Developing an HR Homepage Using Web Dynpro, Part II

SAP DEVELOPER NETWORK

SDN Community Contribution

(This is not an official SAP document.)

Developing an HR Homepage Using Web Dynpro, Part II

SAP DEVELOPER NETWORK

Applies To:

SAP Technology, SAP Web Application Server, Web Dynpro

Summary

This tutorial describes how to design, implement, deploy, and run basic human resources (HR) applications – using Web Dynpro – that access persistent data from a remote SAP backend system. The HR application, an Employee Self-Service (ESS), allows employees in a company to create, display, and change their own data. As a result, employees in the HR department can concentrate on tasks of greater strategic importance.

By: V. Ramakrishna

Titles: Associate Consultant (Wipro Technologies Ltd, Bangalore, India)

Reviewed By: UmaSankar Subramanian, Solution Architect, SAP NetWeaver Competence Group (NWCG), Wipro Technologies Ltd, Bangalore, India

Company: Wipro Technologies

Date: 09 June 2005

Table of Contents

About this Tutorial

This tutorial describes a step-by-step procedure to design, implement, deploy, and run a basic HR application using Web Dynpro that accesses persistent data from a remote SAP backend system. To read the personal data, the application makes use of existing and custom functions in the form of readable BAPIs. The Web Dynpro framework generates a corresponding Java proxy class for each BAPI that is needed. Different Web Dynpro UI elements are used in this example.

Part I describes the prerequisites for creating the project (HR application), component, and layout; part II explains how to access the backend SAP R/3 system, states the BAPIs we are using, and shows what a typical HR application looks like.

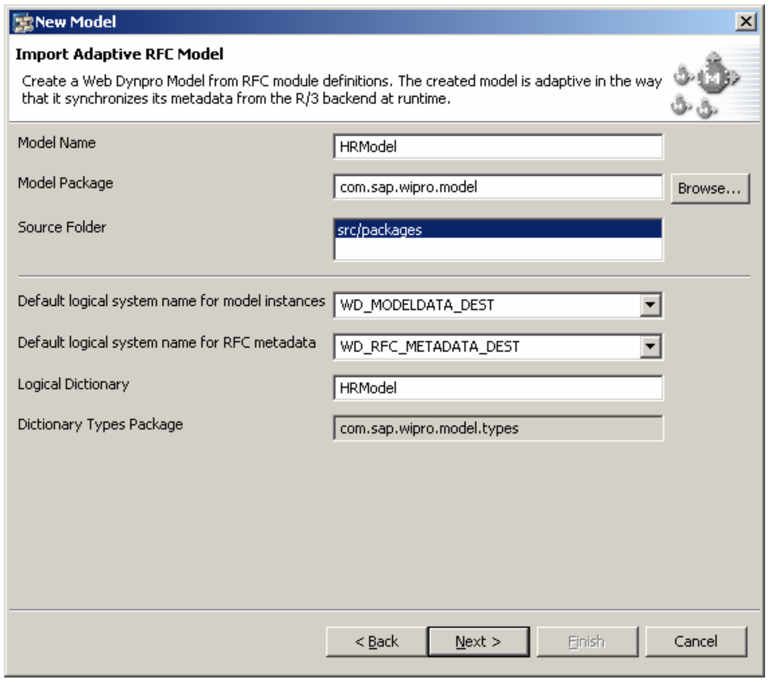
How to Read Personal Data

After successfully creating the project (HR application), component, and layout, as described in Developing an HR Homepage Using Web Dynpro, Part I, follow these steps to access personal SAP R/3 backend data.

Creating a Model

Procedure

1. In the project structure, expand the node Web Dynpro → Models.
2. From the context menu, choose Create Model. The appropriate wizard appears.
3. Choose the Import Adaptive RFC Model option, followed by Next.
4. Enter the model name **`HRModel`** and the package name **`com.sap.wipro.model`**
5. When importing an adaptive RFC model, you have to specify the logical system names for model instances and RFC metadata:
   a. Default logical system name for model instances: **`WD_MODELDATA_DEST`**
   b. Default logical system name for RFC metadata: **`WD_RFC_METADATA_DEST`**



Import Adaptive RFC Model

Create a Web Dynpro Model from RFC module definitions. The created model is adaptive in the way that it synchronizes its metadata from the R/3 backend at runtime.

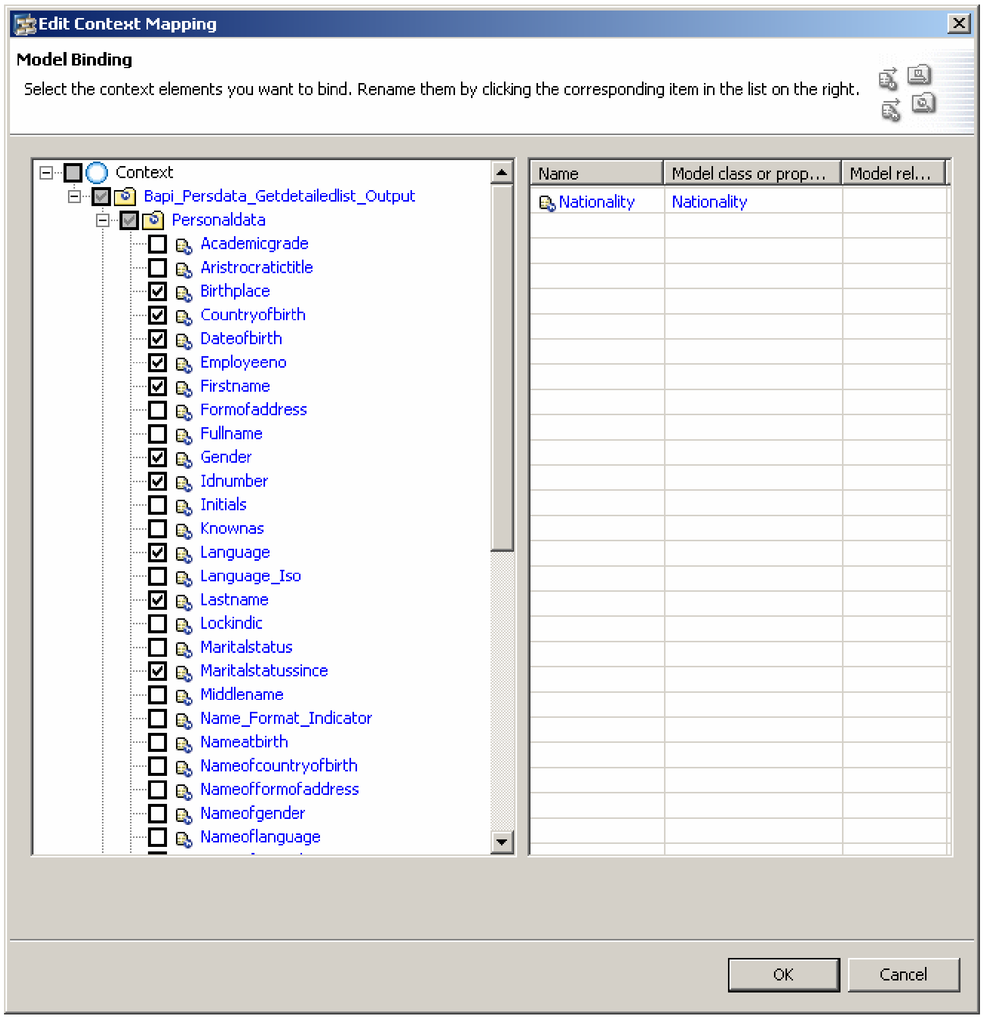
Model Name	HRModel
Model Package	com.sap.wipro.model
Source Folder	src/packages
Default logical system name for model instances	WD_MODELDATA_DEST
Default logical system name for RFC metadata	WD_RFC_METADATA_DEST
Logical Dictionary	HRModel
Dictionary Types Package	com.sap.wipro.model.types

6. Accept the suggested values and choose Next.

7. Enter the appropriate data for logging onto the SAP system and choose Next. When logging on, you can choose one of two options: Either choose a single application server or address the system and log on using *Load Balancing*.

8. Enter either the complete name of the function module - BAPI_PERSDATA_GETDETAILEDLIST in the appropriate field, or enter the start of the name followed by an asterisk (*). Then choose Search.

9. Select the function module BAPI_PERSDATA_GETDETAILEDLIST from the list that appears.



Edit Context Mapping

Model Binding

Select the context elements you want to bind. Rename them by clicking the corresponding item in the list on the right.

```
Context
  Bapi_Persdata_Getdetailedlist_Output
    Personaldata
      [ ] Academicgrade
      [ ] Aristrocratictitle
      [✓] Birthplace
      [✓] Countryofbirth
      [✓] Dateofbirth
      [✓] Employeeno
      [✓] Firstname
      [ ] Formofaddress
      [ ] Fullname
      [✓] Gender
      [✓] Idnumber
      [ ] Initials
      [ ] Knownas
      [✓] Language
      [ ] Language_Iso
      [✓] Lastname
      [ ] Lockindic
      [ ] Maritalstatus
      [✓] Maritalstatussince
      [ ] Middlename
      [ ] Name_Format_Indicator
      [ ] Nameatbirth
      [ ] Nameofcountryofbirth
      [ ] Nameofformofaddress
      [ ] Nameofgender
      [ ] Nameoflanguage
```

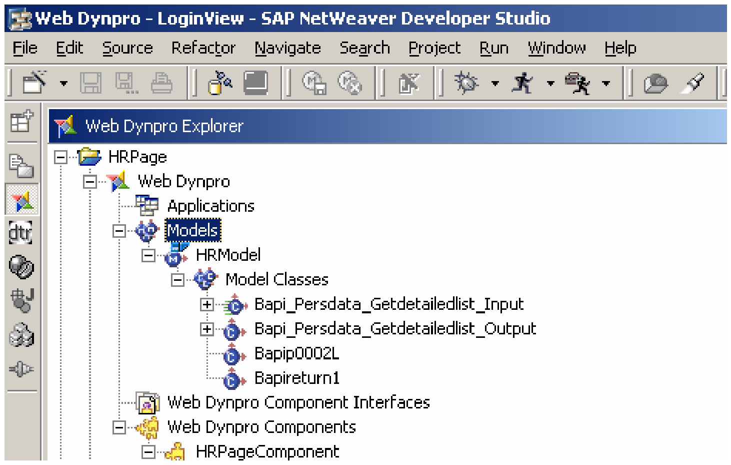
Name	Model class or prop...	Model rel...
Nationality	Nationality	

9. Choose Next. By doing so, you automatically trigger the generation process. The import process is logged by a detailed description, which you can see in the next dialogue.

10. Choose Finish.

Result

The Java proxies are generated and a new model node *`BAPI_PERSDATA_GETDETAILEDLIST`* is inserted into the project structure.


Web Dynpro - LoginView - SAP NetWeaver Developer Studio
File Edit Source Refactor Navigate Search Project Run Window Help
Web Dynpro Explorer
HRPage
Web Dynpro
Applications
Models
HRModel
Model Classes
Bapi_Persdata_Getdetailedlist_Input
Bapi_Persdata_Getdetailedlist_Output
Bapip0002L
Bapireturn1
Web Dynpro Component Interfaces
Web Dynpro Components
HRPageComponent

The newly-created adaptive RFC model *`BAPI_PERSDATA_GETDETAILEDLIST`* can be used now in any component in the current project. Similarly, Addressmodel and Bankmodel have to be created for the custom BAPIs, *`YZ_BAPI_ADDRESSEMP_GETDETAIL, YZ_BAPI_BANK_DETAIL.`*

Creating a Custom Controller Context and Binding it to the Model

The custom controller *HRPageCust* is responsible for retrieving flight data from an SAP system, so it needs to be able to map the corresponding input and output for the flight model. To establish this correspondence between the custom controller and the model, you will create an appropriate controller context and then bind the context nodes to the model structure. In this way, you can ensure that the model data is stored and manipulated in a central location.

Adding a Model to the Web Dynpro Component

1. In the project structure, expand the tree up to the node *Web Dynpro* → *Web Dynpro Components* → *HRPageCust*.

2. Select the node *Used Models* and open the context menu.

3. Choose *Add*.

4. In the list that appears, select the model *`HRModel`* and confirm by choosing *OK*. By doing so, you specify that all views and controllers of *HRPageComponent* have a dependent relationship with the model *`HRModel`*.

Creating a Context for the Custom Controller

1. In the project structure, double-click the name of the custom controller (in this case, **HRPageCust**).

2. Choose the *Context* tab if necessary.

3. Open the context menu for the root node *Context* and choose the option *New* → *Model Node*.

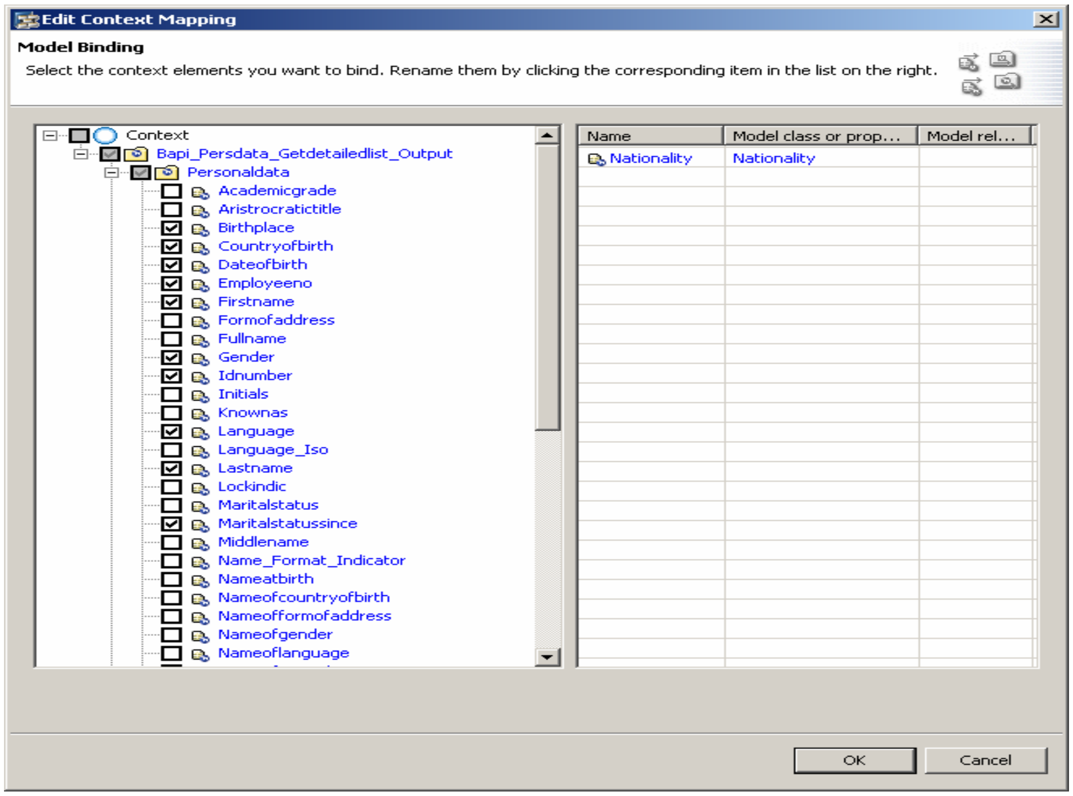
4. Enter the name *`BAPI_PERSDATA_GETDETAILEDLIST _OUTPUT`* for the model node and choose *Finish*.

5. From the context menu for the model node that you have just created, choose *Edit Model Binding*….

6. Choose the model class *`BAPI_PERSDATA_GETDETAILEDLIST _OUTPUT`*, followed by *Next*.

7. Activate the following entries:

   `Birthplae,countryofbirth,dateofbirth,employeeno,firstname,gender,language, lastname,martialstatussince,bloodgroup,nationality` `etc`. Choose *Finish*.



Edit Context Mapping

Model Binding

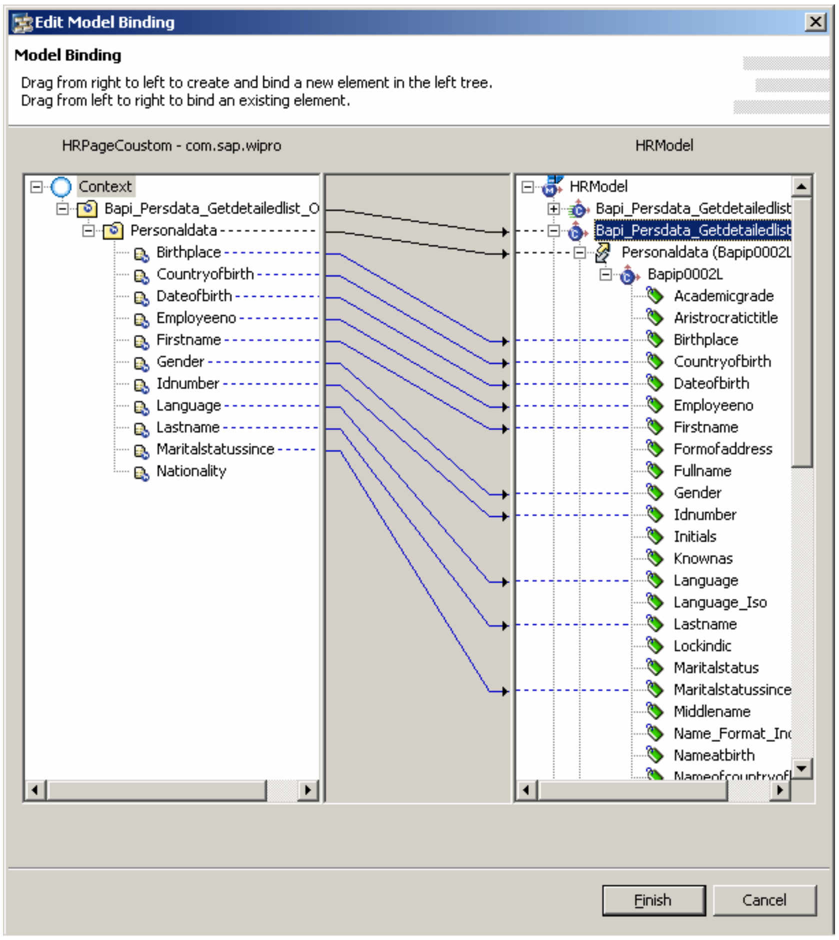
Select the context elements you want to bind. Rename them by clicking the corresponding item in the list on the right.

- ☐ Context
  - ☑ Bapi_Persdata_Getdetailedlist_Output
    - ☑ Personaldata
      - ☐ Academicgrade
      - ☐ Aristrocratictitle
      - ☑ Birthplace
      - ☑ Countryofbirth
      - ☑ Dateofbirth
      - ☑ Employeeno
      - ☑ Firstname
      - ☐ Formofaddress
      - ☐ Fullname
      - ☑ Gender
      - ☑ Idnumber
      - ☐ Initials
      - ☐ Knownas
      - ☑ Language
      - ☐ Language_Iso
      - ☑ Lastname
      - ☐ Lockindic
      - ☐ Maritalstatus
      - ☑ Maritalstatussince
      - ☐ Middlename
      - ☐ Name_Format_Indicator
      - ☐ Nameatbirth
      - ☐ Nameofcountryofbirth
      - ☐ Nameofformofaddress
      - ☐ Nameofgender
      - ☐ Nameoflanguage

Name	Model class or prop…	Model rel…
Nationality	Nationality	

The Developer Studio refreshes the context tree appropriately. In this way, you have complete specification of all the context nodes for output data.

8. Save your work by choosing the icon (*Save All Metadata)* from the toolbar.



Thus, with the model definition as a starting point, we have created a context for the custom controller *HRPageCust* and bound the appropriate context nodes for the input and output structures to the corresponding model nodes. A similar procedure is followed for models Addressmodel and Bankmodel; these have to be created for the custom BAPIs, *`YZ_BAPI_ADDRESSEMP_GETDETAIL, YZ_BAPI_BANK_DETAIL`* and the attributes activated.

Mapping View Context Elements to Custom Context Elements

In this step, we will map context elements of the view *LoginView* to the appropriate context elements of the custom controller *HRPageCust.*

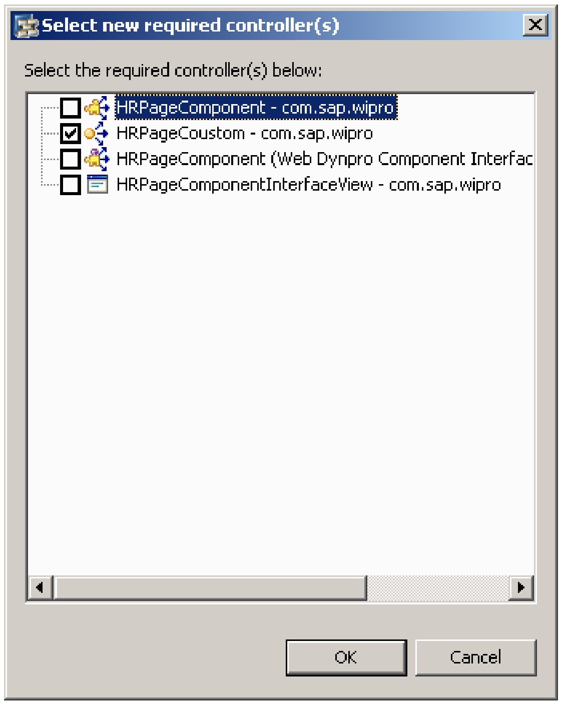
Adding Dependencies to the Views

In the project structure, double-click the node for the *LoginView* (*Web Dynpro* → *Web* Dynpro Components → HRPageCust → Views → LoginView). The View Designer for the LoginView appears.

Choose the *Properties* tab.

Under Required Controllers, choose Add.

In the list that appears, choose the **`HRPageCust`** component



Select the required controller(s) below:

- [ ] HRPageComponent - com.sap.wipro
- [✓] HRPageCoustom - com.sap.wipro
- [ ] HRPageComponent (Web Dynpro Component Interfac
- [ ] HRPageComponentInterfaceView - com.sap.wipro

Confirm by choosing *OK*.

Creating a Context for the LoginView

Open the *View Designer* for the **`LoginView`** again.

Choose the *Context* tab.

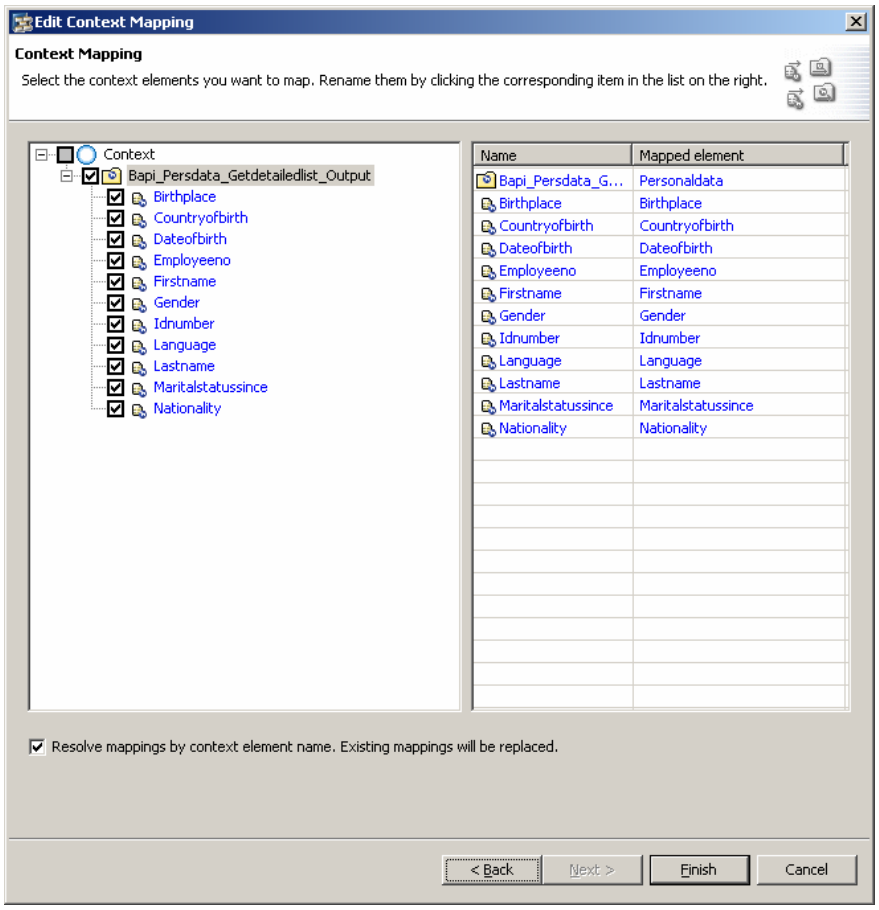
Open the context menu for the root node *Context* and choose the option *New* → *Model Node*.

Enter the name **`BAPI_PERSDATA_GETDETAILEDLIST _OUTPUT`** for the model node and choose *Finish*.

From the context menu for the model node of that name that you have created, choose *Edit Context Mapping*…

Choose the custom context node **`BAPI_PERSDATA_GETDETAILEDLIST _OUTPUT`**, followed by *Next*.

Activate only **`required attributes`** Choose *Finish*.



Context Mapping

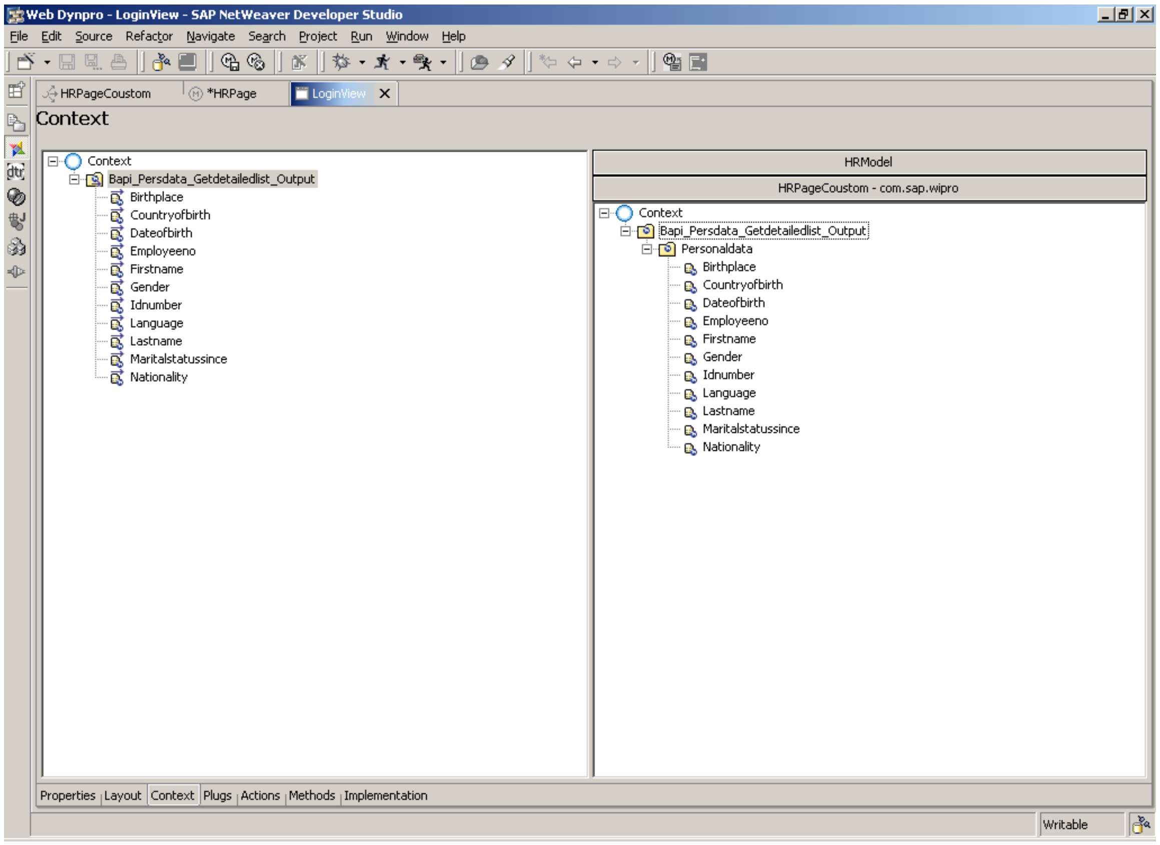
Select the context elements you want to map. Rename them by clicking the corresponding item in the list on the right.

- Context
  - ☑ Bapi_Persdata_Getdetailedlist_Output
    - ☑ Birthplace
    - ☑ Countryofbirth
    - ☑ Dateofbirth
    - ☑ Employeeno
    - ☑ Firstname
    - ☑ Gender
    - ☑ Idnumber
    - ☑ Language
    - ☑ Lastname
    - ☑ Maritalstatussince
    - ☑ Nationality

Name	Mapped element
Bapi_Persdata_G...	Personaldata
Birthplace	Birthplace
Countryofbirth	Countryofbirth
Dateofbirth	Dateofbirth
Employeeno	Employeeno
Firstname	Firstname
Gender	Gender
Idnumber	Idnumber
Language	Language
Lastname	Lastname
Maritalstatussince	Maritalstatussince
Nationality	Nationality

☑ Resolve mappings by context element name. Existing mappings will be replaced.

The Developer Studio refreshes the context tree appropriately.


Web Dynpro - LoginView - SAP NetWeaver Developer Studio
File Edit Source Refactor Navigate Search Project Run Window Help
HRPageCoustom
*HRPage
LoginView
Context
HRModel
HRPageCoustom - com.sap.wipro
Properties Layout Context Plugs Actions Methods Implementation
Writable

Context
- Bapi_Persdata_Getdetailedlist_Output
  - Birthplace
  - Countryofbirth
  - Dateofbirth
  - Employeeno
  - Firstname
  - Gender
  - Idnumber
  - Language
  - Lastname
  - Maritalstatussince
  - Nationality

Context
- Bapi_Persdata_Getdetailedlist_Output
  - Personaldata
    - Birthplace
    - Countryofbirth
    - Dateofbirth
    - Employeeno
    - Firstname
    - Gender
    - Idnumber
    - Language
    - Lastname
    - Maritalstatussince
    - Nationality

Save your work by choosing the icon (*Save All Metadata)* from the toolbar.
The above steps are repeated for the other two BAPIs.

Creating Actions and Declaring Methods

To trigger the display of the personal data from the SAP system, in the *LoginView*, you need to create an action that can be bound to a UI element, such as a button. You can then implement the event handler, which reacts to this action.

Procedure

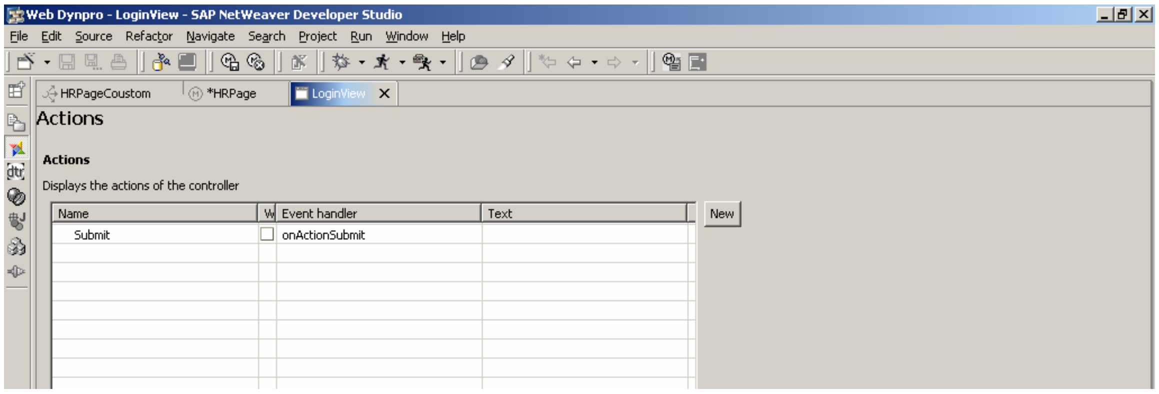
Creating the Submit Action

1. Open the *View Designer* for the **`LoginView`**.

2. Choose the *Actions* tab.

3. Choose the *New* pushbutton. You can create a new action in the wizard that appears.

4. Enter the name **`Submit`** for this new action, leave the other settings unchanged, and choose *Finish.* An event handler, **`onActionSubmit`**, is automatically created for this new action.


Web Dynpro - LoginView - SAP NetWeaver Developer Studio
File Edit Source Refactor Navigate Search Project Run Window Help
HRPageCoustom
*HRPage
LoginView
New

Actions

Actions

Displays the actions of the controller

Name	W	Event handler	Text
Submit	☐	onActionSubmit	

Declaring the executebapi_persdata_getdetailedlist _output Method

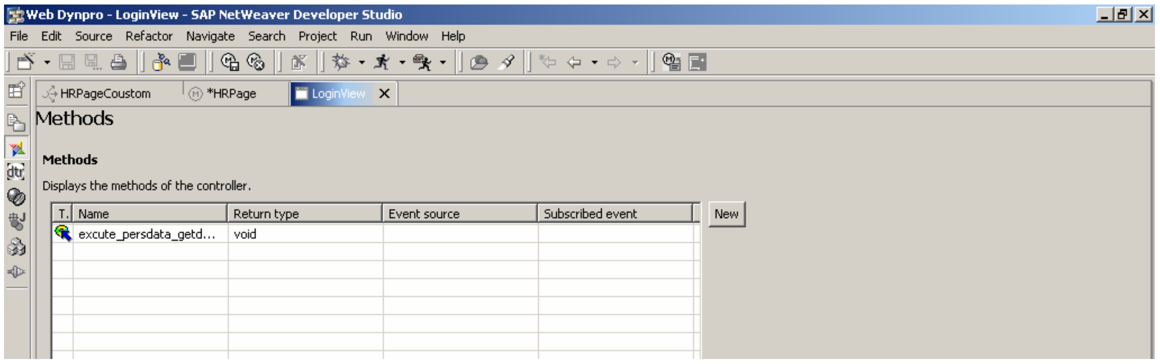
1. Open the editor for the custom controller **`HRPageCust`** again.

2. Choose the *Methods* tab.

3. Choose *New.*

4. Select the *Method* option and choose *Next*.

5. In the wizard screen that appears, enter the name **`executebapi_persdata_getdetailedlist _output`** for this new method and assign it the return type **`void`**. Choose *Finish*. The method **`executebapi_persdata_getdetailedlist _output`** is added to the custom controller.


Web Dynpro - LoginView - SAP NetWeaver Developer Studio
File Edit Source Refactor Navigate Search Project Run Window Help
HRPageCoustom
*HRPage
LoginView
New

Methods

Methods

Displays the methods of the controller.

T..	Name	Return type	Event source	Subscribed event
	excute_persdata_getd...	void		

Save the new metadata by choosing the icon (*Save All Metadata)* from the toolbar.
Follow similar steps for the other two BAPIs

Result

You have created the *Submit* action for the LoginView. In the next step, you will bind it to the appropriate button using the *Source* property.

You have also declared a new method **`executebapi_persdata_getdetailedlist _output`** for the custom controller. Later, you will use this method to implement the adaptive RFC to the SAP system.

Editing the UI Elements

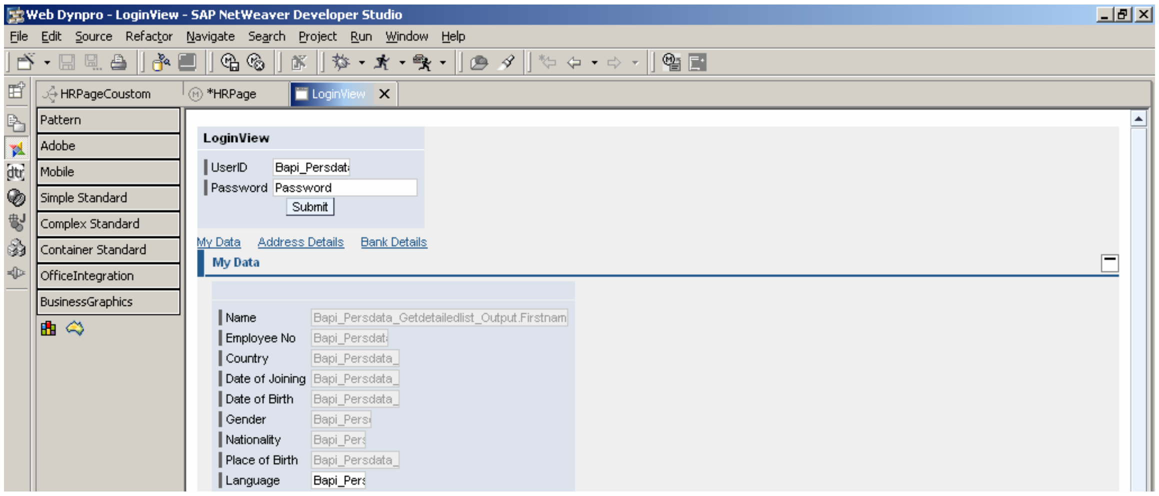
Login View

1. Open the **`LoginView`** in the *View Designer* by clicking the *Layout* tab of the View Editor. The *View Designer* displays predefined default text. Simultaneously, the *Outline* view displays a list of the UI elements included. If you select an element in the *Outline* view or on the *Layout* tab, its associated element properties are shown in the *Properties* view.

2. Choose Input filed and button UI elements that have been included in the project template and give them appropriate properties.

Example:

Property	Value
For UserIDInputField	
Value	Bapi_Persdata_Getdetailedlist_Output.Employeeno
For SubmitButton	
Event > onAction	Submit



3. Save the new metadata by choosing the icon (*Save All Metadata)* from the toolbar.

Adding the Implementation of the Backend Connection

Implementing the Action Event Handler

1. In the *View Designer,* choose the *Implementation* tab for the **`LoginView`**. The Developer Studio runs several generation routines, and then displays the updated source code for the implementation of the view controller.

2. Insert the following line of code in the onActionSubmit() method (immediately after clicking Submit, user is directed to Mydata, AddressDeatils, BankDetails links and Mydata here if click on respective link, that data will be shown screen shots has shown below):

```
public void onActionSubmit(com.sap.tc.webdynpro.progmodel.api.IWDCustomEvent
wdEvent )
  {
    //@@begin
wdThis.wdGetFlightListCustController().Bapi_Persdata_Getdetailedlist_Output
();
wdContext.currentContextElement ().setLoginVisible(WDVisibility.NONE);
wdContext.currentContextElement().setLinksVisible(WDVisibility.VISIBLE)wdCont
ext.currentContextElement().setMyDataVisible(WDVisibility.VISIBLE);
wdContext.currentContextElement().setAddressVisible(WDVisibility.NONE);
wdContext.currentContextElement().setBankDetailsVisible(WDVisibility.NONE);

   //@@end
  }
```

Adding the Implementation for the Custom Controller HRPageCust

1. Open the Controller Editor for the custom controller **`HRPageCust`** again.
2. Choose the *Implementation* tab.
3. In the standard method **`wdDoInit()`**, between `//@@begin wdDoInit()` and `//@@end`, add the following lines of code:

```
public void wdDoInit()
  {
//@@begin
// Create a new element in the Bapi Bapi_Persdata_Getdetailedlist_Output node
Bapi_Persdata_Getdetailedlist_Output = new
Bapi_Persdata_Getdetailedlist_Output ();
yz_bapi_addressemp_getdetail_output = new yz_bapi_addressemp_getdetail_output
();
yz_bapi_bank_detail_output = new yz_bapi_bank_detail_output ();
wdContext.nodeBapi_Persdata_Getdetailedlist_Output().bind(input);
wdContext.nodeyz_bapi_addressemp_getdetail_output ().bind(input);
wdContext.nodeyz_bapi_bank_detail_output ().bind(input);
 //@@end
  }
```

4. In the method executeBapi_Persdata_Getdetailedlist_Output()between //@@begin executeBapi_Persdata_Getdetailedlist_Output ()and //@@end, add the following lines of code:

```
/** declared method */
 public void executeBapi_Persdata_Getdetailedlist_Output(){
    //@@begin
   try
    {
      // Calls remote function module BAPI_PERSDATA_GETDETAILEDLIST
wdContext.current
Bapi_Persdata_Getdetailedlist_OutputElement().modelObject().execute();
     // Synchronise the data in the context with the data in the model
       wdContext.nodeOutput().invalidate();
    }
    catch (Exception ex)
    {
      // If an exception is thrown, then the stack trace will be printed
      ex.printStackTrace();
    }
    //@@end
  }
```

5. Save the new metadata by choosing the icon (*Save All Metadata)* from the toolbar.

Steps 4-5 are repeated for excuteyz_bapi_addressemp_getdetail_output and excuteyz_bapi_bank_detail_output methods.

Building, Deploying, and Running the Application

Building the Project

1. Save the metadata for your project in its current state.

2. In the *Web Dynpro Explorer*, from the context menu of the project node **`HRPage`**, choose Application Create Application (HRPageApp).

Deploying the Project

1. In the *Web Dynpro Explorer*, expand the project node **`HRPage`** and choose select the application HRPageApp
2. Choose *Deploy New Archive and Run*.

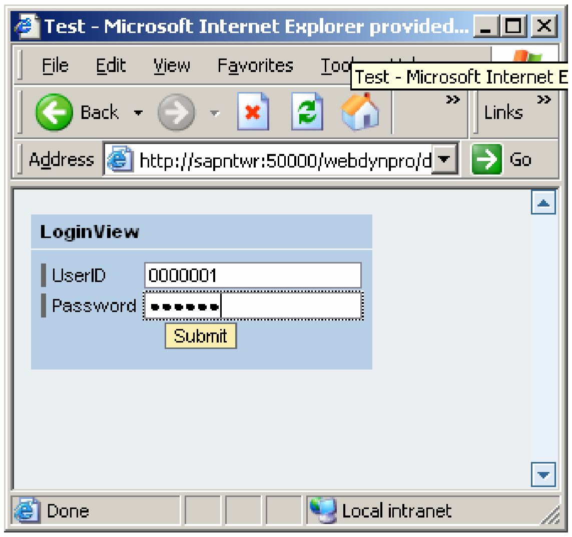
Result

The Developer Studio performs the deployment process and then automatically launches the HRPage application in the web browser.

Test the Web Dynpro application by entering a valid EmployeeNo and PassWord for the *UserID and Password* and then clicking the *Submit* button. Provided the system contains the appropriate flight data, it will display it in the browser as shown below.

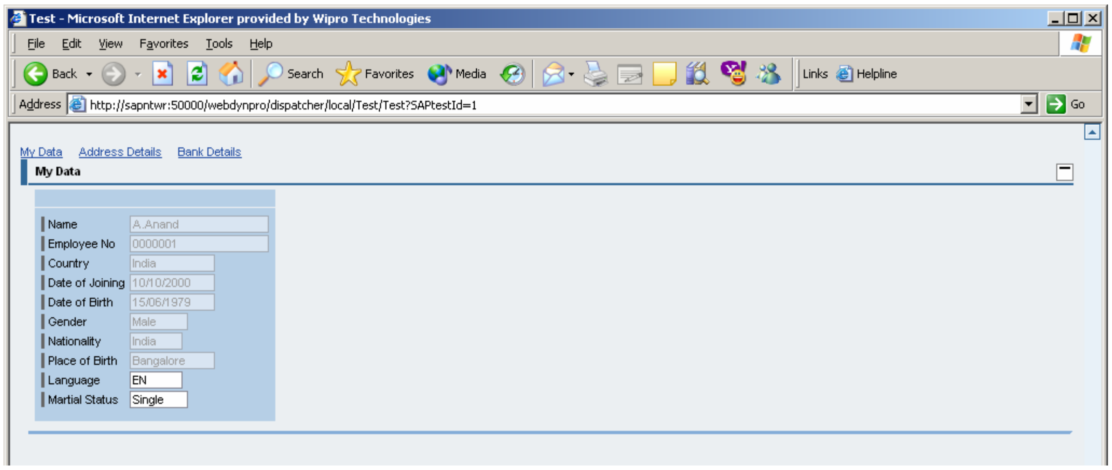
LoginView

User enters valid EmployeeID and Password he will be directed to Mydata, AdressDetails, Bankdetails, Links along with Mydata.


Test - Microsoft Internet Explorer provided...
File
Edit
View
Favorites
Back
Links
Address
http://sapntwr:50000/webdynpro/d
Go
LoginView
Submit
Done
Local intranet

UserID	0000001
Password	••••••

Mydata: Employee Details


Test - Microsoft Internet Explorer provided by Wipro Technologies

Address: http://sapntwr:50000/webdynpro/dispatcher/local/Test/Test?SAPtestId=1

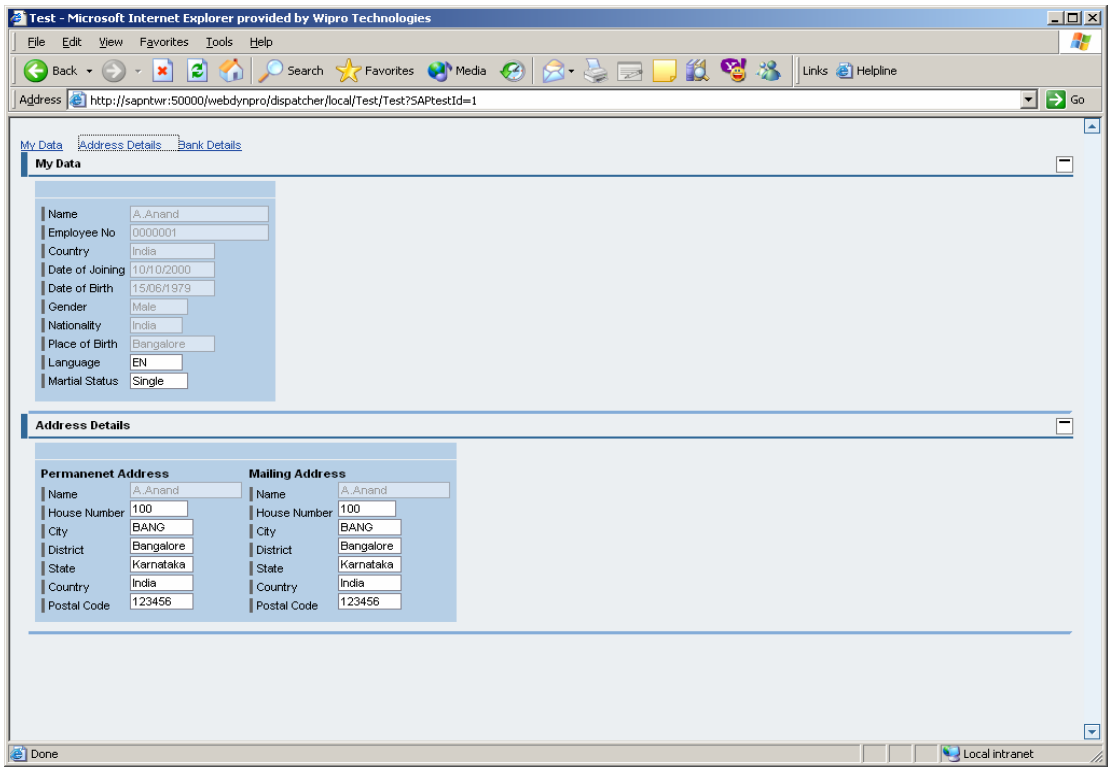
My Data Address Details Bank Details

My Data

Name	A.Anand
Employee No	0000001
Country	India
Date of Joining	10/10/2000
Date of Birth	15/06/1979
Gender	Male
Nationality	India
Place of Birth	Bangalore
Language	EN
Martial Status	Single

Address Details

On clicking the Addressdetails link:


Test - Microsoft Internet Explorer provided by Wipro Technologies

Address: http://sapntwr:50000/webdynpro/dispatcher/local/Test/Test?SAPtestId=1

My Data Address Details Bank Details

My Data

Name	A.Anand
Employee No	0000001
Country	India
Date of Joining	10/10/2000
Date of Birth	15/06/1979
Gender	Male
Nationality	India
Place of Birth	Bangalore
Language	EN
Martial Status	Single

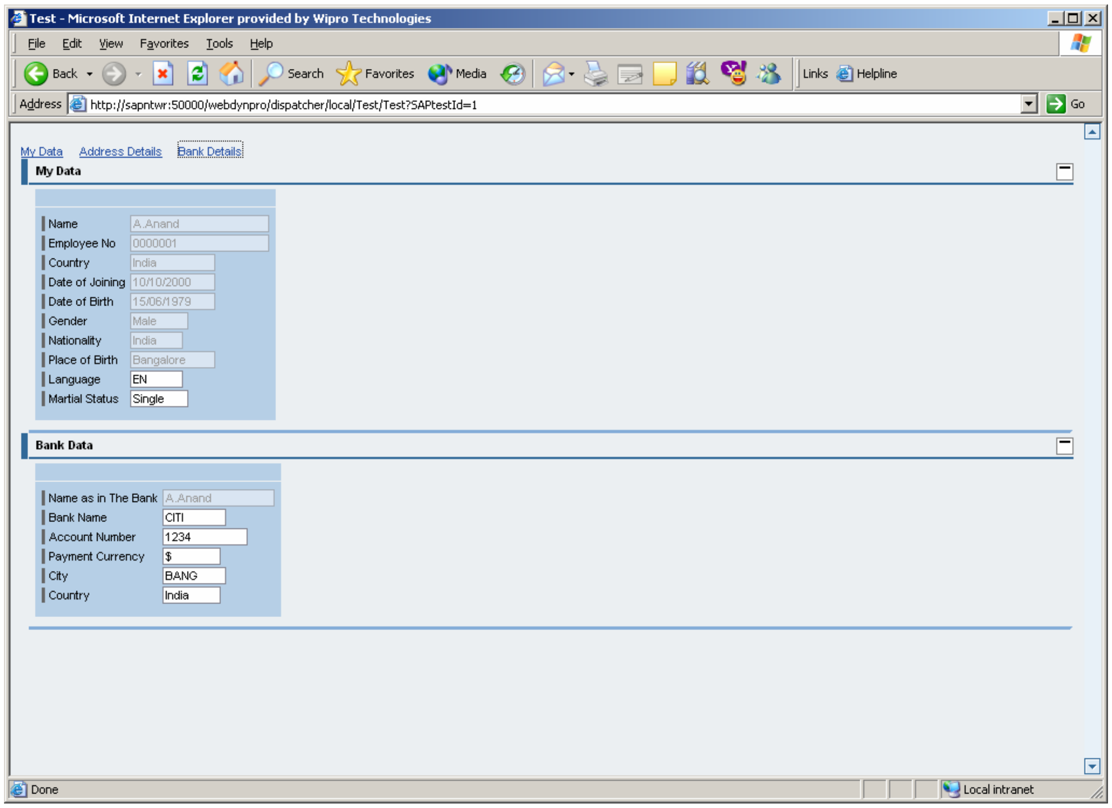
Address Details

Permanenet Address		Mailing Address	
Name	A.Anand	Name	A.Anand
House Number	100	House Number	100
City	BANG	City	BANG
District	Bangalore	District	Bangalore
State	Karnataka	State	Karnataka
Country	India	Country	India
Postal Code	123456	Postal Code	123456

Done Local intranet

Bank Details

On clicking Bankdetails link:


Test - Microsoft Internet Explorer provided by Wipro Technologies
File Edit View Favorites Tools Help
Back Search Favorites Media Links Helpline
Address http://sapntwr:50000/webdynpro/dispatcher/local/Test/Test?SAPtestId=1
Go
My Data Address Details Bank Details
Done
Local intranet

My Data

Name	A.Anand
Employee No	0000001
Country	India
Date of Joining	10/10/2000
Date of Birth	15/06/1979
Gender	Male
Nationality	India
Place of Birth	Bangalore
Language	EN
Martial Status	Single

Bank Data

Name as in The Bank	A.Anand
Bank Name	CITI
Account Number	1234
Payment Currency	$
City	BANG
Country	India

Conclusion

This tutorial describes how to design, implement, deploy, and run a basic HR application using Web Dynpro that accesses persistent data from a remote SAP backend system. The HR application lets employees in a company create, display, and change their own data. This is an alternative to standardized HR processes available in Employee Self-Service (ESS) business packages for backend SAP R/3 systems <= 4.6c.

Rather than waiting for the backend SAP R/3 system to be upgraded to the latest release to utilize the latest ESS business package (based on Web Dynpro), this provides an alternative approach to customers to get the benefits of Web Dynpro. Web Dynpro-based applications provide a better look and feel since there is no ITS involved. Web Dynpro iViews show better performance. Based on the complexity of the HR processes and number of applications, the effort for the Web Dynpro development varies. As a result, employees in the Human Resources Department can concentrate on other tasks of greater strategic importance.

Author Bio

V. Ramakrishna has been working as an associate consultant for SAP NetWeaver Competence Group (NWCG) at Wipro Technologies, Bangalore, India since September 2004. He has five years of IT experience since completing his PG in structural engineering from IIT Madras in February 2000. His areas of interest are cutting edge technologies, such as web application servers, Web Dynpro integration, JDI, Visual Composer, Solution Manager, Enterprise Portal, and J2EE with SAP NetWeaver Developer Studio.