Integration of EJB 3.0 in Web Dynpro Using Enterprise JavaBean Model

Applies to:

SAP NetWeaver 7.1, Web Dynpro for java, JEE
For more information, visit the User Interface Technology homepage.

Summary

The EJB 3.0 is standard and powerful Java Enterprise Edition (EE) technology for business components. That includes JPA, JTA and session bean 3.0, message bean 3.0.

JPA is used to generate entities from database tables.

In this article we are going to see how to implement a simple Web Dynpro application that access database using EJB 3.0.

Here we will use JPA to create entities having structure similar to the database table.

Author: Snehal Kendre

Company: L&T Infotech

Created on: 30 September 2008

Author Bio

Snehal Kendre is an Web Dynpro consultant at L&T Infotech.

Table of Contents

Getting started

Here we are going to create one java dictionary project, one EJB project, one EAR project, and one web dynpro application.

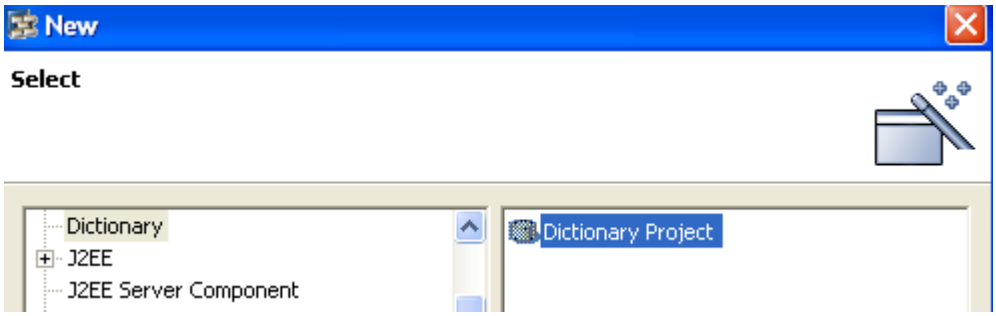
Scenario

Application will maintain employee information in database and provide basic functionality to view and Insert data in database.

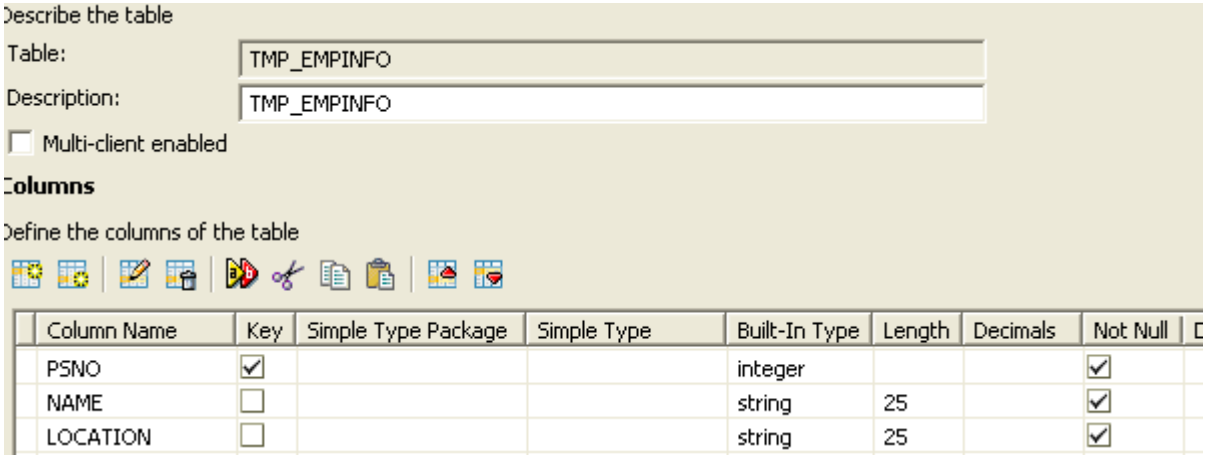
Creating a java dictionary

First step is creation of java dictionary **jd_EmployeeInfo.**

Open Dictionary Perspective.



Create a table **TMP_EMPINFO**

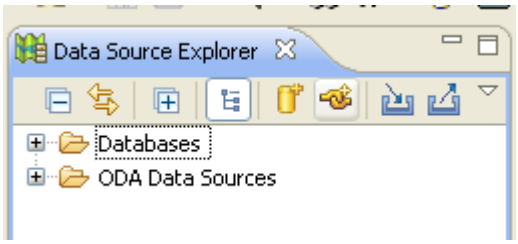


Create archive of dictionary project and deploy it on server.

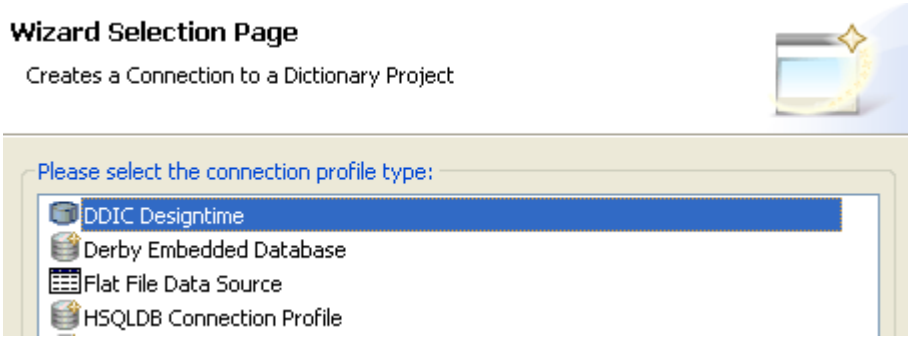
Creating a New Connection Profile

Before creating an EJB project we need to create a Connection to a Dictionary Project so that we can create JPA entities.

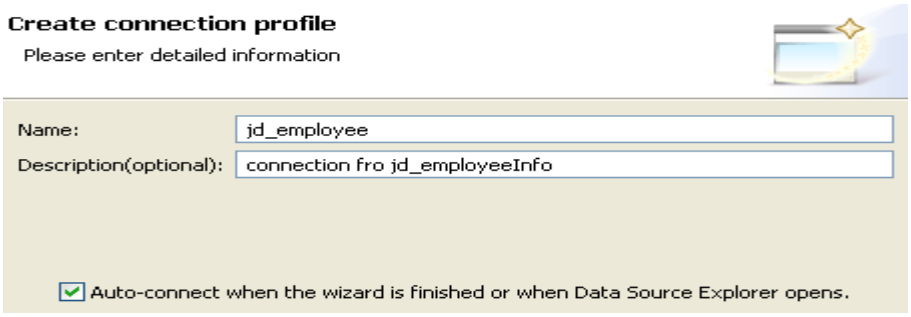
Open Database Development perspective



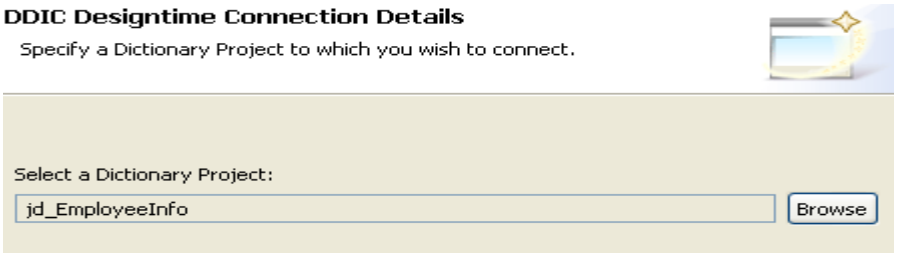
To create a new connection profile for java dictionary, choose *DDIC Designtime option.*



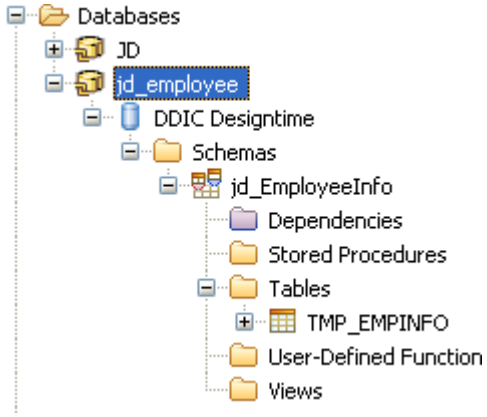
Create a new connection profile **jd_employee**. Check the auto-connect option.



Select java dictionary

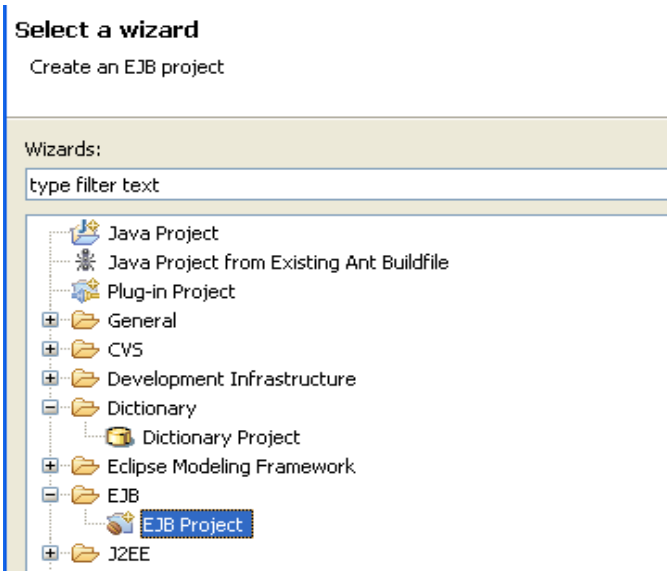


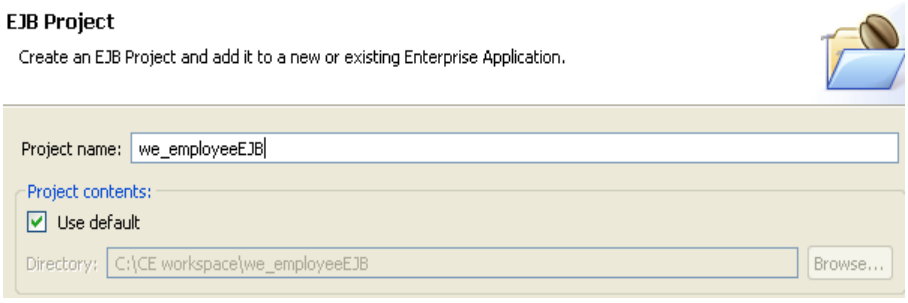
A new connection profile will be created for jd_employeeInfo



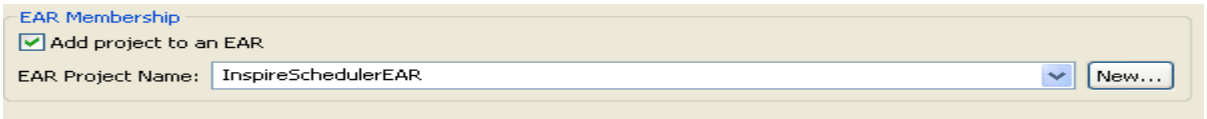
Creating an EJB Project

Open JEE perspective. Create a new EJB project as **we_employeeEJB**

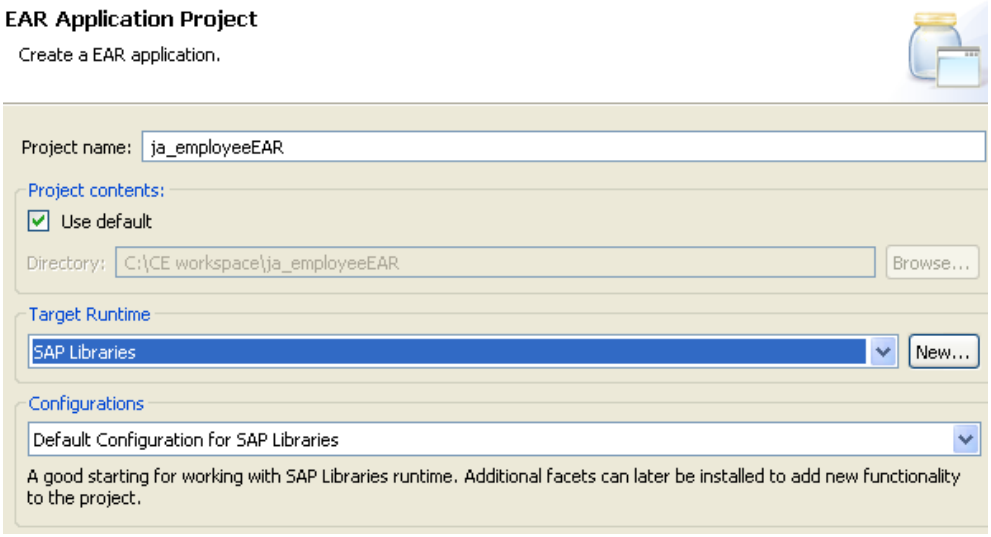




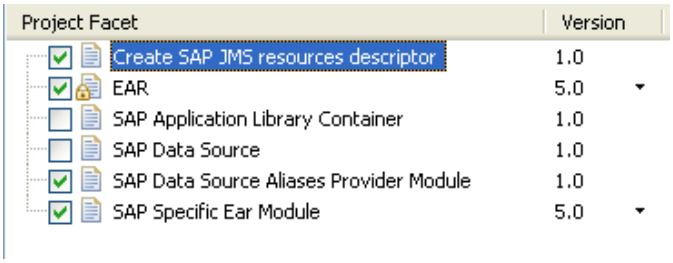
Select add to new EAR Application project.



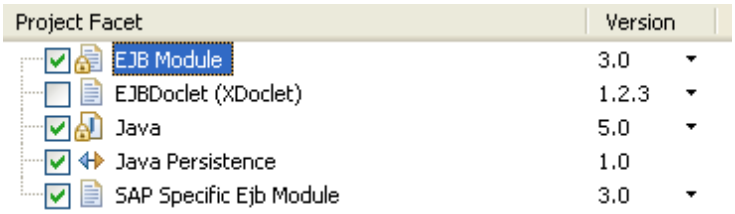
Create a new EAR Application project as **ja_employeeEAR**



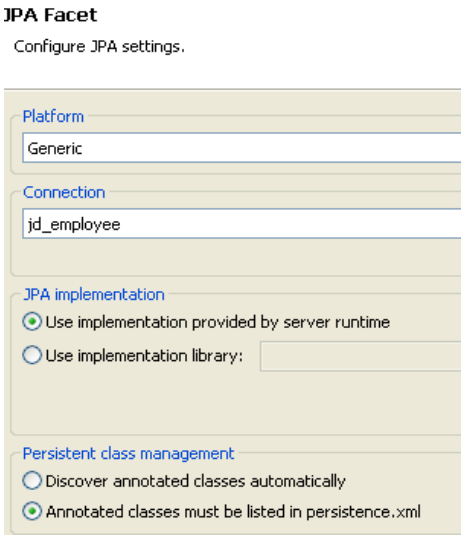
Select the configuration as below



After creating an EAR application project choose following project facates for EJB project.



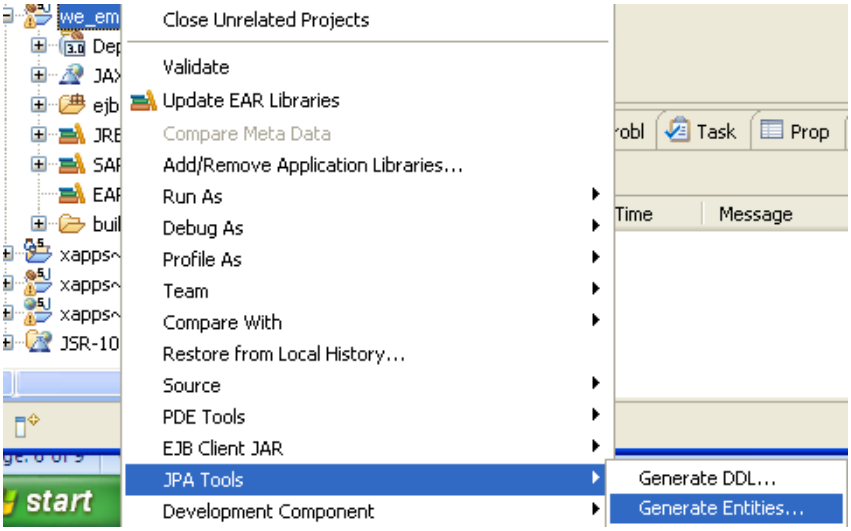
Configure the JPA settings as below and choose the **jd_employee** database connection



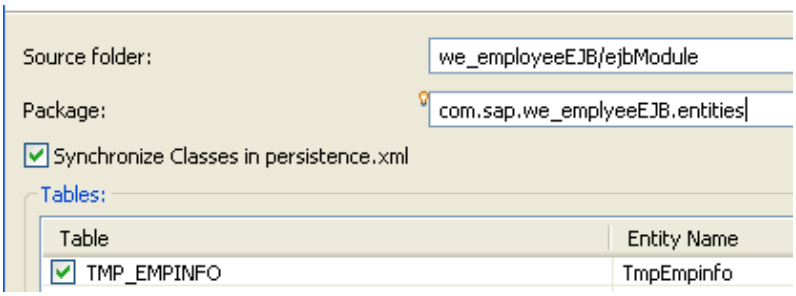
Now we are ready to implement the project.

Create JPA Entities

First we need to create JPA entities to access database tables.



Create the entity **TmpEmpinfo**



Specify the entities in **persistence.xml** file as below

Give the jta-data-source and location of JPA entity classes.

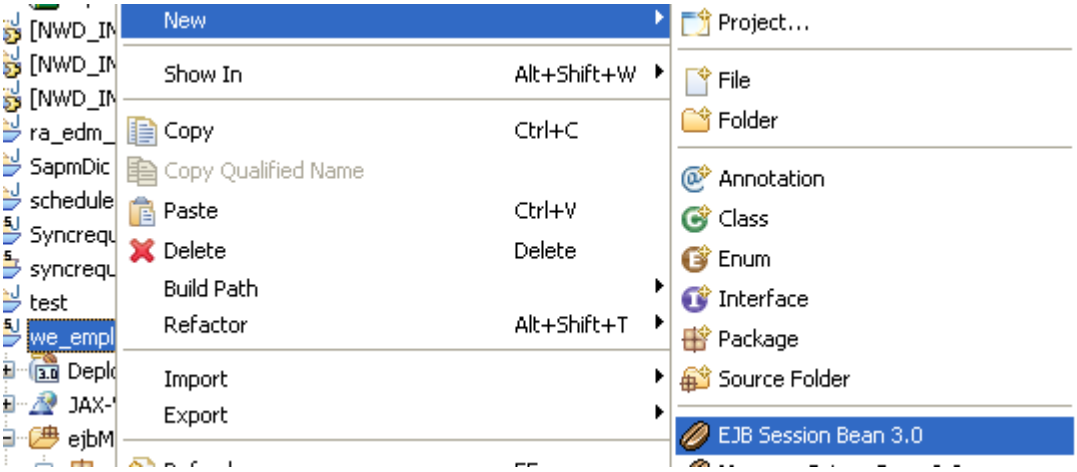
```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="1.0" xmlns="http://java.sun.com/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd">
   <persistence-unit name="we_employeeEJB">
          <jta-data-source>TMP_DB</jta-data-source>
          <class>com.sap.we_emplyeeEJB.entities.TmpEmpinfo</class>
   </persistence-unit>
</persistence>
```

Add NamedNativequery to fetch information from database

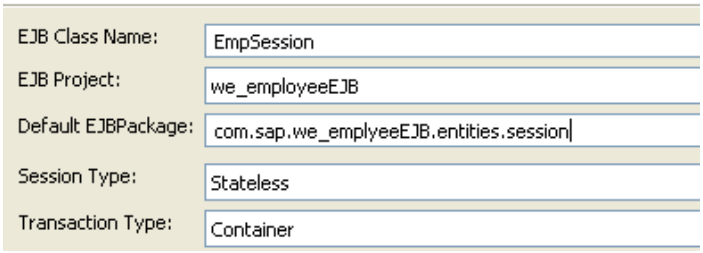
@NamedNativeQueries(value={@NamedNativeQuery(name="TmpEmpinfo.getAll", query="Select emp from TmpEmpinfo emp")})

Create a Session Bean

Now create a session bean to access the entities and to implement business logic.



Create a stateless session bean as **EmpSession.**



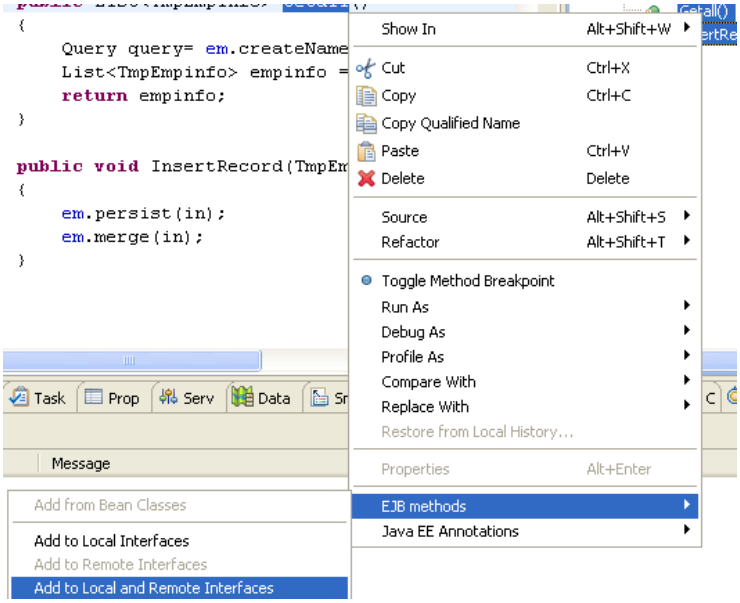
Implement the session bean as follow

```
@Stateless
public class EmpSessionBean implements EmpSessionLocal {

@PersistenceContext(unitName="we_employeeEJB",
type=PersistenceContextType.TRANSACTION)
   EntityManager em;

   public List<TmpEmpinfo> Getall()
   {
          Query query= em.createNamedQuery("TmpEmpinfo.getAll");
          List<TmpEmpinfo> empinfo = query.getResultList();
          return empinfo;
   }

   public void InsertRecord(TmpEmpinfo in)
   {
          em.persist(in);
          em.merge(in);
   }
}
```

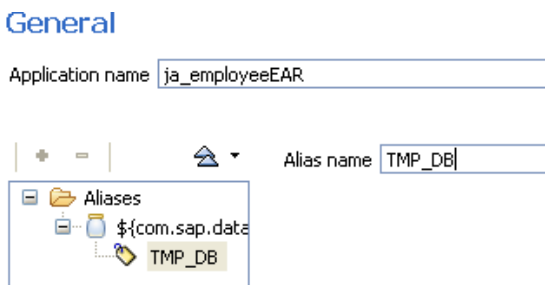
Here we need to create an entity manager to manage and perform operations on entities.

The two methods Getall() and InsertRecord() will perform two basic operations to show and insert the records respectively.

Add these methods to local and remote interface of session bean.

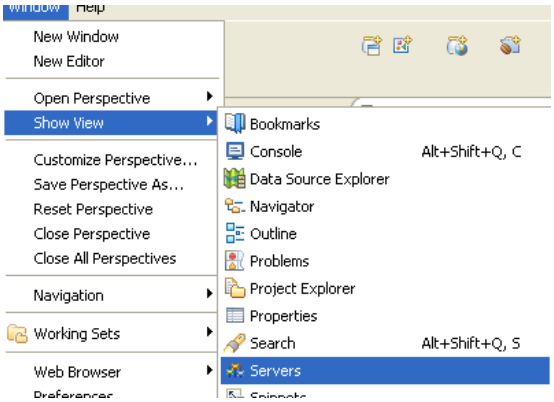


Specify the data source alias in EAR application project

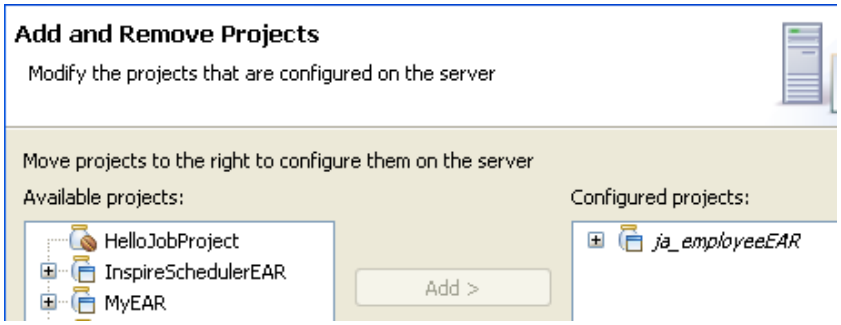


Deploy EAR Application Project

Open the server view



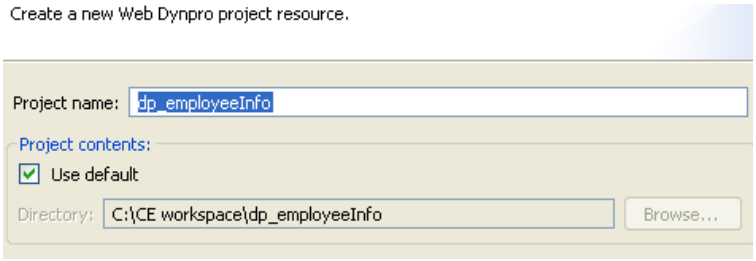
Add the EAR application project



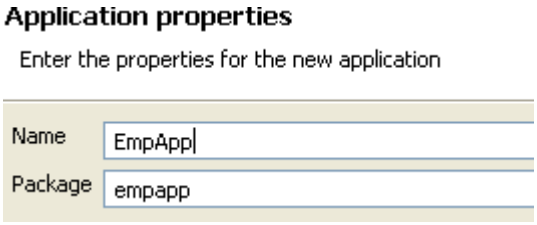
And finish it.

Create a Web Dynpro Project

Open web Dynpro development perspective and create a new project **dp_employeeInfo.**



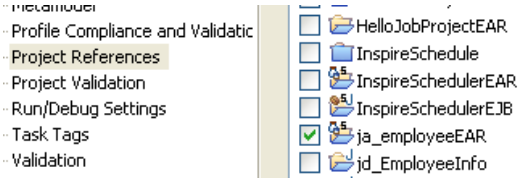
Create a new application **EmpApp** with new component, view and window.



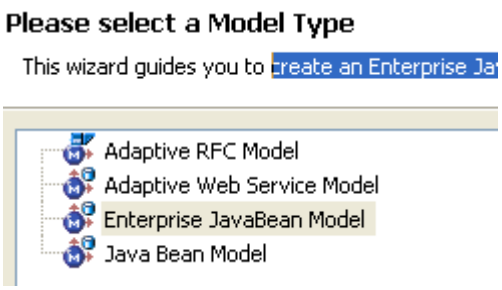
Now to accommodate EJB in webdynpro project create an Enterprise JavaBean Model.

First we need to give reference of EAR application project that contains the EJB project.

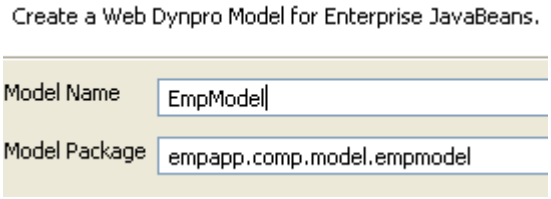
Project->properties->project references



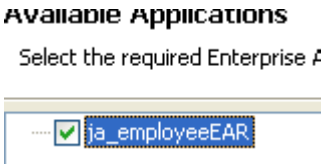
Create an Enterprise Javabean Model



Create a model EmpModel

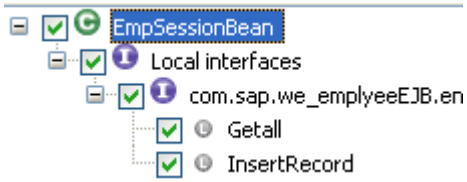


Next step is choose EAR application project



According to the local or remote interface of session bean the methods will be displayed.

Choose the required methods



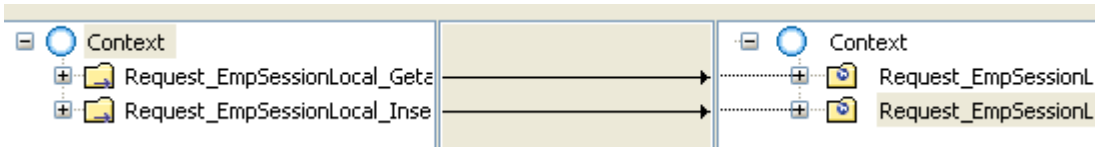
It will create required request and response model class names for methods

S.	Unique Model Class Name	Editable Model Class Name
	com_sap_we_emplyeeEJB_entities_TmpEmpinfo	TmpEmpinfo
	Request_EmpSessionBean_com_sap_we_empl...	Request_EmpSessionLocal...
	Request_EmpSessionBean_com_sap_we_empl...	Request_EmpSessionLocal...
	Response_EmpSessionBean_com_sap_we_em...	Response_EmpSessionLoc...

After creation of model add model as used model in web dynpro component. Do the necessary context mapping between model and controller and controller and view.

Controller to Model context mapping

View to controller context mapping



Controller Implementation

First change the cardinality of every request model node to 1:1.

Model	
Model Class	EmpModel.Request_EmpSessionLocal_Getall
Supplying Relation Role	
Misc	
Name	Request_EmpSessionLocal_Getall
Collection Cardinality	1..1
Initialize Lead Selection	true
Selection Cardinality	0..1

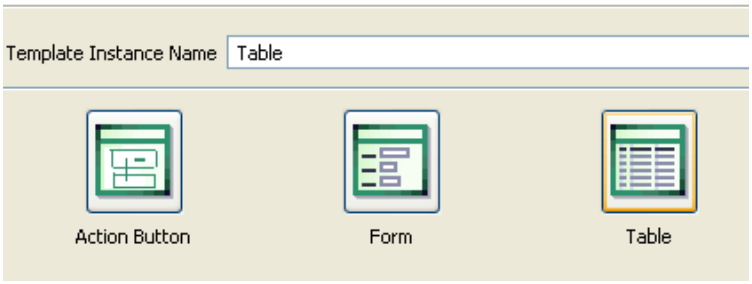
Coding part in controller is only for model initialization.

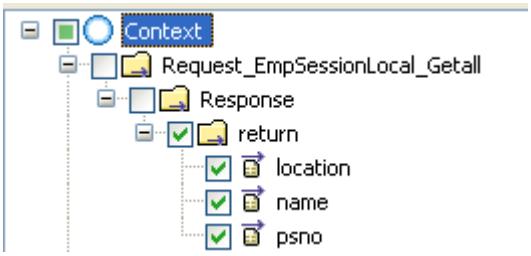
```
    public void wdDoInit()
    {
      //@@begin wdDoInit()
       empapp.comp.model.empmodel.EmpModel empmodel = new
empapp.comp.model.empmodel.EmpModel();

wdComponentAPI.getModelInstanceMap().putDefaultInstance("empapp.comp.model.empmodel.E
mpModel",empmodel);
      //@@end
    }
```

View Implementation

Create a table to display data.

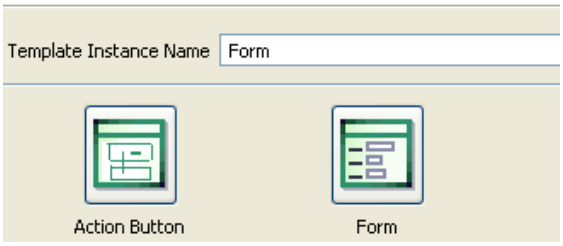
To create a table use apply template and assign context under Request_EmpSessionLocal_Getall



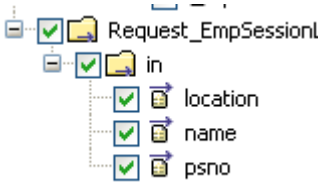


Create a form to enter data.

To create a form use apply template



And assign context to it.



Create two buttons **show** and **Insert** to display data from backend and to insert data into backend.

Assign onActionShow method to **Show** button and implement it as below.

```
  public void onActionShow(com.sap.tc.webdynpro.progmodel.api.IWDCustomEvent wdEvent
)
  {
    //@@begin onActionShow(ServerEvent)
     try {

   wdContext.currentRequest_EmpSessionLocal_GetallElement().modelObject().execute();
   } catch (Exception e) {
         // TODO Auto-generated catch block

   wdComponentAPI.getMessageManager().reportException("error"+e.getCause(),true);

   wdComponentAPI.getMessageManager().reportException("error"+e.getCause().getCause()
,true);
   }
    //@@end
  }
```

Assign `onActionInsert` method to **Insert** button and implement it as below.

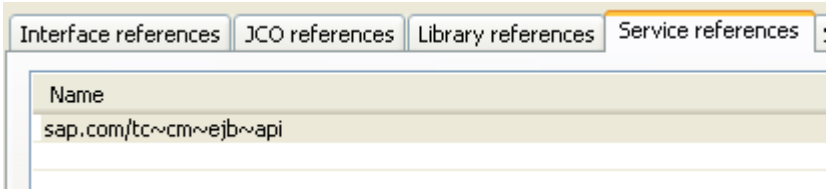
```
  public void onActionInsert(com.sap.tc.webdynpro.progmodel.api.IWDCustomEvent
wdEvent )
  {
    //@@begin onActionInsert(ServerEvent)

     try {

   wdContext.currentRequest_EmpSessionLocal_InsertRecordElement().modelObject().execu
te();

   wdContext.currentRequest_EmpSessionLocal_GetallElement().modelObject().execute();
         } catch (Exception e) {
               // TODO Auto-generated catch block

   wdComponentAPI.getMessageManager().reportException("error"+e.getCause(),true);

   wdComponentAPI.getMessageManager().reportException("error"+e.getCause().getCause()
,true);
         }
    //@@end
  }
```

Additional Web Dynpro Settings

Before Building and deploying application we need to add some references to web dynpro application as follow.

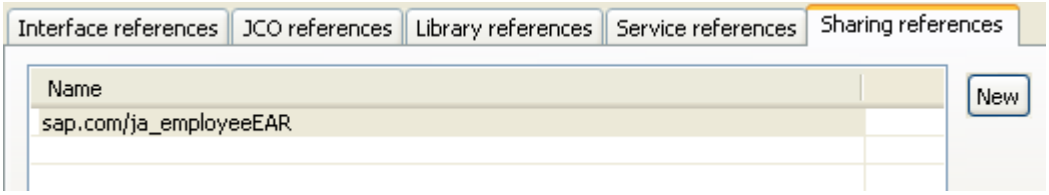
Dp_employeeInfo->Properties->Web Dynpro->Service references

Add sap.com/tc~cm~ejb~api



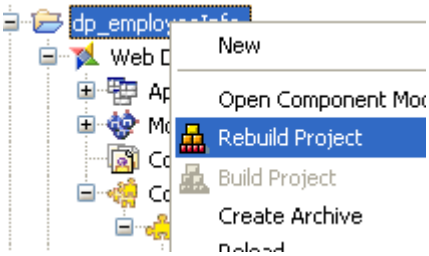
Dp_employeeInfo->Properties->Web Dynpro->Sharing references

Add sap.com/ja_employeeEAR

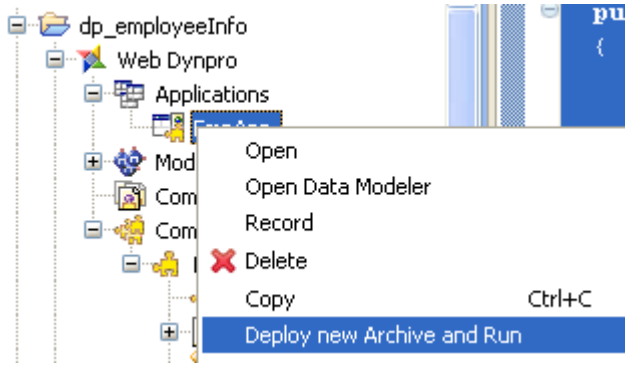


Build and Deploy

Rebuild the whole Project and create its archive

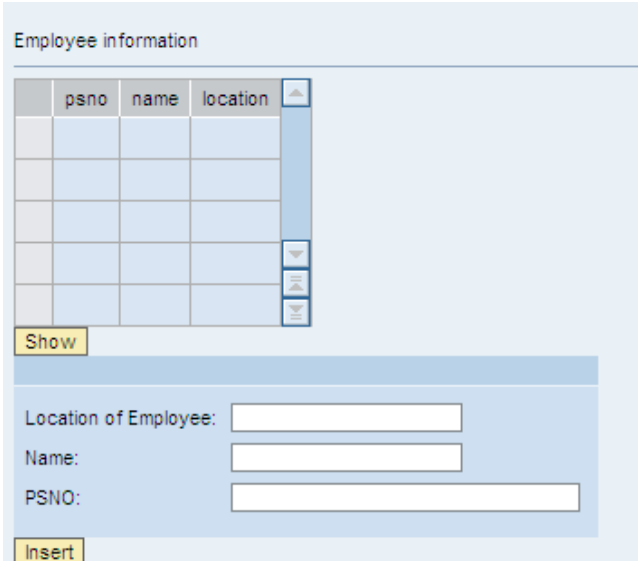


Deploy the Project and run the application.

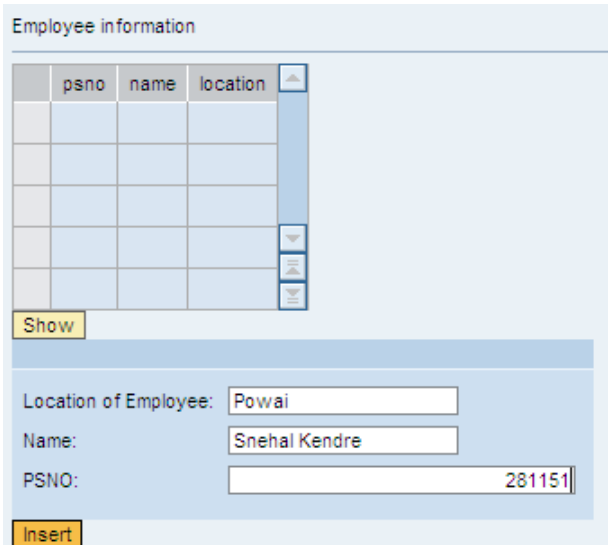


Application

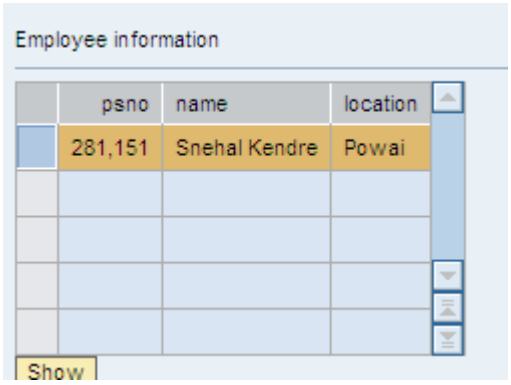
Run application will open employee information application



Enter the employee information and click insert button



After inserting the value you will be able to see records in table



Related Content

https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/83086a1d-0e01-0010-2797-f1ce7f8f562d

https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/20b21892-c31c-2a10-f484-fcef1eaf8c4f

For more information, visit the User Interface Technology homepage.

Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.