatusCode.java
      ▷ MEDIUMName.java
      ▷ ObjectFactory.java
      ▷ package-info.java
      ▷ PersonName.java
      ▷ PersonNameFormatCode.java
      ▷ PersonNameSupplementCode.java
      ▷ WorkAgreementID.java
    ◢ com.sap.xi.sapglobal20.global
      ▷ ObjectFactory.java
      ▷ package-info.java
      ▷ StandardMessageFault.java
    ▷ wsdl.com.sap.xi.ea_hr.se.global.ECC_EMPLOYEEBASICDBYEMPLOYE
    ▷ Libraries
  ▷ build
    cfg
  ▷ def
  ▷ gen
```

4.6 Create a Java class

Use this procedure to invoke the Enterprise Service proxy in your application

1. From the context menu of the *Java Resources: source* folder in the *Web Module* project, create a Java class. Enter **`EmployeeClient`** in the *Name* field, **`com.sap.tutorial.jsf.es.beans`** in the *Package* field

2. Add the following annotated field to the Java class to inject the service reference:

   Note

   The proxy is consumed from an EJB or Web module and is created and configured by the responsible managing container. Therefore, to allow the container to create and configure the proxy, a Java EE injection mechanism has to be used. This is achieved with the @WebServiceRef(name="MyName").The string "MyName" has to be unique for the Enterprise JavaBean that uses the client or for the whole Web module.

```
public class EmployeeClient {

   @WebServiceRef(name = "EmployeeBasicData")

   EmployeeBasicDataByEmployeeQueryResponseInService service;

}
```

3. Declare the following attributes and generate the corresponding *Getters* and *Setters* methods

```
String employeeID;

String employeeName;

String employeeLastname;

XMLGregorianCalendar employeeBirthday;

String employeeMaritalStatus;
```

4. Create a *search* method and implement it with the following code

   Note

   Notice the *search* method is returning a String that will indicate whether the specified employee was found. You will need to hook up this method to The CommandButton UI element in your JSF view to execute the search

```
public String search(){

   try {

      // Retrieve an instance of the configuration port

      EmployeeBasicDataByEmployeeQueryResponseIn port = service

         .getEmployeeBasicDataByEmployeeQueryResponse_InSoapBinding();

      // specify username and password issued for you via SDN

      // to authenticate to the backend as follows

      javax.xml.ws.BindingProvider bp =

         (javax.xml.ws.BindingProvider) port;

      Map<String, Object> context = bp.getRequestContext();

      context.put("javax.xml.ws.security.auth.username", "username");

      context.put("javax.xml.ws.security.auth.password", "password");

      // Construct the input to be applied to the service call

      EmployeeID employeeID = new EmployeeID();
```

```
            employeeID.setValue(this.employeeID);
            EmployeeBasicDataSelectionByEmployee employeeByEmployee =
               new EmployeeBasicDataSelectionByEmployee();
            employeeByEmployee.setEmployeeID(employeeID);
            EmployeeBasicDataByEmployeeQueryMessage input =
               new EmployeeBasicDataByEmployeeQueryMessage();
            input.setEmployeeBasicDataSelectionByEmployee(employeeByEmployee);
            // Invoke the business methods
            EmployeeBasicDataByEmployeeResponseMessage output = port
                  .employeeBasicDataByEmployeeQueryResponseIn(input);
            Employee employee = output.getEmployee();
            //Get the response
            this.employeeName = employee.getCommon().getName().getGivenName();
            this.employeeLastname =
               employee.getCommon().getName().getFamilyName();
            this.employeeBirthday = employee.getCommon().getBirthDate();
            this.employeeMaritalStatus =
               employee.getCommon().getMaritalStatusName().getValue();
            return "ok";
         } catch (Exception e) {
            //Clean the object and return an error
            this.employeeID = "";
            this.employeeName = "";
            this.employeeLastname = "";
            this.employeeBirthday = null;
            this.employeeMaritalStatus = "";
            return "error";
         }
      }
```

5. Configure the *EmployeeClient* Java class in the application configuration resource file faces-config.xml using the managed-bean XML element. Enter **`employeeClient`** in the *Name* field to reference the *EmployeeClient* java class and select **`session`** in the *Scope* field. The following XML code will be added in the Source tab

```
<managed-bean>
   <managed-bean-name>employeeClient</managed-bean-name>
   <managed-bean-class>
      com.sap.tutorial.jsf.es.beans.EmployeeClient
```

```
        </managed-bean-class>
        <managed-bean-scope>session</managed-bean-scope>
    </managed-bean>
```

4.7 Create JSF Pages

1. From the context menu of the *Java Resources: source* folder in the *Web Module* project, create a new package **`com.sap.tutorial.jsf.es.util`**
2. Create the ResourceBundle. Choose the **`com.sap.tutorial.jsf.es.util`** package and enter **`messages.properties`** in the *File Name*
3. Enter the following keys and values for the English version of the localized messages

```
title=Search Employee
emp_id=Employee ID
search_button=Search
emp_name=Name
emp_lastname=Last Name
emp_birthday=Birthday
emp_marital=Marital Status
errorOccurred=Sorry, there is no employee with the ID specified.
errorBack=Please go back and try again
back_button=Back
```

4. For simplicity, only the English version of the localized message is created. Optionally you can create other versions of the localized messages and specify which languages are supported for this application as indicated in the Product Offer tutorial Part 3 (International JSF application [extern]).
5. Expose the ResourceBundles by adding the following XML code in the Source tab of the faces-config.xml file

```
<application>
    <resource-bundle>
        <base-name>com.sap.tutorial.jsf.es.util.messages</base-name>
        <var>msgs</var>
    </resource-bundle>
</application>
```

6. Create a Style file as indicated in the Product Offer tutorial (Convert and Validate Data [extern]). You are going to use the same CSS classes, so you can copy the **`styles.css`** file from that tutorial.
7. Drill into the Web Module project and right click on the *WebContent* folder and in the context menu select *New → JSP.*

8. Enter the file name **`index.jsp`** and click the *Finish* button. The JSP page will be created. The *index.jsp* page should be opened in the *Web Page Editor*
9. Include the style sheet by adding a *link* element inside the *head* element as shown in the following code

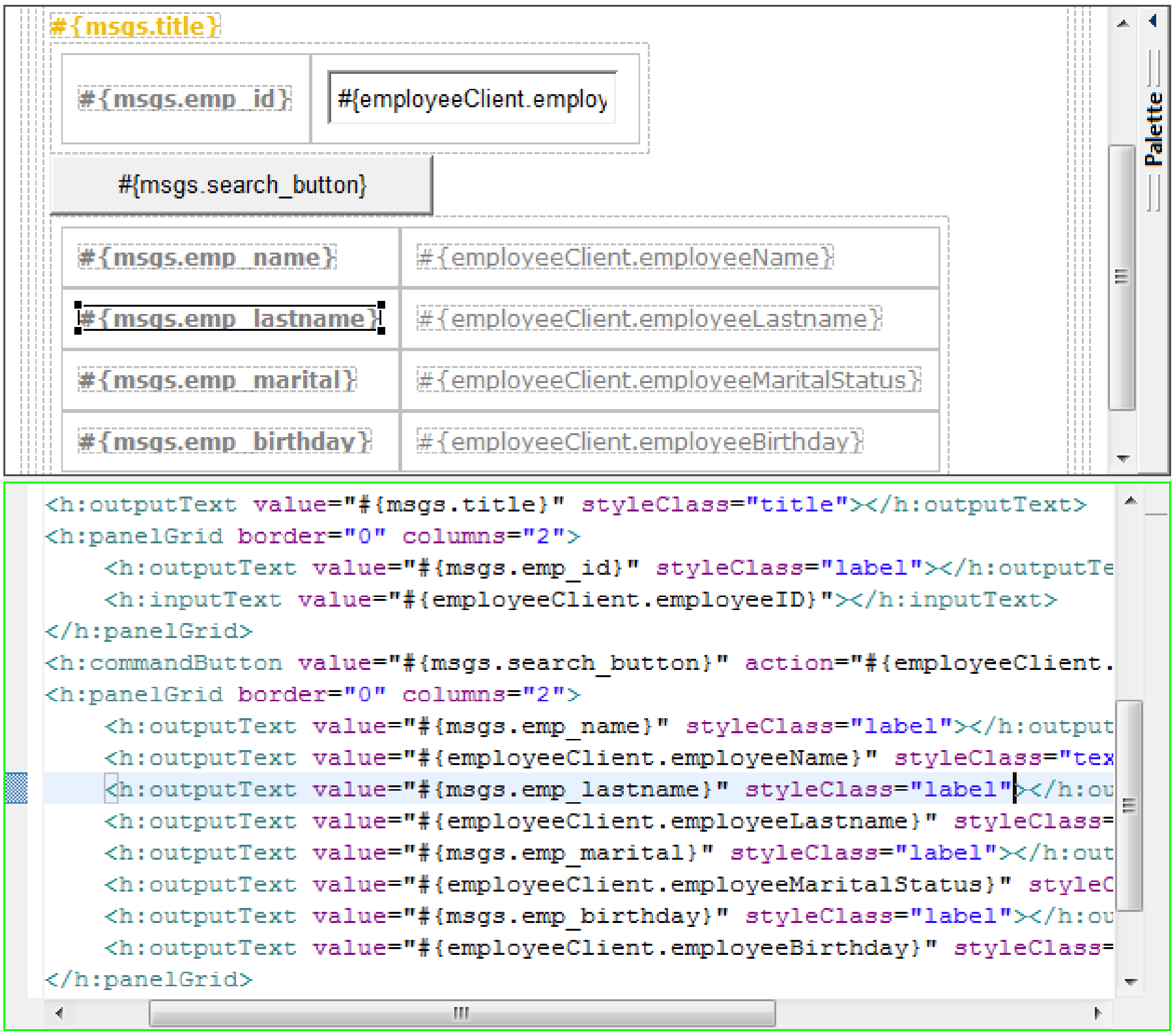
```
<head>
    <link href="styles.css" rel="stylesheet" type="text/css"/>
    …
</head>
```

10. The following table contains the hierarchy of the UI elements contained in the *index* view:

Property	Value
***ViewRoot* UI element**	
Form* UI element in the UI-element *ViewRoot	
OutputText* UI element in the UI-element *Form	
value	#{msgs.title}
styleClass	title
PanelGrid* UI element in the UI-element *Form	
Border	0
Columns	2
OutputText UI element in the UI-element *PanelGrid*	
value	#{msgs.emp_id}
styleClass	label
InputText UI element in the UI-element *PanelGrid*	
value	#{employeeClient.employeeID}
CommandButton UI element in the UI-element *Form*	
value	#{msgs.search_button}
action	#{employeeClient.search}
PanelGrid* UI element in the UI-element *Form	
Border	0
Columns	2
OutputText UI element in the UI-element *PanelGrid*	
value	#{msgs.emp_name}
styleClass	label
OutputText UI element in the UI-element *PanelGrid*	
value	#{employeeClient.employeeName}
styleClass	text

OutputText UI element in the UI-element *PanelGrid*	
value	#{msgs.emp_lastname}
styleClass	label
OutputText UI element in the UI-element *PanelGrid*	
value	#{employeeClient.employeeLastname}
styleClass	text
OutputText UI element in the UI-element *PanelGrid*	
value	#{msgs.emp_marital}
styleClass	label
OutputText UI element in the UI-element *PanelGrid*	
value	#{employeeClient.employeeMaritalStatus}
styleClass	text
OutputText UI element in the UI-element *PanelGrid*	
value	#{msgs.emp_birthday}
styleClass	label
OutputText UI element in the UI-element *PanelGrid*	
value	#{employeeClient.employeeBirthday}
styleClass	text

11. Result of index.jsp



```
<h:outputText value="#{msgs.title}" styleClass="title"></h:outputText>
<h:panelGrid border="0" columns="2">
    <h:outputText value="#{msgs.emp_id}" styleClass="label"></h:outputTe
    <h:inputText value="#{employeeClient.employeeID}"></h:inputText>
</h:panelGrid>
<h:commandButton value="#{msgs.search_button}" action="#{employeeClient.
<h:panelGrid border="0" columns="2">
    <h:outputText value="#{msgs.emp_name}" styleClass="label"></h:output
    <h:outputText value="#{employeeClient.employeeName}" styleClass="tex
    <h:outputText value="#{msgs.emp_lastname}" styleClass="label"></h:ou
    <h:outputText value="#{employeeClient.employeeLastname}" styleClass=
    <h:outputText value="#{msgs.emp_marital}" styleClass="label"></h:out
    <h:outputText value="#{employeeClient.employeeMaritalStatus}" styleC
    <h:outputText value="#{msgs.emp_birthday}" styleClass="label"></h:ou
    <h:outputText value="#{employeeClient.employeeBirthday}" styleClass=
</h:panelGrid>
```

12. Save the changes you made.

13. Create the another view selecting *New* → *JSP* in the context menu of the *WebContent* folder

14. Enter the file name **`errorDisplay.jsp`** and click the *Finish* button. The JSP page will be created. The *errorDisplay.jsp* page should be opened in the *Web Page Editor*

15. Include the style sheet by adding a *link* element inside the *head* element as shown in the following code

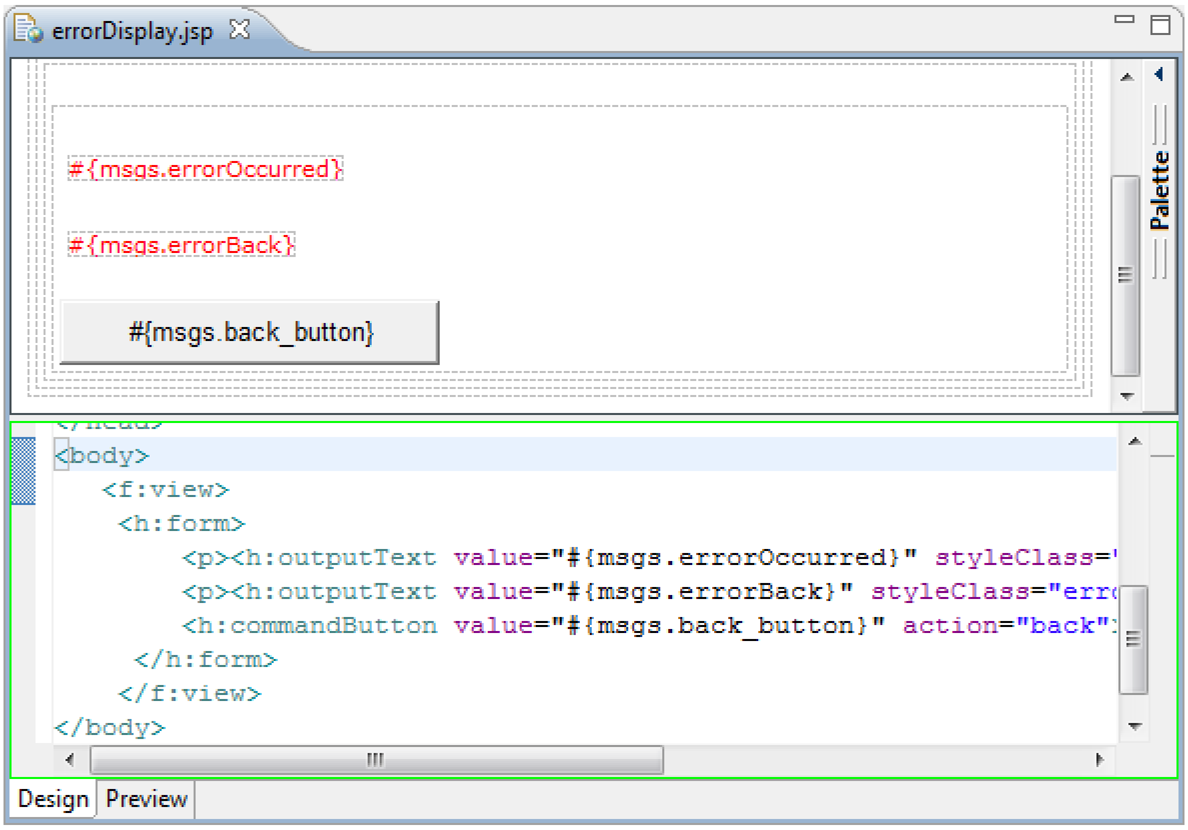
```
<head>
    <link href="styles.css" rel="stylesheet" type="text/css"/>
    …
</head>
```

16. The following table contains the hierarchy of the UI elements contained in the *index* view:

Property	Value
***ViewRoot* UI element**	
Form* UI element in the UI-element *ViewRoot	
OutputText* UI element in the UI-element *Form	

value	#{msgs.errorOccurred}
styleClass	errorMessage
OutputText* UI element in the UI-element *Form	
value	#{msgs.errorBack}
styleClass	errorMessage
CommandButton UI element in the UI-element *Form*	
value	#{msgs.back_button}
action	back

17. Result of errorDisplay.jsp



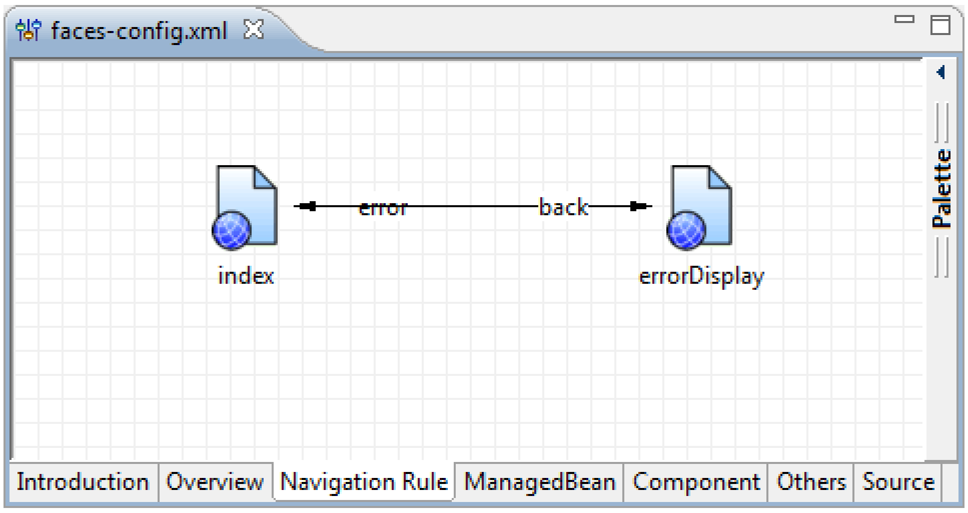
```
<body>
    <f:view>
     <h:form>
         <p><h:outputText value="#{msgs.errorOccurred}" styleClass='
         <p><h:outputText value="#{msgs.errorBack}" styleClass="err
         <h:commandButton value="#{msgs.back_button}" action="back"
      </h:form>
     </f:view>
</body>
```

18. To complete the JSF application, we need to specify the navigation rules, drill into the Web Module project, in the *WebContent* → *WEB-INF* folder and open the *faces-config.xml* file.

19. Go to the Navigation Rule tab to define the navigation. The navigation flow should look like the following image:

Note

The *CommandButton* UI element in the *index.jsp* view has the *search* method in the *action* property. If it returns the “error” string, the application will navigate to the *errorDisplay.jsp* view.



20. Select the *Source* tab. The following XML code should be added automatically between the <faces-config … > </faces-config> tags

```
<navigation-rule>
   <display-name>index</display-name>
   <from-view-id>/index.jsp</from-view-id>
   <navigation-case>
      <from-outcome>error</from-outcome>
      <to-view-id>/errorDisplay.jsp</to-view-id>
      <redirect />
   </navigation-case>
</navigation-rule>
<navigation-rule>
   <display-name>errorDisplay</display-name>
   <from-view-id>/errorDisplay.jsp</from-view-id>
   <navigation-case>
      <from-outcome>back</from-outcome>
      <to-view-id>/index.jsp</to-view-id>
   </navigation-case>
</navigation-rule>
```

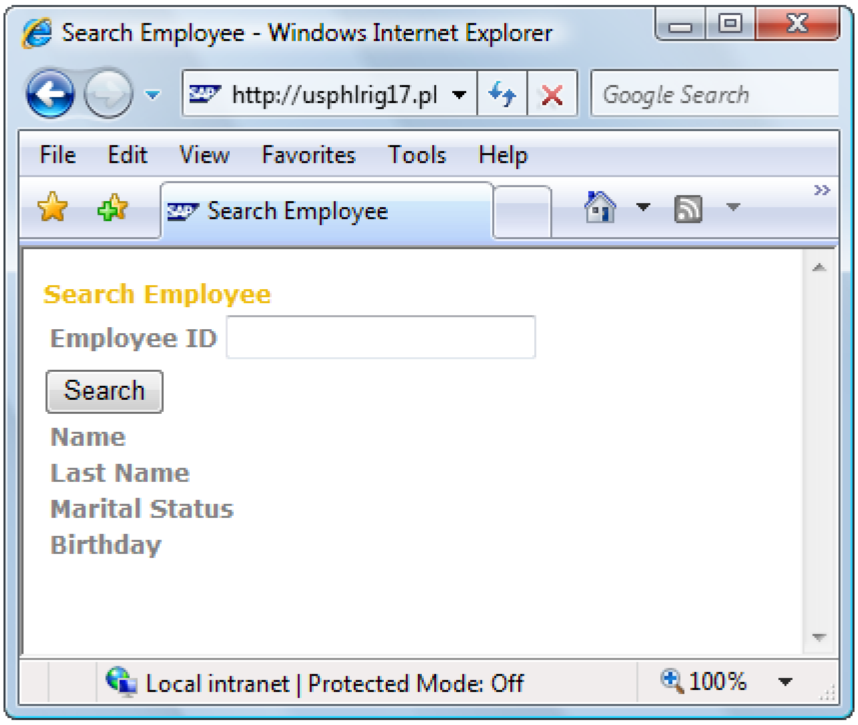
21. Save the changes you made

4.8 Build, Deploy and Run your application

1. Create the application.xml deployment descriptor, sets the WAR file to "demo.sap.com~esjsf~web.war" and the context root to "esjsf" as indicated in the Hello World JSF tutorial (Create a Hello World Application using JavaServer Faces [Extern]).
2. Save changes.
3. Build and deploy the application.
4. Run the application using the following simplified URL:

http://<servername>:<httpport>/esjsf/faces/index.jsp

5. Results:


Search Employee - Windows Internet Explorer
http://usphlrig17.pl
Google Search
File Edit View Favorites Tools Help
Search Employee
Local intranet | Protected Mode: Off
100%

Search Employee

Employee ID

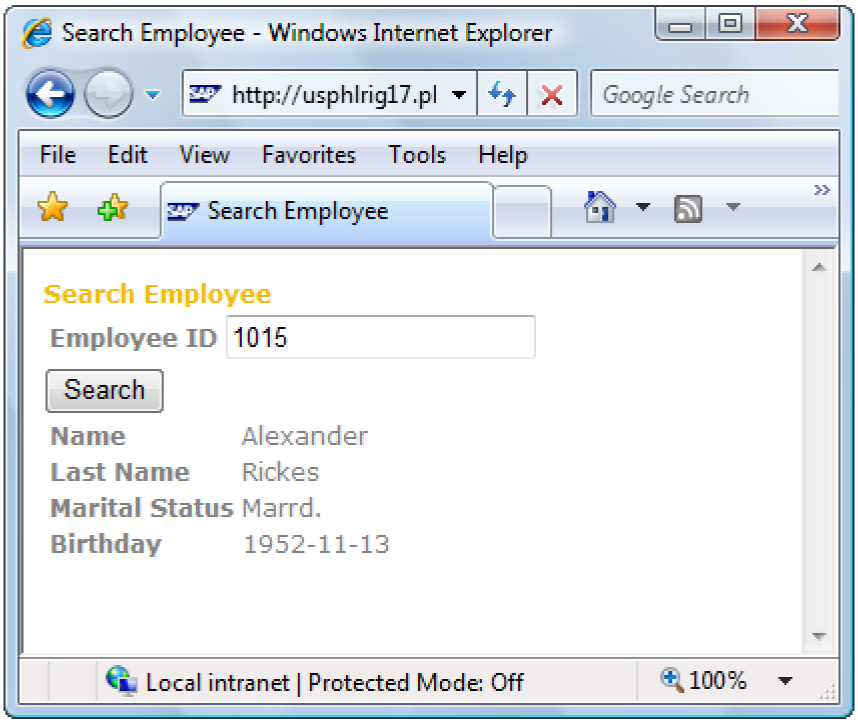
Search

Name

Last Name

Marital Status

Birthday

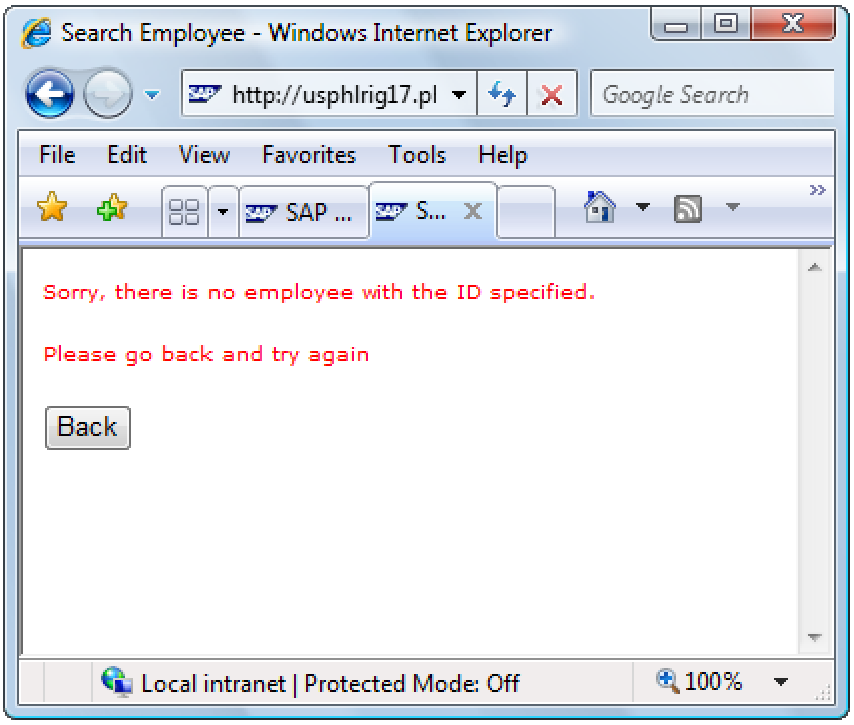

Search Employee - Windows Internet Explorer
http://usphlrig17.pl
Google Search
File Edit View Favorites Tools Help
Search Employee
Local intranet | Protected Mode: Off
100%

Search Employee

Employee ID	1015

Search

Name	Alexander
Last Name	Rickes
Marital Status	Marrd.
Birthday	1952-11-13


Search Employee - Windows Internet Explorer
http://usphlrig17.pl
Google Search
File Edit View Favorites Tools Help
SAP ...
S...
Local intranet | Protected Mode: Off
100%

Sorry, there is no employee with the ID specified.

Please go back and try again

Back