



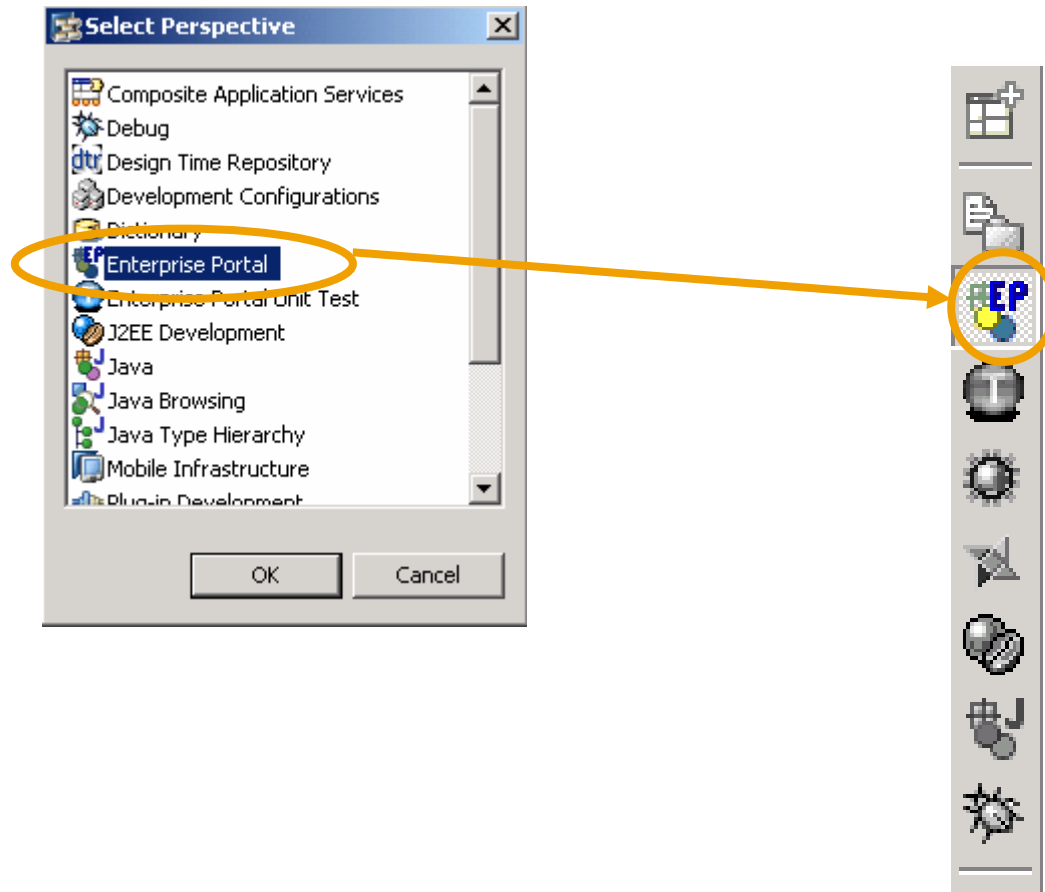
SAP NetWeaver Developer Studio for KMC

Requirements

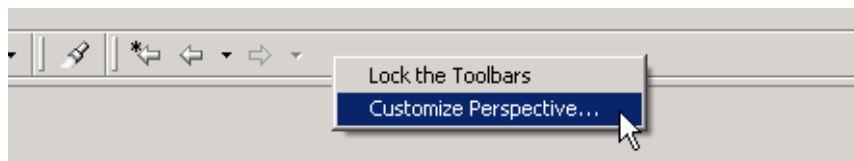
- Install a local JDK
- Install a local SAP NetWeaver Developer Studio
 - ◆ <http://service.sap.com/instguidesNW04> -> Installation -> Installation Guide -> SAP NetWeaver Developer Studio)
- Remote or local SAP Enterprise Portal with KMC

Activate the Enterprise Portal Perspective:

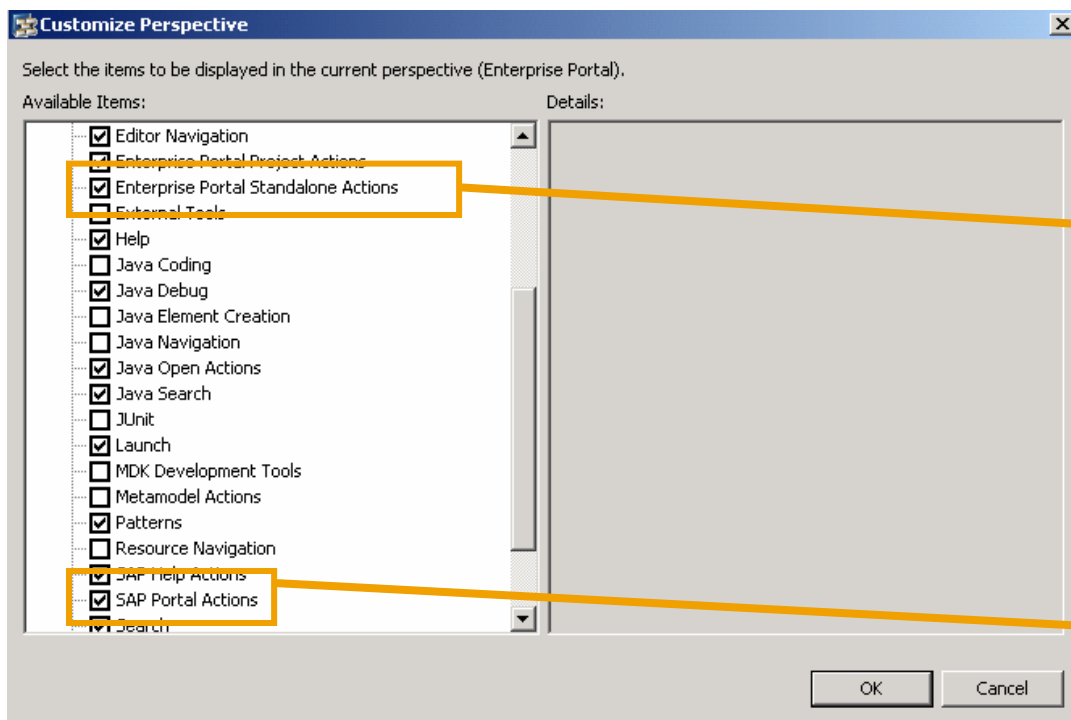
- Go to Window – Open Perspective – Other – Select Enterprise Portal



Customize the Toolbar to get the EP & KM PlugIn icons: right-click the toolbar -> Customize Perspective



Select the items for the Portal and KM Plugins



EP Plugins



KM Plugins

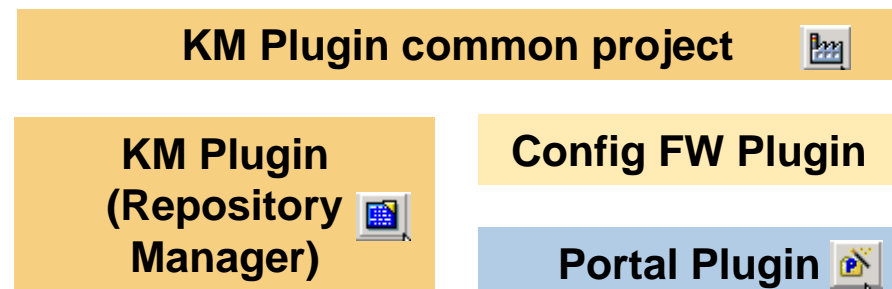


Portal-Plugins

- **Creates an empty portal application project**
 - ◆ **Creates source development folders for API & Core coding**
 - ◆ **Creates a filesystem structure for PAR file creation**
 - Public area for client content
 - Private area for server side content
 - ◆ **Creates a deployment descriptor file (portalapp.xml)**
- **Config Framework plugin**
- **Support for Portal specific components (portal services, portal components)**
- **Support for WebServices**

KM-Plugins

- **Creates source code skeletons for CM components**
- **Creates configuration entries (XML files) for config framework**
- **Creates Wrapper classes for registration at CrtClassLoader**

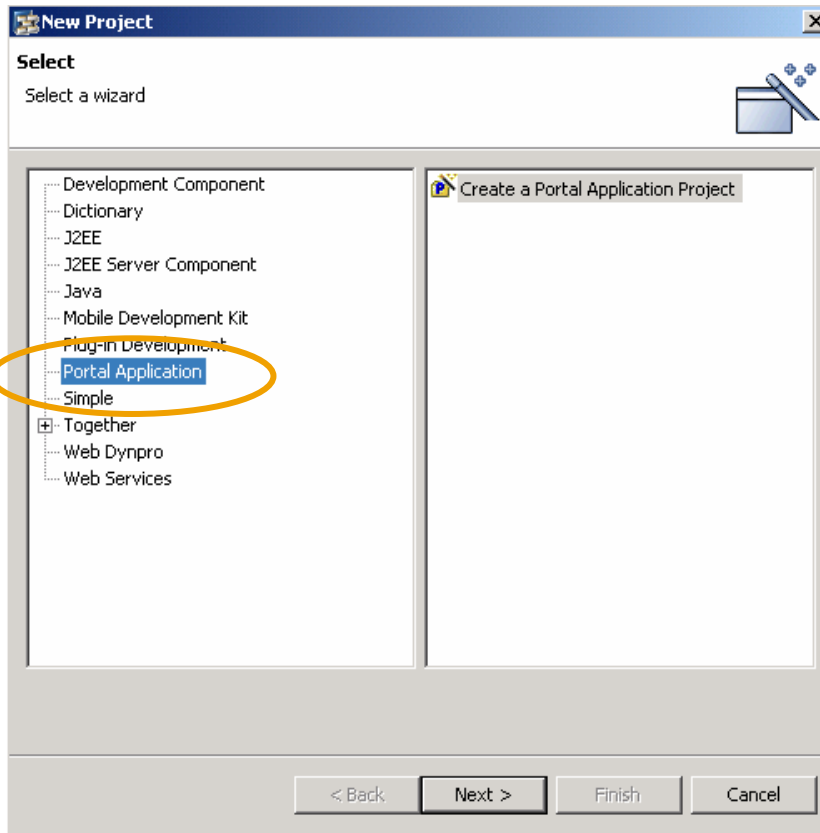


Steps:

1. **Create a new Portal Application Project**
2. **Integrate the KM jar files**
3. **Create a new Portal Application Object (KM component)**
4. **Select the Portal Application Project you want to create the KM component in.**
5. **Specify technical data**
 - **KM component ID**
 - **Java package name**
 - **Options used by KM component (optional)**
6. **Finalize project**

Create a new Portal Application project (1)

- From the menu, choose: **File** -> **New ...** -> **Project**

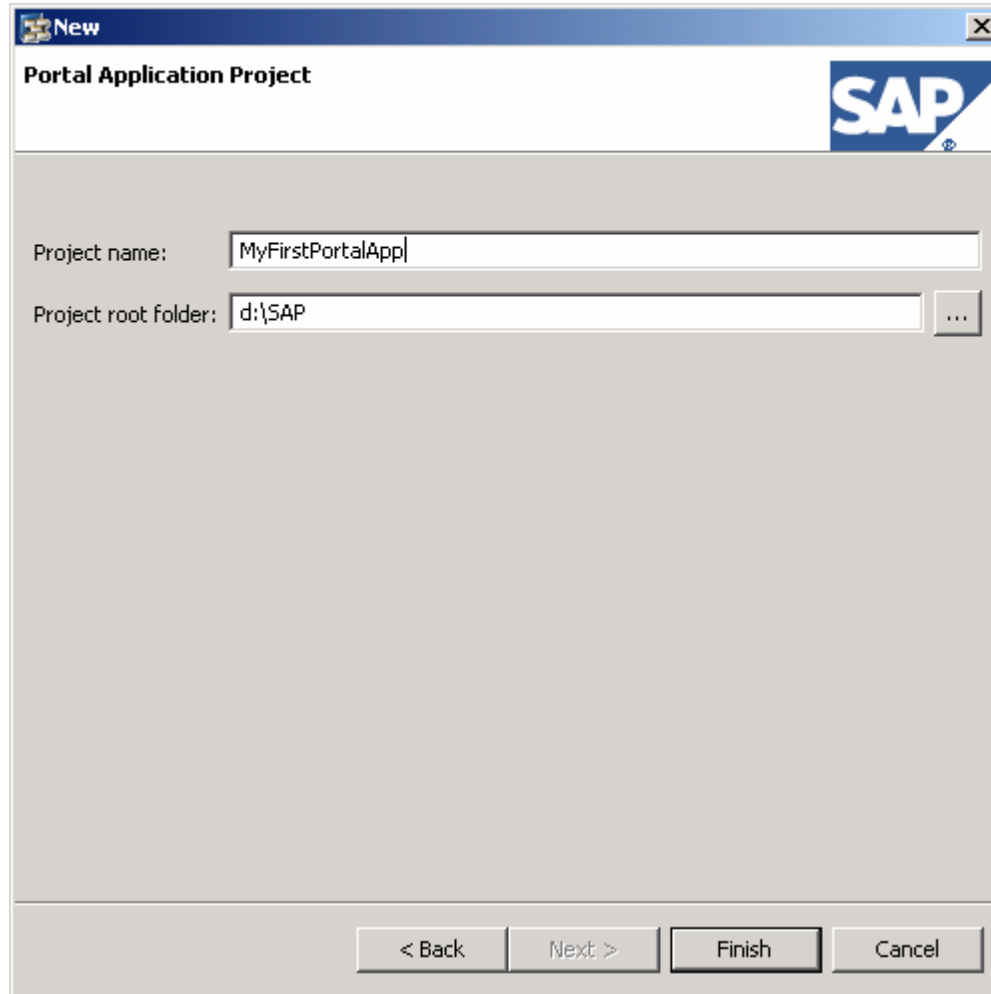


Or choose the appropriate icon in the toolbar:



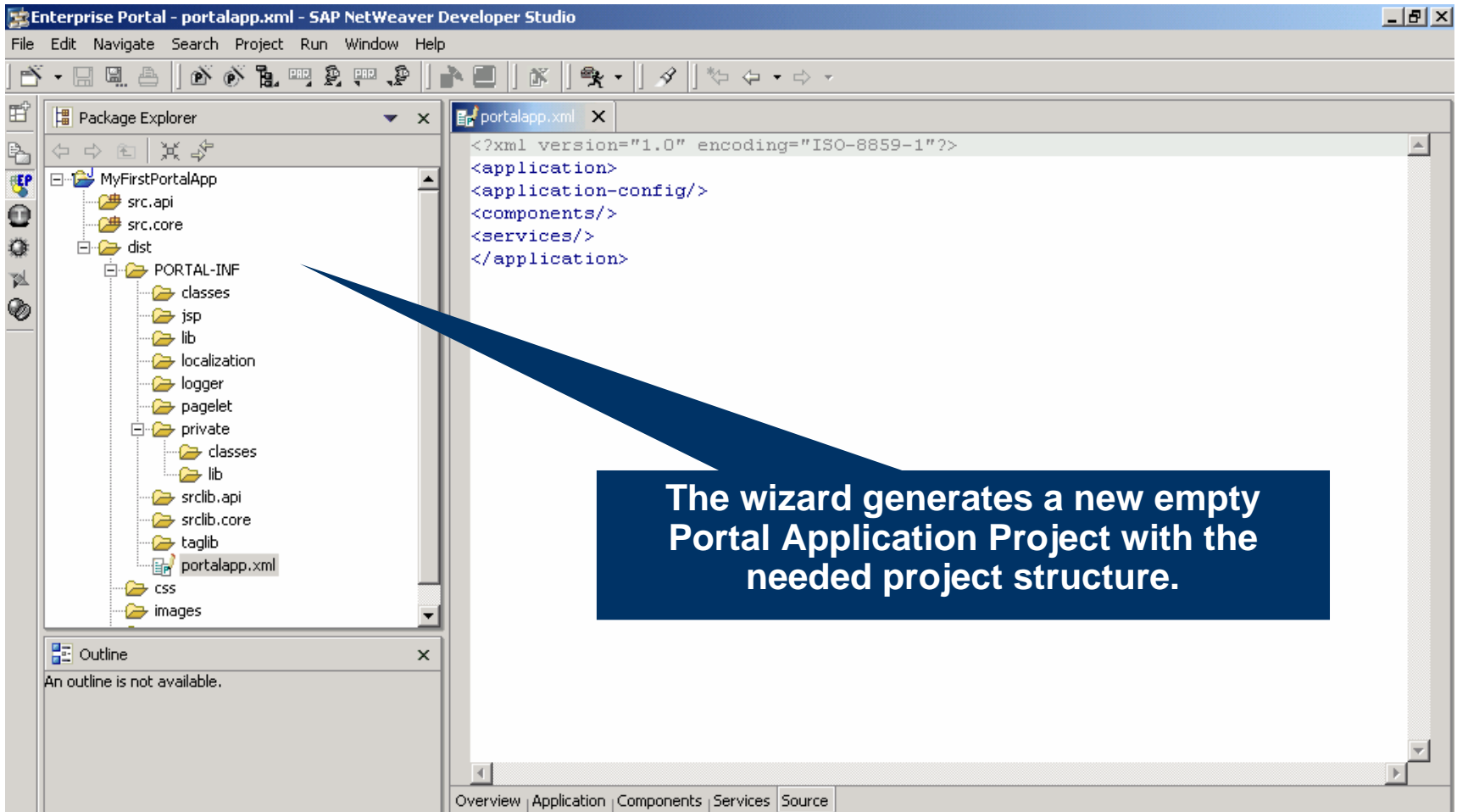
Create a new Portal Application project (2)

Define a **project name** and the **root folder** for your project:



The screenshot shows a dialog box titled "New Portal Application Project" with the SAP logo in the top right corner. The dialog contains two input fields: "Project name:" with the text "MyFirstPortalApp" and "Project root folder:" with the text "d:\SAP". At the bottom of the dialog, there are four buttons: "< Back", "Next >", "Finish", and "Cancel".

Create a new Portal Application project (3)



The screenshot displays the SAP NetWeaver Developer Studio interface. The Package Explorer on the left shows a project named 'MyFirstPortalApp' with a hierarchical structure of folders: 'src.api', 'src.core', 'dist', 'PORTAL-INF' (containing 'classes', 'jsp', 'lib', 'localization', 'logger', 'pagelet'), 'private' (containing 'classes', 'lib'), 'srclib.api', 'srclib.core', 'taglib', 'portalapp.xml', 'css', and 'images'. The main editor window shows the 'portalapp.xml' file with the following XML content:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<application>
<application-config/>
<components/>
<services/>
</application>
```

A blue callout box with white text points to the 'portalapp.xml' file in the Package Explorer and the XML content in the editor, stating: "The wizard generates a new empty Portal Application Project with the needed project structure."

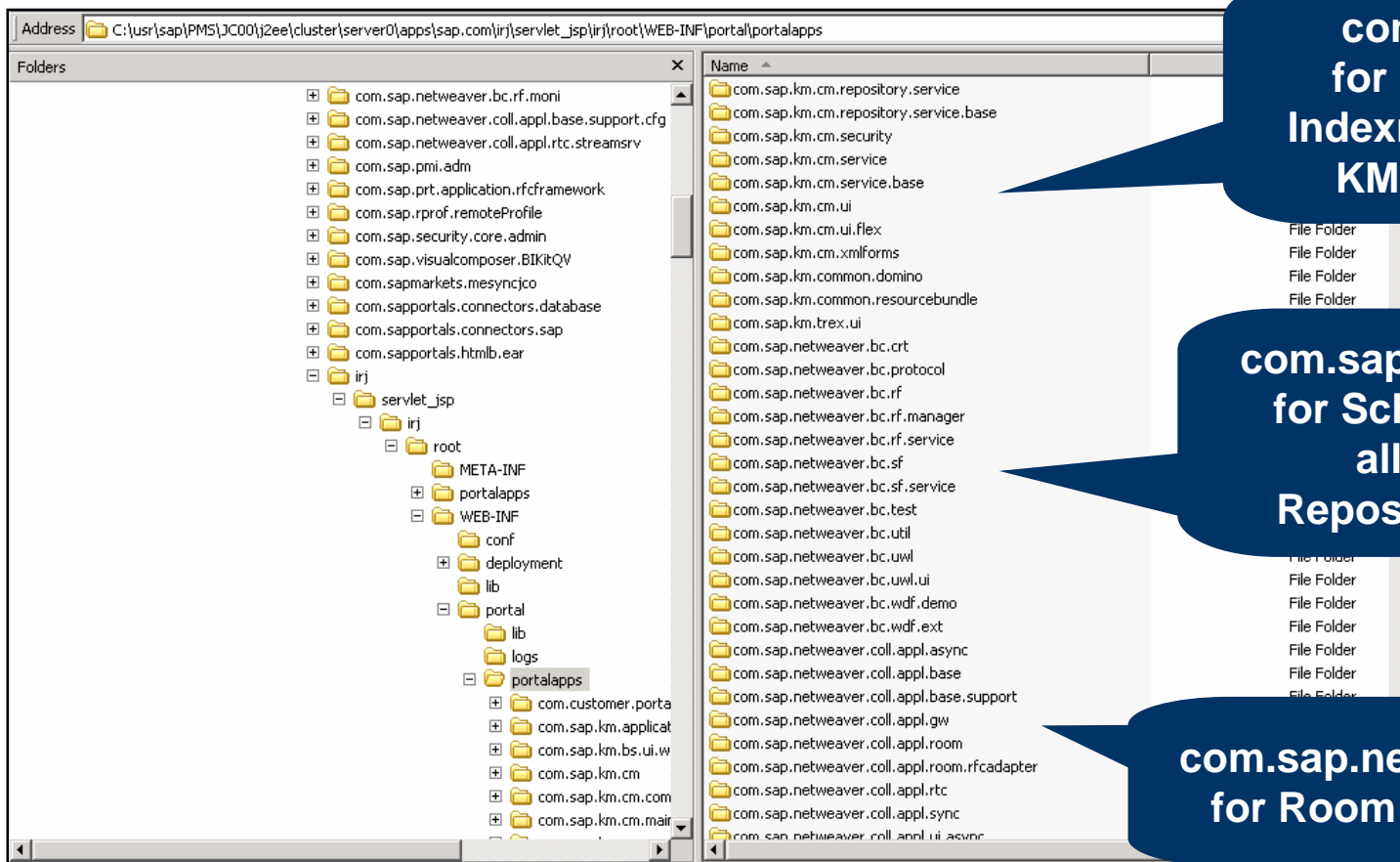
At the bottom of the editor window, there are tabs for 'Overview', 'Application', 'Components', 'Services', and 'Source', with 'Source' currently selected.

Portal Applications ...

- ... are bundles of Portal Components and Portal Services.
- ... are packaged as PAR (Portal Application Archive) files – format that the Portal Runtime accepts for deployment of Portal Applications.
- A PAR file is an archive file/standard JAR file (ZIP format) containing a Portal Application with a XML-based Portal Application Deployment Descriptor (DD) called portalapp.xml.
- A DD describes the portal application itself and provides information required at runtime.

- **Portal Applications contain two types of resources:**
 - ◆ **Web Resources**
 - Accessible via http(s) requests to a web server
 - All files NOT under PORTAL-INF (WEB-INF)
 - ◆ **Non-Web Resources**
 - Not accessible via an http(s) request
 - All files under PORTAL-INF (WEB-INF)
- **A Portal Application can contain:**
 - ◆ several Portal Components and several Portal Services
 - ◆ only 1-n Portal Components
 - ◆ only 1-n Portal Services
- **Each Portal Component can be accessed by**
<par file name>.<component name>

Integrate the KM libraries from your Portal installation: take the public JAR files from the par files



Hint: copy all needed jar files to a local folder

Go to the SDN – Knowledge Management – Quick Links: Knowledge Management and Collaboration Developer's Guide

Address <https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/com.sapportals.km.docs/documents/a1-8-4/Knowledge%20Management%20and%20Collaboration%20Developer> Go Links >>

Knowledge Management and Collaboration (KMC)

Knowledge Management provides functionalities to manage unstructured information (like documents) and link them with structured information (like Business Objects). It also provides search and navigation functionalities.

Collaboration embraces features to allow teams to collaborate, using the portal (like chatting, online meetings or sharing documents in a team room).

Introduction to the KMC Platform	An overview of the several layers of the KMC Platform.
The Development Environment	Explains how to develop, build and deploy KMC applications and extensions with the NetWeaver [®] 04 IDE.
Using and Extending the KMC Platform	Various "How-to" guides that help you to extend the KMC Platform. For example, "How to Implement Flexible Groupware Connectivity".
Guide to Examples	A list of examples, available as ZIP files, to project into the NetWeaver [®] 04 IDE.
API Reference (JavaDoc)	The JavaDoc for the KMC APIs.

[Overview](#) [Package](#) **[Class](#)** [Use Tree](#) [Deprecated](#) [Index](#) [Help](#)

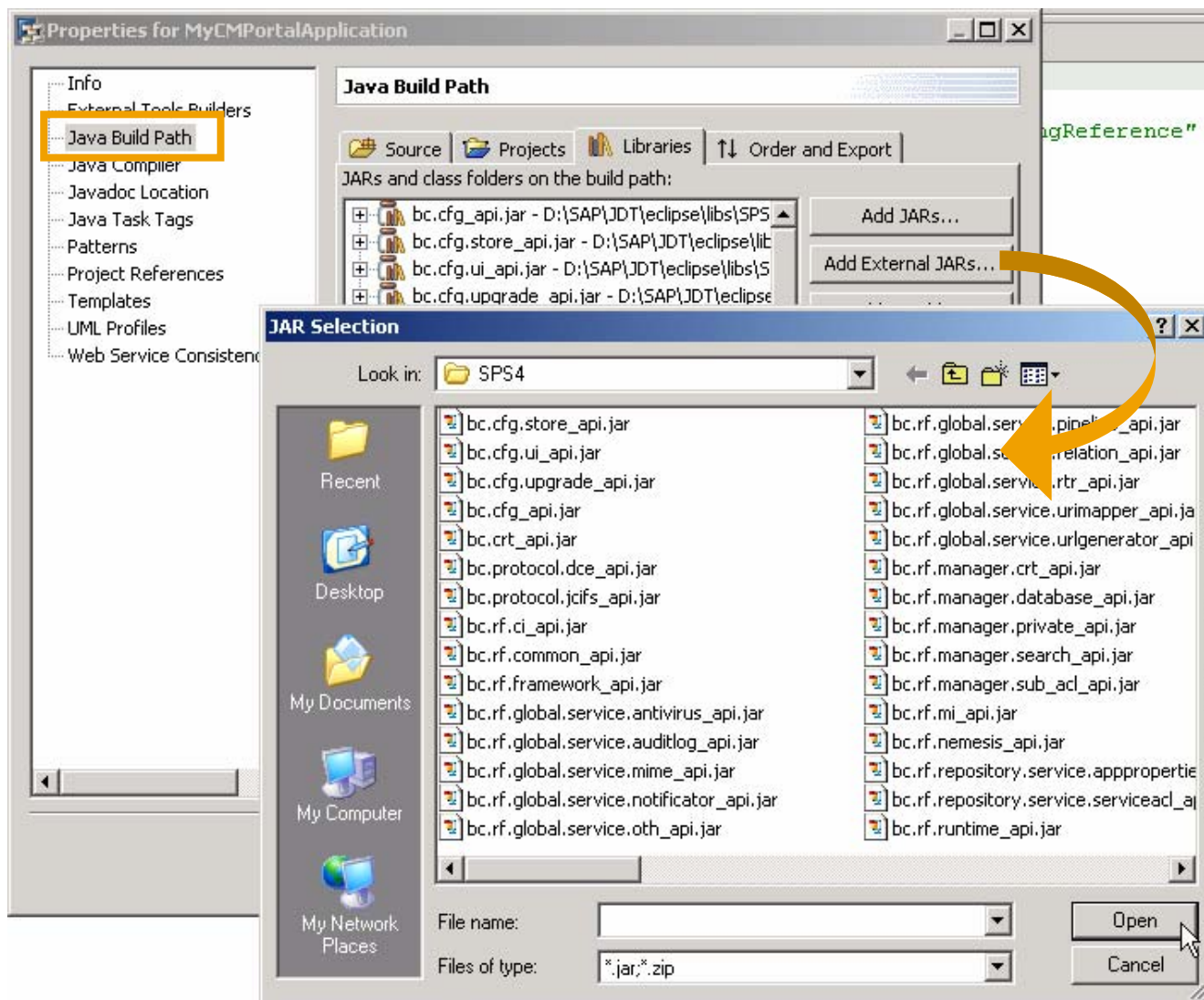
[PREV CLASS](#) [NEXT CLASS](#) [FRAMES](#) [NO FRAMES](#)

SUMMARY: [INNER](#) | [FIELD](#) | [CONSTR](#) | [METHOD](#) DETAIL: [FIELD](#) | [CONSTR](#) | [METHOD](#)

```
com.sapportals.wcm.rendering.collection
Class AbstractComponent
[contained in: com.sap.km.cm.ui.flex.par - km.shared.ui.flex.collection_api.jar]
java.lang.Object
|--com.sapportals.wcm.rendering.collection.AbstractComponent
```

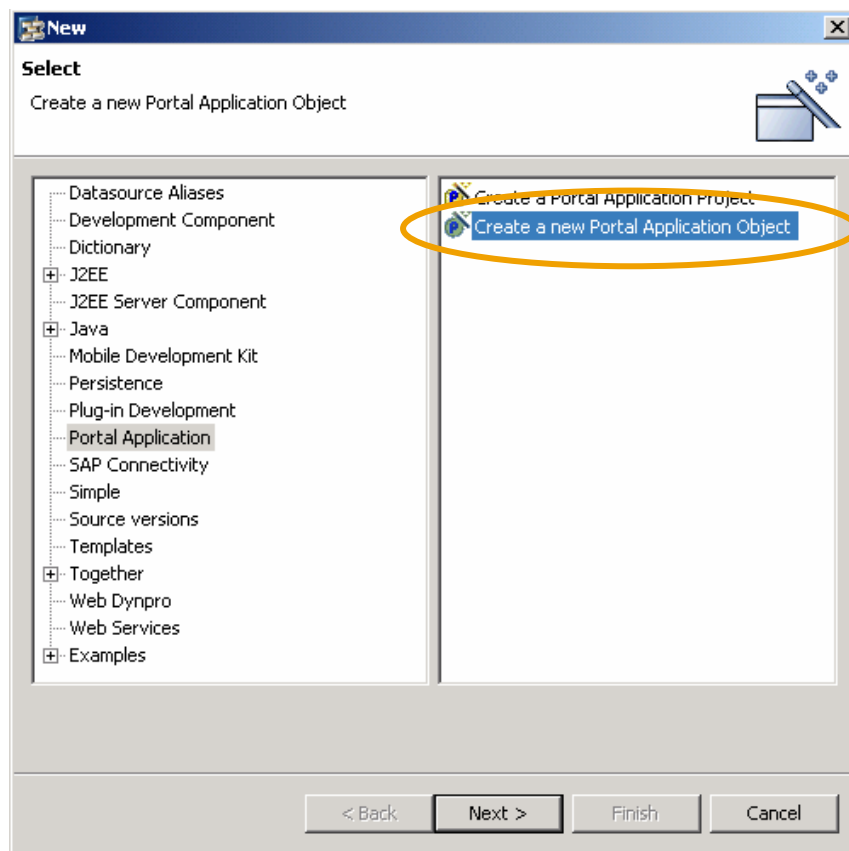
All Implemented Interfaces:
[IComponent](#)

Add KM libraries to your Portal Application KM project



Create a new Portal Application Object (1)

- From the menu, choose: **File → New ... → Create a new Portal Application Object**
- OR **File → New ... → Other → Create a new Portal Application Object**



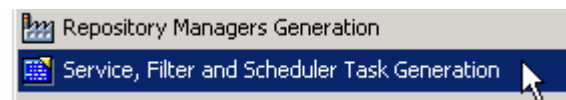
OR choose the appropriate icon in the toolbar:



OR KM wizards directly via icons in the toolbar

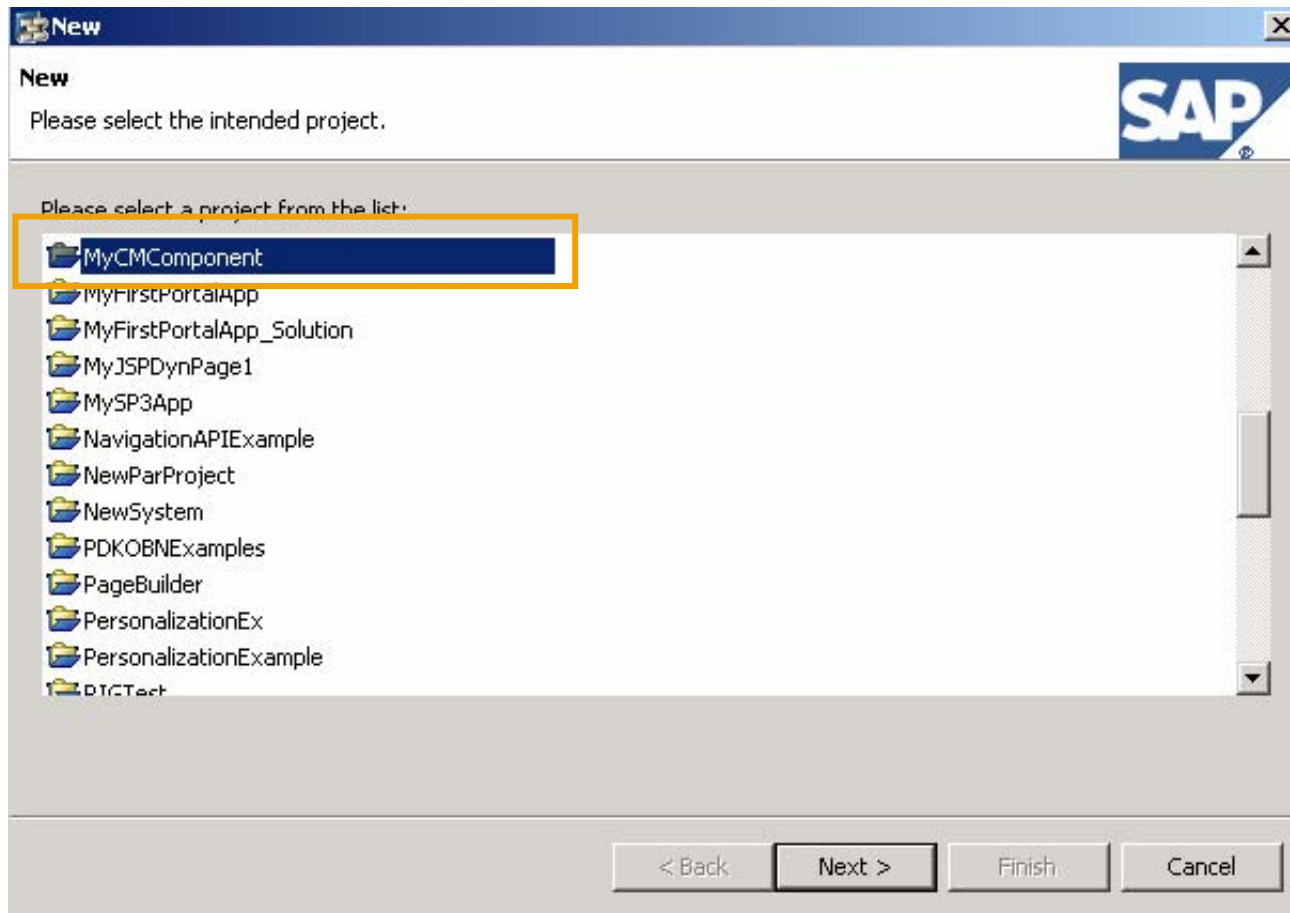


OR right-clicking your project



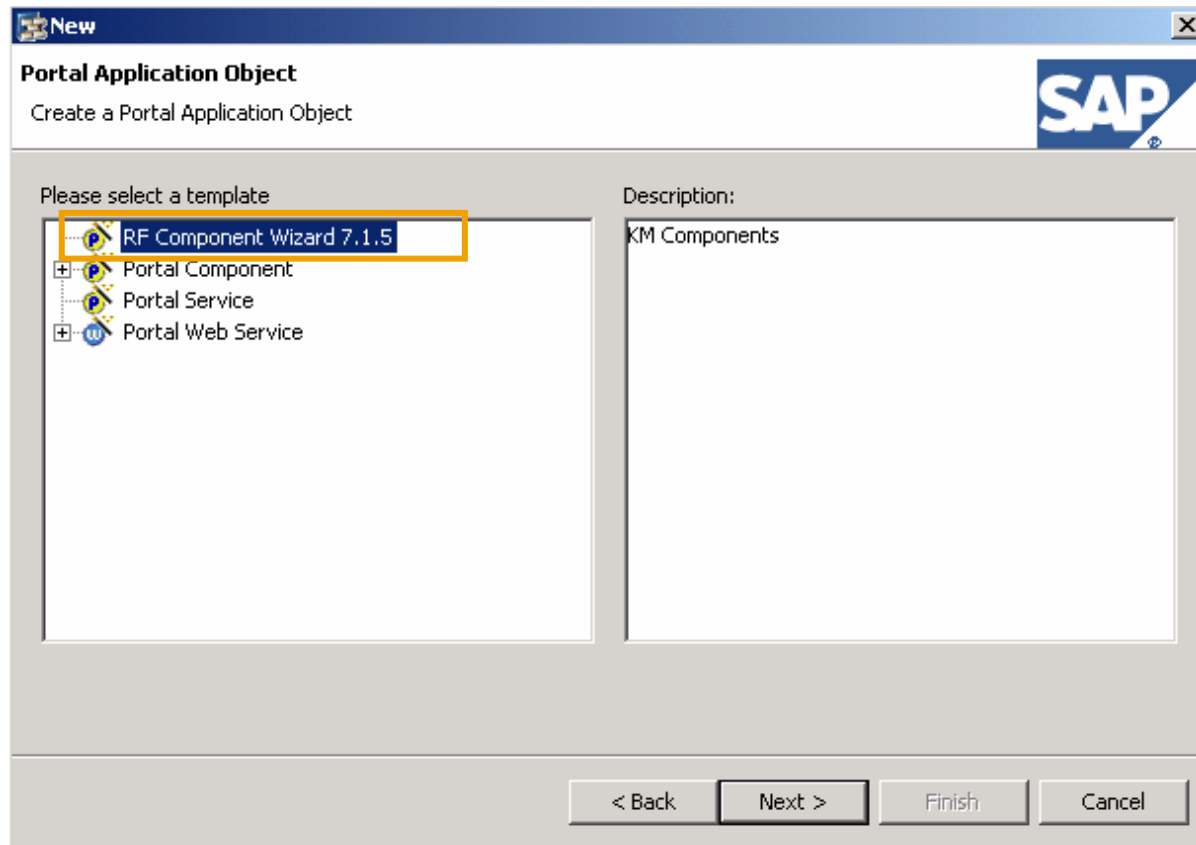
Create a new Portal Application Object (2)

Select the Portal Application Project and click „Next“:



Create a new Portal Application Object (3)

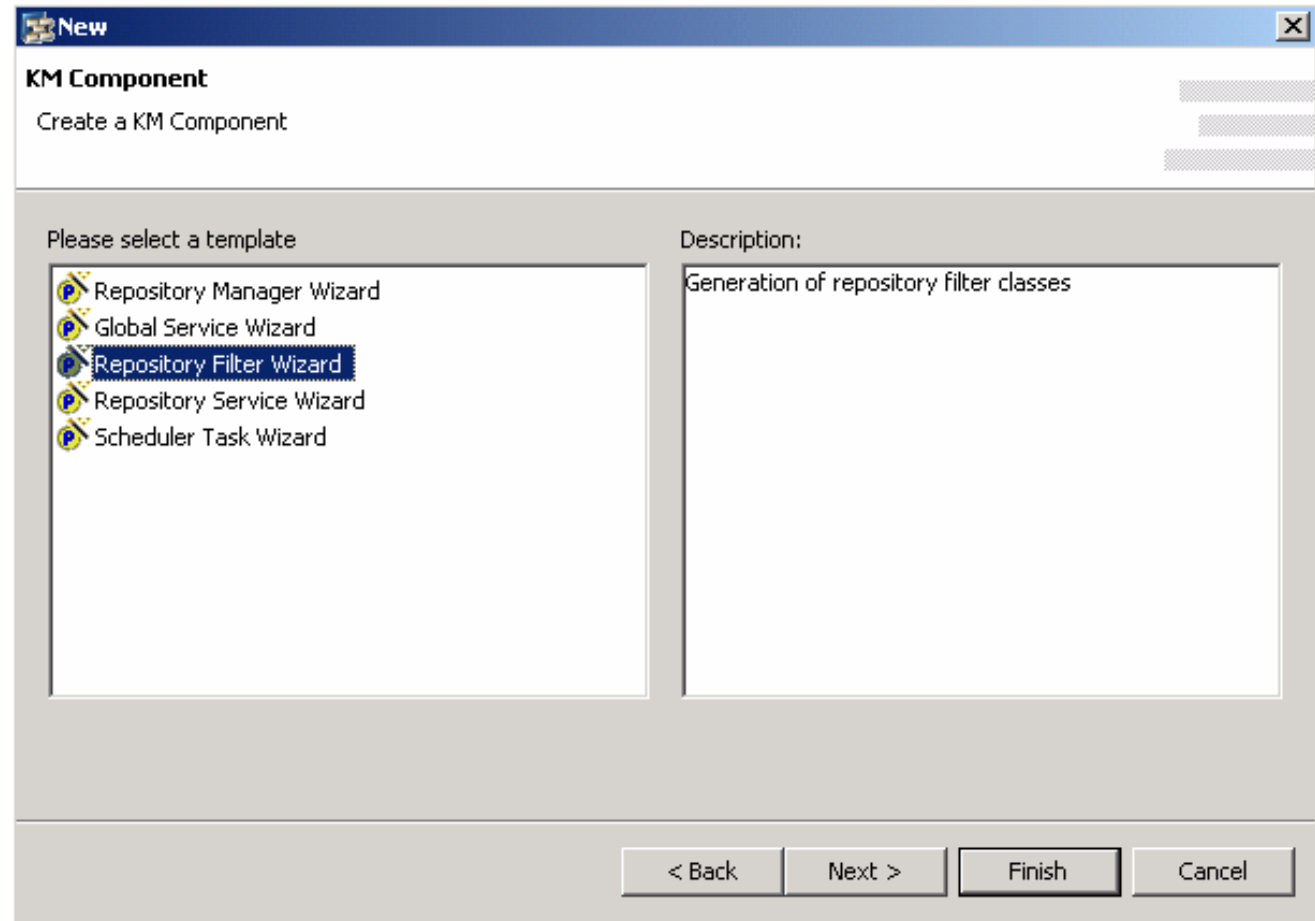
Create a new Repository Framework Component for your Portal Application Project:



Create a new Portal Application Object (4)

Select which KM Component you want to implement:

- Repository Manager
- Global Service
- Repository Filter
- Repository Service
- Scheduler Task



Example: Create a Repository Filter → Namespace Filter

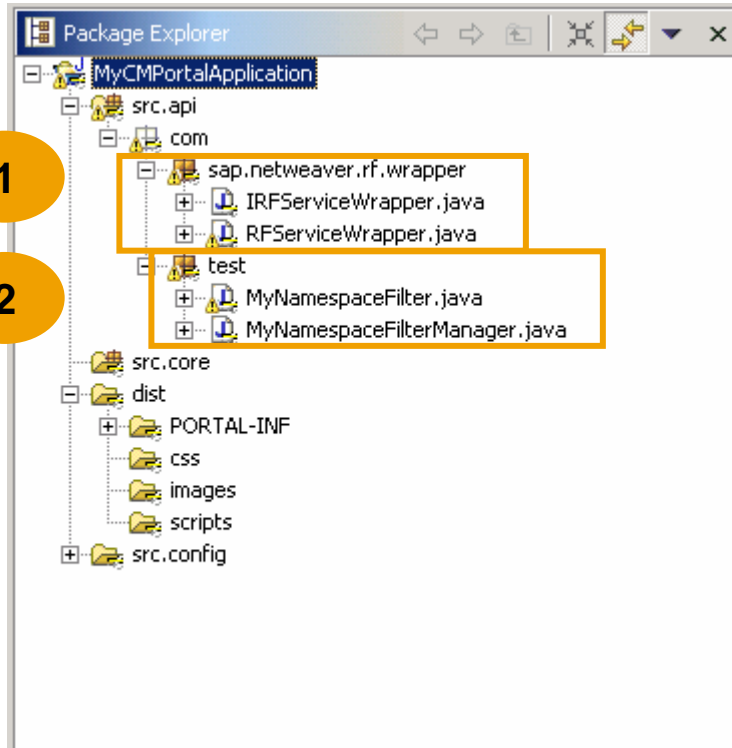
The screenshot shows the 'KMC Wizard' dialog box, specifically the 'Repository Filter Wizard' step. The title bar reads 'KMC Wizard' and the subtitle is 'Repository Filter Wizard'. Below the subtitle, it says 'Generation of repository filter classes'. The main area contains several configuration options:

- Config Framework Version:** A dropdown menu showing 'NWD4'.
- Generate project classpath with BC libraries from plugin**
- Class name:** A text field containing 'MyNamespaceFilter'.
- Package:** A text field containing 'com.test'.
- Name (if different from class name):** An empty text field.
- Types:** A dropdown menu with 'amespace' selected. A mouse cursor is pointing at the dropdown, and a list of options is visible: 'Namespace', 'Property', 'Content', and 'mychrisapplication'.

At the bottom of the dialog, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

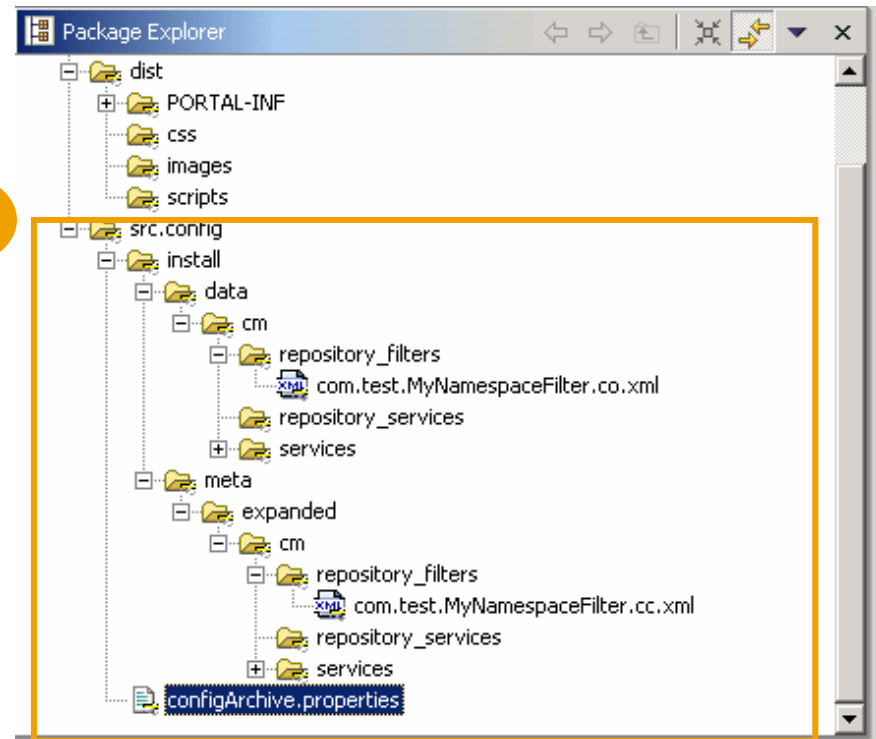
Project structure

1. Portal service wrapper in src.API
2. CM Component classes
3. Config Archive for config framework

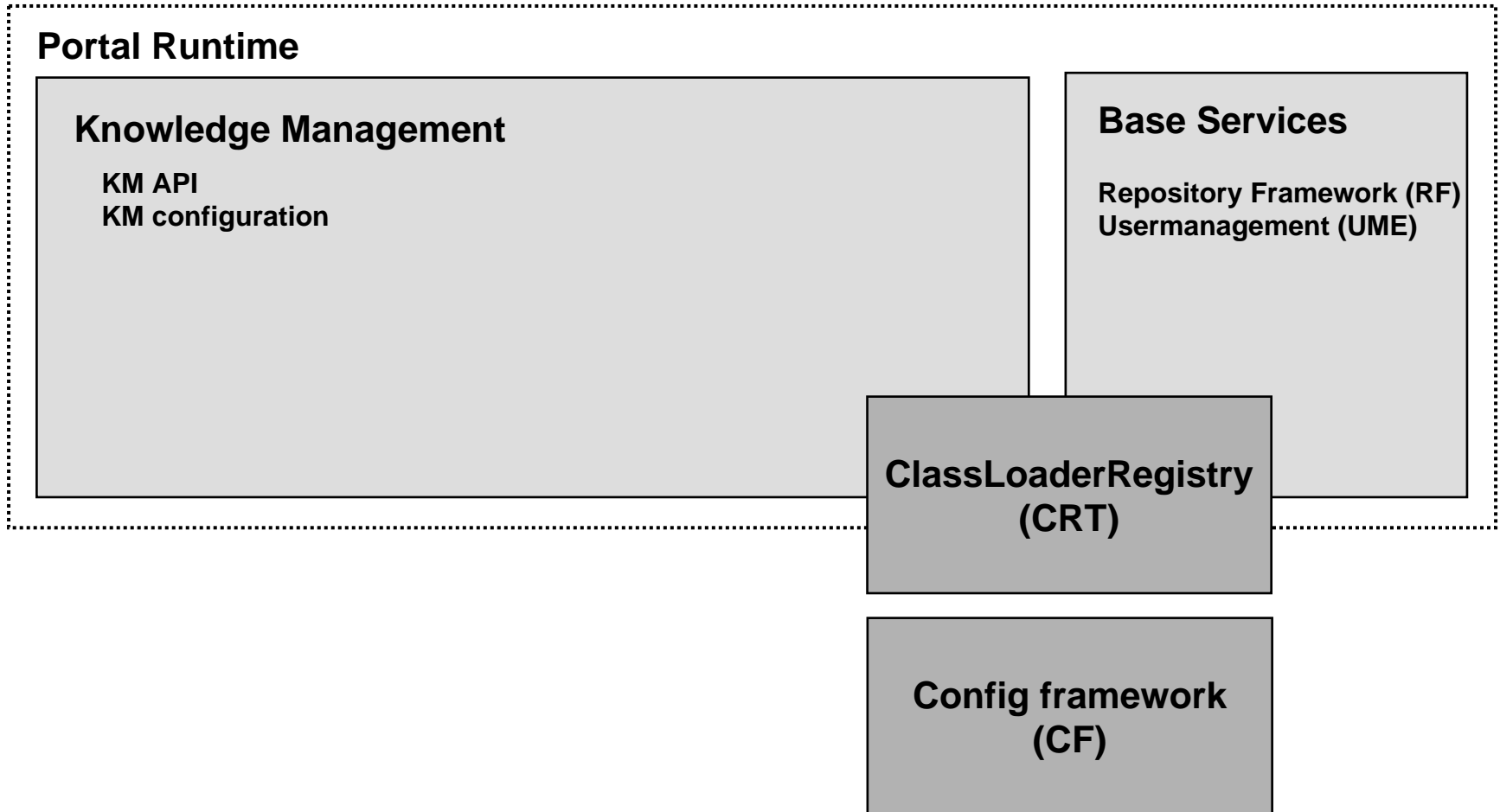


3

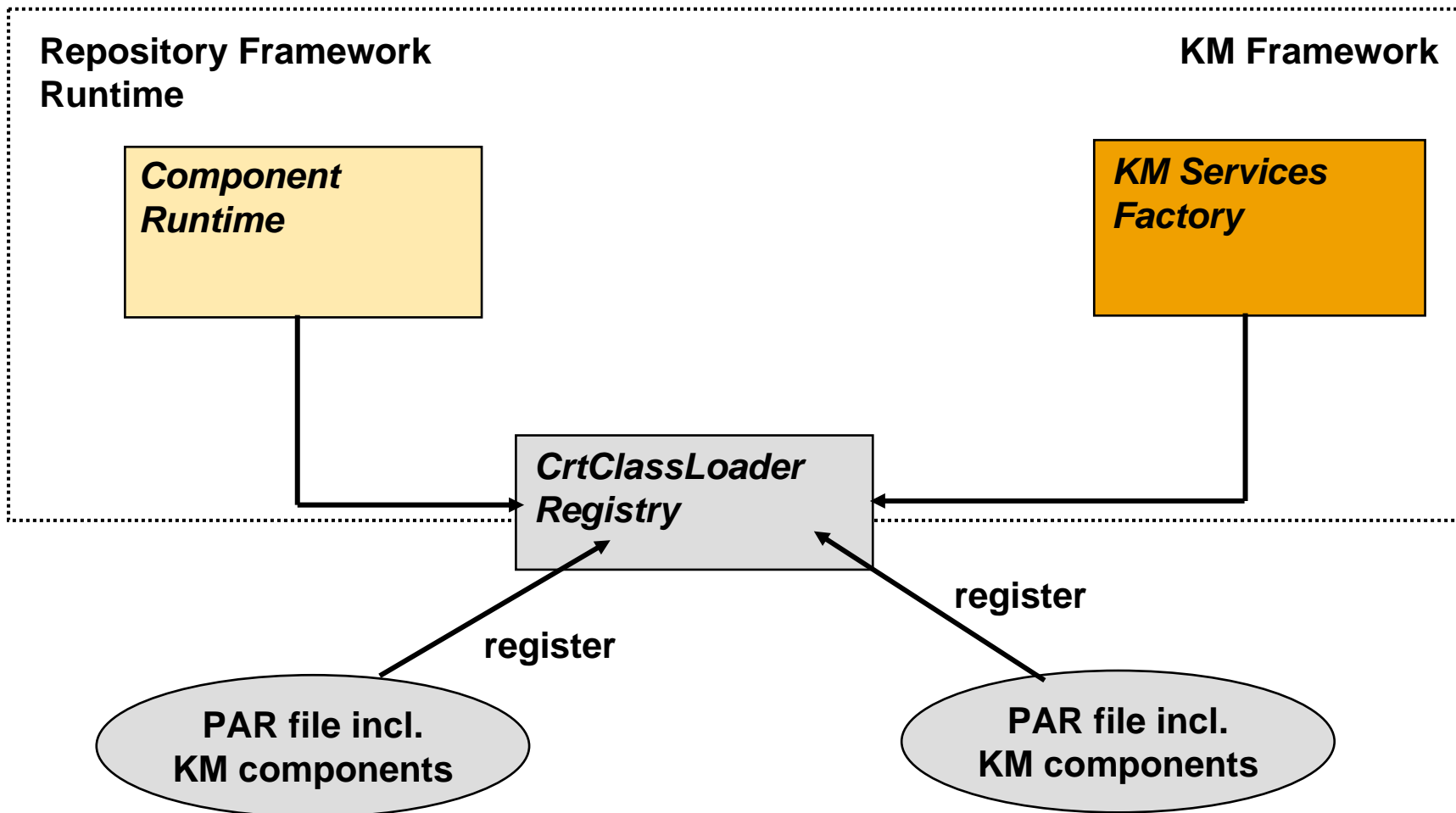
+



Necessary components for KM component implementation & deployment process



CrtClassLoader Registration



A standard Portal Service with interface and implementation class uses the CrtClassLoader Registration

The Service KEY has to be defined:

```
package com.sap.netweaver.rf.wrapper;  
  
import com.sapportals.portal.prt.service.IService;  
  
public interface IRFServiceWrapper extends IService  
{  
    public static final String KEY = "/* To be inserted */";  
}
```

Portal Service KEY:=
<Servicename>

Naming convention:

<ServiceName> := <ApplicationName>.<Service>
<ApplicationName> := parfilename

A standard Portal Service implements the CrtClassLoader Registration

```
public class RFServiceWrapper implements
    IRFServiceWrapper {

    public void init(IServiceContext serviceContext){
        mm_serviceContext = serviceContext;
        CrtClassLoaderRegistry.addClassLoader(
            this.getKey(), this.getClass().getClassLoader()
        );
    }
}
```

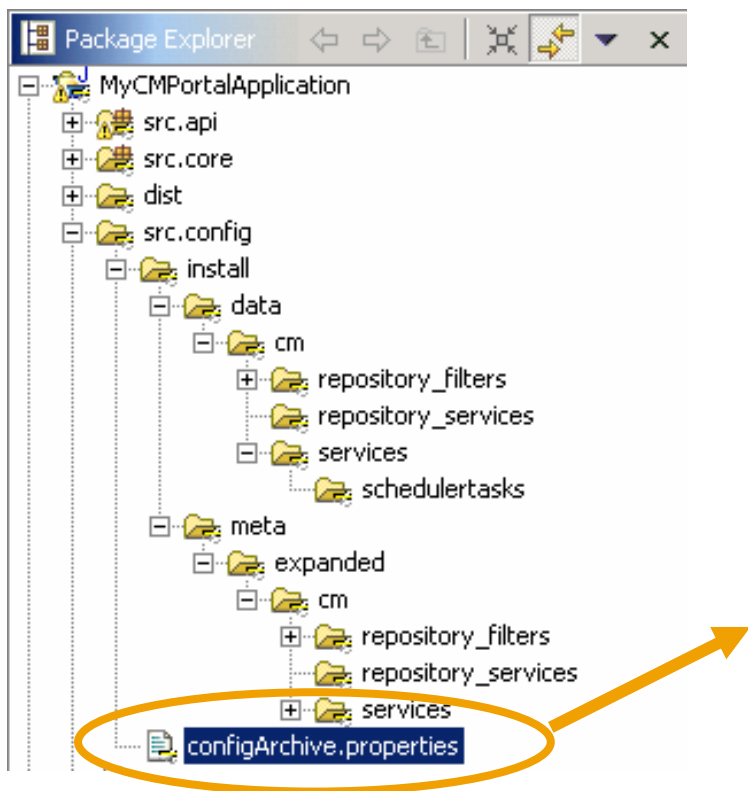
Registers this service wrappers classloader at the KM CrtClassLoader registry.

→ Mandatory implementation for all KM components

Using the configuration framework (KM base services)

- **KM components are exposed as configurables**
- **Access the configuration data with base service wrapper classes (IConfigurable)**
- **Deploy pre-defined Classes definitions and configurable instances with a KM component PAR files**

Create / Check KM component configuration



configArchive.properties:

ca.name=MyCMPortalApplication.prjconfig
ca.version=6.0.1.1
ca.creation.time=1117
ca.creation.date=20040826
ca.creation.user=unknown
ca.creation.machine=unknown
ca.dependencies: bc.util.prjconfig,
bc.sf.prjconfig, bc.sf.service.prjconfig,
bc.rf.prjconfig

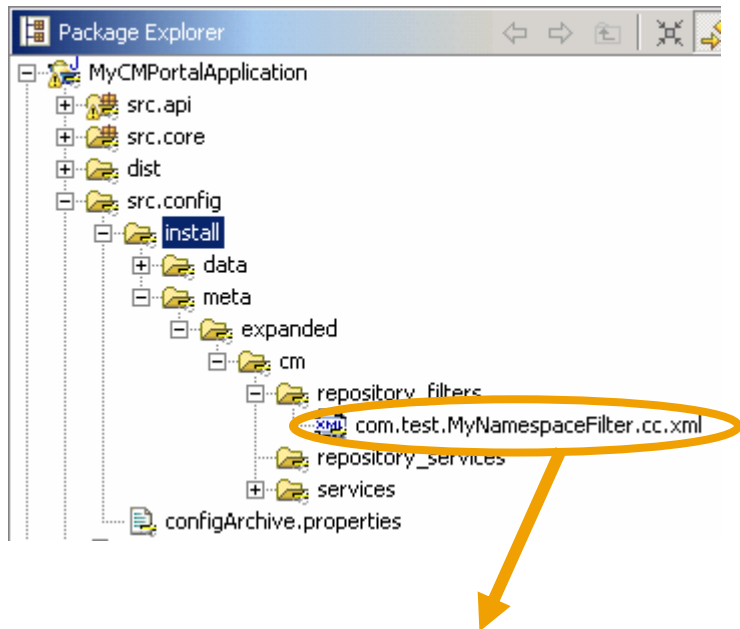
cma.name=MyCMPortalApplication.prjconfig
cma.version=6.0.1.1
cma.storage=sfs
cma.creation.time=1117
cma.creation.date=20040826
cma.creation.user=unknown
cma.creation.machine=unknown
cma.dependencies: bc.util.prjconfig,
bc.sf.prjconfig, bc.sf.service.prjconfig,
bc.rf.prjconfig

Structure:

src.config/install/data/cm/...
src.config/install/meta/expanded/cm/

...

Create / Check KM component configuration



1. Contains the class definition
2. Contains instance files

2. Instance definition (com.test.MyNamespaceFilter.co.xml):

```
<Configurable configclass="com.test.MyNamespaceFilter">  
  <property name="name"  
    value="com.test.MyNamespaceFilter" />  
  <property name="active" value="true" />  
  <property name="description" />  
</Configurable>
```

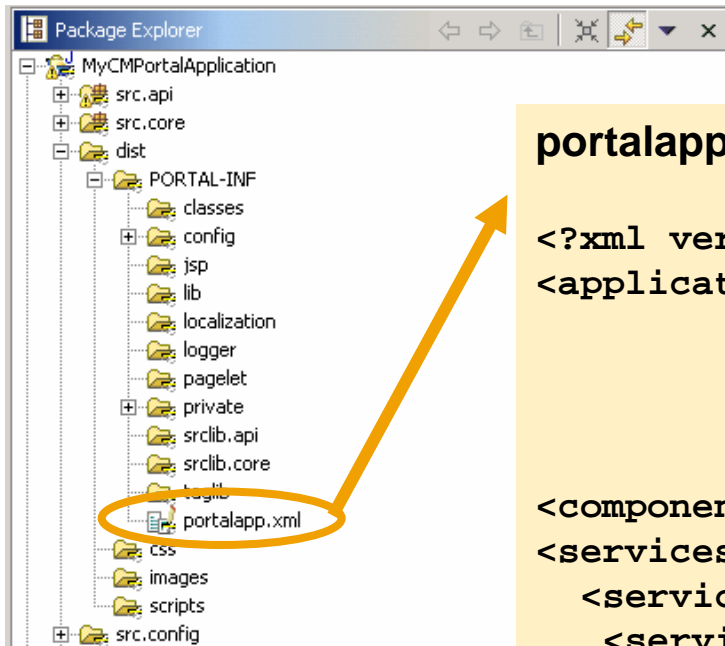
1. Class definition for this namespace filter (com.test.MyNamespaceFilter.cc.xml):

```
<ConfigClass name="com.test.MyNamespaceFilter"  
  extends="RepositoryFilter">  
  <attribute name="class" type="class"  
    constant="com.test.MyNamespaceFilterManager" />  
</ConfigClass>
```

Deployment using PAR Export Wizards

- **Standard portal deployment**
 - ◆ maintained portalapps.xml is mandatory
 - ◆ additional information can be provided in the public part of this PAR file
- **Creates a new wrapper service for each PAR file including KM components**
 - ◆ Service has no functionality, except of classloader registration
- **Configuration for configuration framework**
 - ◆ automatically bundle the archive containing
 - meta data (Classes definitions)
 - instances

Create / Check portal service configuration

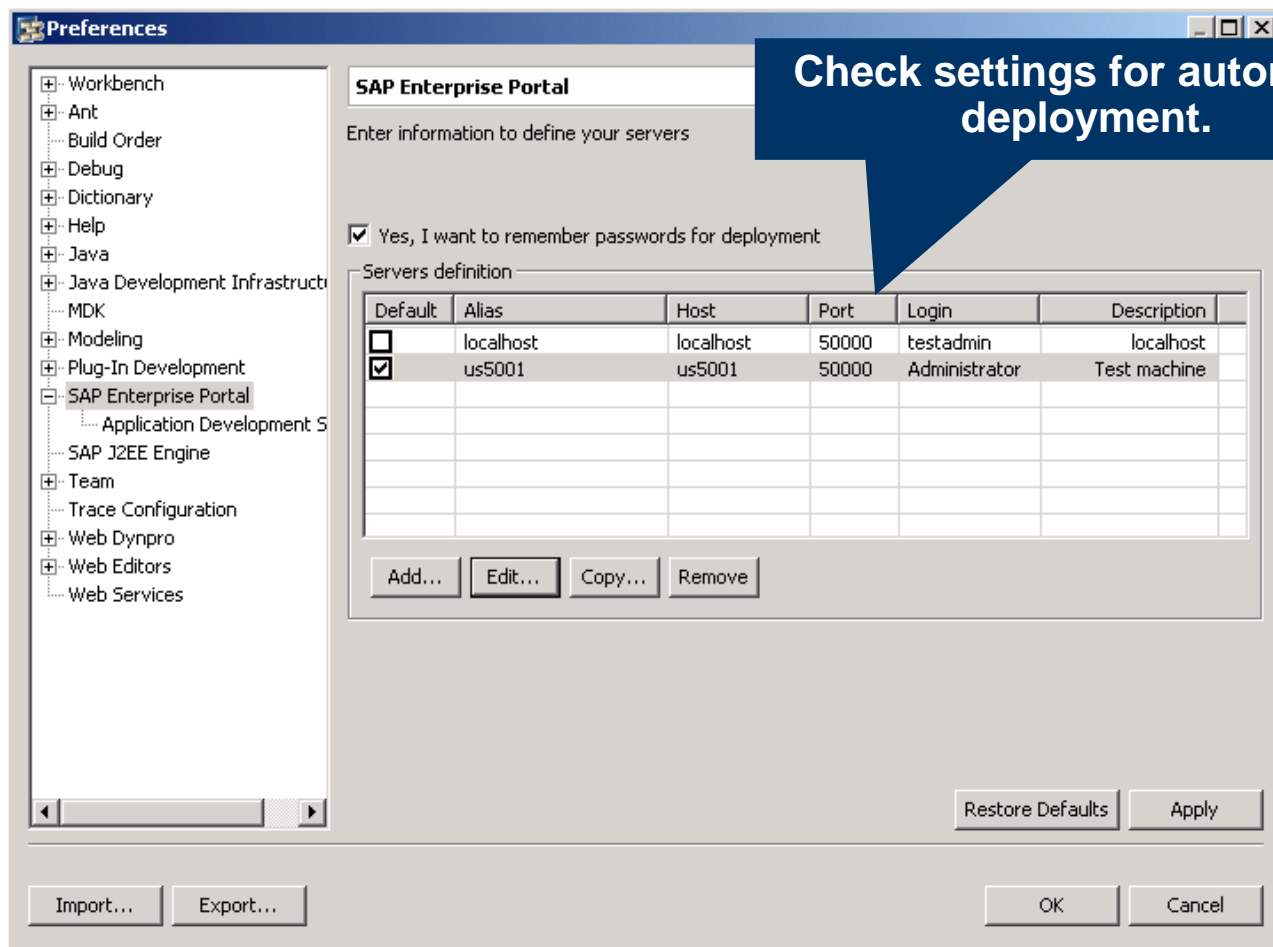


portalapp.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<application>
  <application-config>
    <property name="SharingReference"
      value="knowledgemanagement, ..."/>
  </application-config>
</application>
<components/>
<services>
  <service name="RFServiceWrapper">
    <service-config>
      <property name="className"
        value="com.sap.netweaver.rf.wrapper.RFServiceWrapper"/>
      <property name="startup" value="true"/>
    </service-config>
  </service></services>
</application>
```

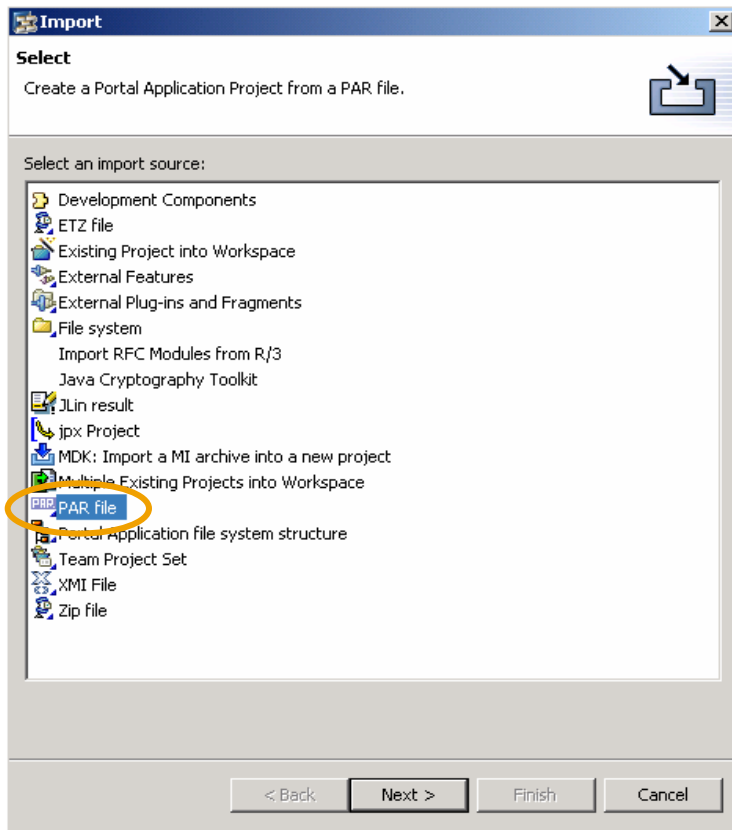
Server configuration – Connection to local/remote Portal

- From the menu, choose:
Window -> Preferences -> SAP Enterprise Portal



Create a PAR file

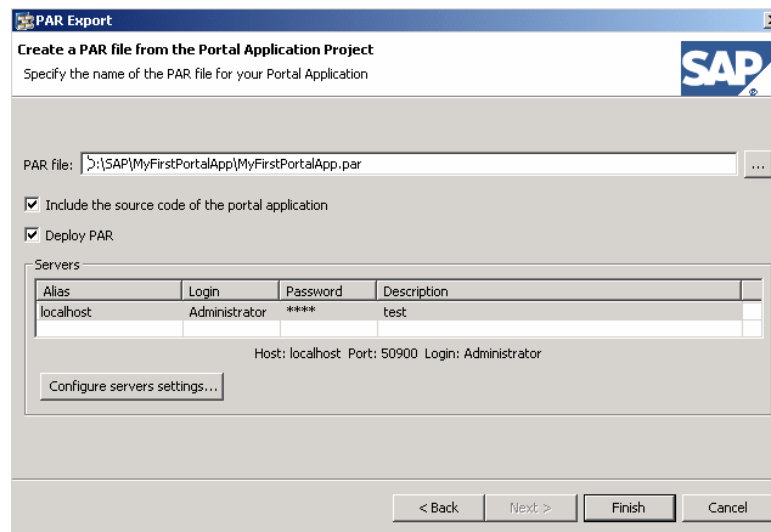
- From the menu, choose: **File** → **Export** → **PAR File** – click **Next**



Or choose the appropriate icon in the toolbar:



Or right-click your project and choose: **PAR Quick PAR Upload**



After deployment -> restarting the SAP J2EE Engine is needed

- No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.
- Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.
- Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.
- IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, and Informix are trademarks or registered trademarks of IBM Corporation in the United States and/or other countries.
- Oracle is a registered trademark of Oracle Corporation.
- UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.
- Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.
- HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.
- Java is a registered trademark of Sun Microsystems, Inc.
- JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.
- MaxDB is a trademark of MySQL AB, Sweden.
- SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.
- These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.