Portal Integration of Web Dynpro Applications

SAP NetWeaver 04
Icons in Body Text

<table>
<thead>
<tr>
<th>Icon</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>🚧</td>
<td>Caution</td>
</tr>
<tr>
<td>📚</td>
<td>Example</td>
</tr>
<tr>
<td>💡</td>
<td>Note</td>
</tr>
<tr>
<td>📚</td>
<td>Recommendation</td>
</tr>
<tr>
<td>⚙️</td>
<td>Syntax</td>
</tr>
</tbody>
</table>

Additional icons are used in SAP Library documentation to help you identify different types of information at a glance. For more information, see Help on Help → General Information Classes and Information Classes for Business Information Warehouse on the first page of any version of SAP Library.

Typographic Conventions

<table>
<thead>
<tr>
<th>Type Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example text</td>
<td>Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options.</td>
</tr>
<tr>
<td>Example text</td>
<td>Cross-references to other documentation.</td>
</tr>
<tr>
<td>Example text</td>
<td>Emphasized words or phrases in body text, graphic titles, and table titles.</td>
</tr>
<tr>
<td>EXAMPLE TEXT</td>
<td>Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.</td>
</tr>
<tr>
<td>Example text</td>
<td>Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.</td>
</tr>
<tr>
<td>Example text</td>
<td>Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.</td>
</tr>
<tr>
<td>&lt;Example text&gt;</td>
<td>Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.</td>
</tr>
<tr>
<td>EXAMPLE TEXT</td>
<td>Keys on the keyboard, for example, F2 or ENTER.</td>
</tr>
<tr>
<td>Topic</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Portal Integration of Web Dynpro Applications</td>
<td>5</td>
</tr>
<tr>
<td>Running a Web Dynpro Application on the Portal</td>
<td>6</td>
</tr>
<tr>
<td>Define a System</td>
<td>6</td>
</tr>
<tr>
<td>Create a Web Dynpro iView Template</td>
<td>7</td>
</tr>
<tr>
<td>Create a Web Dynpro iView</td>
<td>9</td>
</tr>
<tr>
<td>Define Role Based iViews</td>
<td>13</td>
</tr>
<tr>
<td>Enterprise Portal Client Framework (EPCF)</td>
<td>15</td>
</tr>
<tr>
<td>Subscribe to a Client Event</td>
<td>15</td>
</tr>
<tr>
<td>Unsubscribe a Client Event</td>
<td>16</td>
</tr>
<tr>
<td>Raise a Client Event</td>
<td>16</td>
</tr>
<tr>
<td>Enterprise Portal Navigation</td>
<td>16</td>
</tr>
<tr>
<td>Absolute Page Navigation</td>
<td>17</td>
</tr>
<tr>
<td>Relative Page Navigation</td>
<td>19</td>
</tr>
<tr>
<td>Enterprise Portal Object Based Navigation (OBN)</td>
<td>20</td>
</tr>
<tr>
<td>Triggering Object Based Navigation</td>
<td>20</td>
</tr>
<tr>
<td>Using the IUserObjectBasedNavigation Service</td>
<td>21</td>
</tr>
<tr>
<td>Accessing an Enterprise Portal Service</td>
<td>25</td>
</tr>
<tr>
<td>Using the Work Protect Mode</td>
<td>27</td>
</tr>
<tr>
<td>Defining the Theme for Web Dynpro Applications</td>
<td>28</td>
</tr>
</tbody>
</table>
Portal Integration of Web Dynpro Applications

SAP Enterprise Portal supports a number of functions you can use to develop your Web Dynpro applications. This includes event handling of portal events, navigation between Web Dynpro applications within the portal or to any portal content, and the use of the WorkProtect mode.

The SAP Enterprise Portal and Web Dynpro for Java are strategic user interface technologies of SAP. Both are part of the Java technology offer from SAP and run on the Web Application Server. They are closely linked.

Interaction between Portal and Web Dynpro for Java

You can use Web Dynpro to create professional, highly interactive Web-based user interfaces for business applications. The Portal allows the role-based and secure access to any kind of information (structured or non-structured), services, and applications using a Web Browser. These two technologies are used in SAP NetWeaver to execute the content created by Web Dynpro in the Portal.

Portal application developers who use Web Dynpro

When creating a Portal content, you can use several tools for the creation of professional roles and targets. The assistants and templates of the Portal Content Studio are designed for content administrators. The Portal Development Kit (PDK) for Java is to be used by J2EE developers.

Web Dynpro, however, is designed for developers of business applications. Since application developers work with the SAP NetWeaver Developer Studio based on Eclipse, they only need to have basic Java knowledge for the use of the programming model. They can integrate their own Web Dynpro applications into the Portal.

Integration of Web Dynpro applications into the Portal

The Portal is the recommended environment for the execution of Web Dynpro applications.

When integrating the Web Dynpro applications into the Portal, the administrator can use several functions especially developed for this purpose. You have the following options:

- Using a template especially developed for the creation of Web Dynpro iViews.
- Accessing your own Web Dynpro administration tools. You can use the Web Dynpro Content Administrator for the Portal content administration.
- Configuring client-side eventing of the Web Dynpro applications among each other and between them and other Portal content.
- Creating single sign-on between Portal, Web Dynpro, and back-end applications.
- Activating the protection against data loss using the WorkProtect mode of the Portal when executing Web Dynpro applications.

Portal and Web Dynpro for Java complement one another at runtime. In the Portal:

- the navigation within the Web Dynpro-based iViews as well as between them and other Portal content (including object-based navigation) is supported.
- Web Dynpro applications automatically use the currently set Portal display theme to ensure a consistent appearance.
Running a Web Dynpro Application on the Portal

Purpose

This describes how to call a Web Dynpro application from an SAP Enterprise Portal iView and to create a Web Dynpro iView with a dedicated Web Dynpro iView template to call a Web Dynpro application remotely using HTTP.

The Web Dynpro runtime uses the currently active portal theme automatically. If you start the Web Dynpro application outside the portal the standard SAP theme is used.

If a customer has defined his own portal theme using the Style Sheet Editor the CSF styles required are also generated automatically. However you can also configure the theme to be used when the Web Dynpro application is called outside the portal.

Prerequisites

- For the Web Dynpro integration a SAP Enterprise Portal 6.0 SP2 or higher is required.
- An up to date version of the SAP WebAS is installed that contains the Web Dynpro runtime environment.
- You are familiar with the Web Dynpro Tools and have already developed a Web Dynpro application.

Features

The integration contains the following features:

- You can create Web Dynpro-based iViews using a special Web Dynpro iView template.
- Web Dynpro applications are called remotely using a URL.
- The Web Dynpro application automatically uses the current portal theme.
- Application parameters can be passed to the Web Dynpro application as startup parameters.

Define a System

Use

In this integration scenario the SAP Enterprise Portal and the Web Dynpro application run on different servers. This means that you have to define the system on which the Web Dynpro application is running and create an alias for the system.

Procedure

Creating a folder and defining a system

Define the system on which the Web Dynpro application is running as follows:

1. In the portal, navigate to System Administration → System Configuration → System Landscape.
2. In the Portal Catalog, navigate to the folder in which you want to create the system. You can create a new folder, if needed.
3. Click a folder in the Portal Catalog with the secondary mouse button. A context menu is displayed.

4. In the context menu, choose New → System. The System Landscape Wizard is launched.

5. Select SAP backend system with Load Balancing from the Template selection window screen and choose Next button.

6. Provide a system name, system ID and prefix for your system. The names are arbitrary, for example:
   a. System Name, for example WD_SystemName
   b. System ID, for example WD_SystemID
   c. System ID Prefix, for example com.sap
   d. Save As “System”
   e. Choose the Master Language

7. Choose Next. On the next screen all entries will be displayed in a summary that allows you to check the entries before finishing. If changes need to be made, choose Back to navigate the relevant screen. Then you can make necessary changes.


**Defining an Alias for the System**

To create the alias for the system, first make sure that the new system is open in the System Landscape Editor. Proceed as follows:

1. In the Display dropdown list box, choose System Aliases.

2. Specify a name for the system alias, for example MySystem. Add the defined alias by choosing Add.

3. To save your changes, choose Save.

**Defining the Properties Specific to the Web Application Server**

1. Open the system in the System Landscape Editor.

2. In the Property Category dropdown list box in the Property Editor, choose Web Application Server (WAS).

3. Maintain the following properties:
   a. WAS Host Name = <host name>.<domain name>:<port number> (this corresponds to the Web Dynpro server location and the port on which it is running, for example, PXYZ.wdf.sap.corp:50000.)
   b. WAS Protocol = http or https (depending on the configuration of the WebAS)

4. Save the settings.

🚀 **Create a Web Dynpro iView Template**

**Use**

To run a Web Dynpro Java or Web Dynpro ABAP application within the SAP Enterprise Portal you have to create specific iViews. These Web Dynpro-based iViews can be handled in the same way as any other iView.
**Procedure**

Use the standard Web Dynpro iView template to create a new Web Dynpro-based iView.

If you wish to create your own Web Dynpro iView template, switch to the content catalog and create a folder for your Web Dynpro portal content.

1. Select the context menu and choose *New From PAR → iView*.


Result
The Web Dynpro iView template is generated.
The template is not necessary to run a Web Dynpro application on the portal but if you want
to create a set of iViews with the same settings the iView template is the a “Master iView”. Make sure that your specific iView template is also part of your shipped portal content.

Create a Web Dynpro iView

Use
Using the standard Web Dynpro iView template, or your specific iView template, define your Web Dynpro iViews.

Procedure
1. Switch to the content catalog and select the folder that contains your new Web Dynpro iView. Use the context menu and select New → iView.

2. The Web Dynpro iView Wizard is displayed in the right window. Choose the SAP Web Dynpro iView template or the template you created (see Create a Web Dynpro iView Template [page 7]).
3. In Step 2 of the Web Dynpro iView Wizard, you have to specify the iView Name and iView ID. The iView Name is displayed as title of the Web Dynpro iView. The iView ID is the technical name and the iView ID Prefix can be used to define your own namespace. You can also specify the Master Language and the description of the iView.

4. In Step 3 of the Web Dynpro iView Wizard, choose the application variant of the iView. You can choose between a Web Dynpro Java or a Web Dynpro ABAP application.
5. In Step 4 of the Web Dynpro iView Wizard you have to specify the Web Dynpro-specific parameters.

   a. System

   Specify the system alias of the system you want to use for your Web Dynpro application. In SAP NetWeaver every Web Dynpro-based iView runs isolated on a page. The iView uses a specific Web Dynpro application URL to start the Web Dynpro application. The system definition is needed to compute the URL.

   A system alias has to exist for every SAP NetWeaver installation to run a Web Dynpro application. If your Web Dynpro application runs on the same installation as the portal itself, you can use the predefined system alias SAP_LocalSystem.

   Currently, there is no system type definition available for a SAP NetWeaver installation. Therefore, you must use one of the available SAP backend system types. You have to define only the WAS hostname and WAS Protocol parameters in the system definition. The WAS hostname must be in the form: <host>.<domain>:<port>.

   b. Namespace

   Define the namespace containing your Web Dynpro application. The semantic behind this namespace depends on the used Web Dynpro variant:

   - Java
Using a Web Dynpro Java application, the namespace is the name of the used development component or the name of the Eclipse project defining the Web Dynpro Java application.

If you do not use JDI/DTR, you have to specify the Eclipse project defining your Web Dynpro Java application. Make sure that both a development component name and the Eclipse project name contain the vendor name. Possible values are sap.com/tc-wd-tools (for a development component) or local/MyTestProject for an Eclipse project.

- ABAP.

Using a Web Dynpro ABAP application, the namespace is the development class of the Web Dynpro application.

The specified namespace is the same name as displayed in the SAP WebAS console using the LIST_APP command or as used in the Web Dynpro application URL.

c. Application Name

The application name is the name of the Web Dynpro application, for example MyTestApplication

d. Application Parameters

If your Web Dynpro application has startup parameters, you can define specific values for this iView.

Application has two parameters: par1 and par2

The parameters are specified as follows:

parameter1=value1&parameter2=value2
6. Now you can define the standard iView parameters with the iView Wizard. If you use the standard Web Dynpro iView template, your Web Dynpro iView has automatic resizing. If you want to define a fixed height or a full-screen iView, you have to adjust this parameter.

**Result**

The Web Dynpro iView is now ready to use. You can add the iView to pages, worksets, or roles.

---

**Define Role Based iViews**

The portal treats Web Dynpro iViews like any other iView, so the role definition not different. With the delta-link mechanism it is possible to configure a Web Dynpro iView (one or more of the iView parameters) for a specific page, workset, or role.

The set of parameters that defines the used configuration is application-specific. With the Web Dynpro iView Wizard any set of these parameters can be defined in the Application Parameters dialog window as iView parameter (see Create a Web Dynpro iView [page 9]).

**How to Define Role-Based Pattern Applications**

In the SAP NetWeaver version of the pattern engine, the configuration is defined with the parameter `configurationId`. By calling the pattern engine with different values of the `configurationId` parameter, the pattern engine executes the different scenarios.

Therefore, you have to specify the `configurationID` using the Application Parameter dialog window of the iView parameters. The value can be customized using the delta-link-mechanism and becomes a role-based pattern application.

**Enable Single Sign-On (SSO) for a Web Dynpro iView**

With SSO the user logs on to the portal once and does not have to log in to every iView later on.

SSO for a Web Dynpro-based iView can use one of the following mechanisms:

- Ticket authentication

  The following steps are necessary to switch to ticket authentication:

  - You must maintain the definition of the system running the Web Dynpro application. You have to define the logon method and the user mapping type:
The ticket provided by the portal system is accepted by the system running the Web Dynpro application.

The portal and the Web Dynpro system have to have the same set of users. Either both systems use the same user store or the users have to be manually added on both systems.

User mapping

The following steps are necessary to switch to user mapping:

- You must maintain the definition of the system running the Web Dynpro application. You must define the logon method and the user mapping type:

  - Logon Method: UID/PW
  - User Mapping Fields
  - User Mapping Type: admin USER

- The portal user must log in (on the system running the Web Dynpro application) during the start of the iView. To prevent this, the user can define the following logon data using the User Mapping section of the personalization dialog:

  - System: testDynpro
  - User: "
  - Password:"

The administrator can set this user mapping for a whole user group or for specific roles.

As user mapping does not need as much configuration effort as the ticket authentication, this is very useful for test environments. For production scenarios, you should normally use the ticket authentication.
Enterprise Portal Client Framework (EPCF)

Purpose

In the SAP Enterprise Portal, different application types in specific iViews can be arranged on one page. To communicate between the different iView types the portal provides the Enterprise Portal Client Framework (EPCF), also known as client-side eventing. This document describes how Web Dynpro applications can use EPCF.

Restrictions

Several Web Dynpro applications running on one portal page can communicate using client events. It is recommended, to use client-side eventing only for "occasional" communications between Web Dynpro applications. For Web Dynpro applications that have to interact more frequently, a "full-screen" Web Dynpro application containing all the components has to be implemented.

With SAP NetWeaver '04 Stack 09, client eventing is only supported for the Web Dynpro HTML client.

Subscribe to a Client Event

The communication between iViews, including Web Dynpro applications, is based on EPCF [extern]. EPCF uses Javascript to allow iView communication and provides an API that can be used by the portal application developer. Web Dynpro applications have to use a set of Java wrapper methods to implement client-side eventing.

It is possible to subscribe or unsubscribe to certain client events. To do so, you must define which Web Dynpro action is to be used as the event handler for the portal event. You can also fire any portal event.

Because of Javascript restrictions, all participants (the portal server and all used servers) have to be in the same domain when EPCF is used.

The following example demonstrates how to subscribe to a certain portal event.

```javascript
WDPortalEventing.subscribe ("urn:com.sap.tc.webdynpro.test.portal", "TestEvent", wdThis.wdGetTestEventAction());
```

You have to define the name space of the event and the name of the event. The combination of these two names must be unique.

The third parameter is the Web Dynpro action that is mapped to the portal event. The event handler is called when the Web Dynpro application receives the specified portal event on the client. The Web Dynpro HTML Client handles the mapping between a portal event and a Web Dynpro action and is absolutely transparent for the Web Dynpro application developer.

You can reuse a Web Dynpro action for several portal events. If you want to receive the transported data of the portal event, you can define the following parameters for your Web Dynpro action:

- **dataObject**
  This parameter contains the transported parameter of the portal event.
- **Namespace**
  This parameter contains the name space of the received portal event.

- **Name**
  This parameter contains the name of the received portal event.

Adding the nameSpace and name parameters to the Web Dynpro action is useful when the action is reused for several portal events to distinguish the caller.

![Important Note]
It is important to remember that in the current version, an event subscription is valid for a Web Dynpro view. Therefore you should add the necessary Java coding, for example, in the `wdDoInit()` method of the generated view class. If you navigate between different views, you have to subscribe every view for the event.

### Unsubscribe a Client Event

Unsubscribing a client event is very similar to subscribing:

```java
WDPortalEventing.unsubscribe("urn:com.sap.tc.webdynpro.test.portal", "TestEvent", wdThis.wdGetTestEventAction());
```

Make sure that you unsubscribe every single Web Dynpro view, as the subscription and unsubscription is valid only for the current view.

### Raise a Client Event

The following example demonstrates how to raise a portal event:

```java
WDPortalEventing.fire ("urn:com.sap.tc.webdynpro.test.portal", "TestEvent", "AParameter");
```

You can fire a portal event at any time in your Web Dynpro application. The event is transported with the next response to the client. You can also raise more than one portal event in one request-response cycle. Normally, you will fire a portal event in a Web Dynpro action event handler (for example, pressing a button).

### Enterprise Portal Navigation

#### Purpose

The SAP Enterprise Portal structures the content based on roles. The portal navigation uses the Top Level Navigation (TLN) component and/or the Detailed Navigation Component.
Every application running as portal content (ipage or iView) can trigger the portal navigation to iViews or pages.

The integration of the portal navigation features into Web Dynpro is very similar to the client-side eventing [page 15]. Portal application use a Javascript API for the portal navigation. The Web Dynpro runtime offers a The WDPortalNavigation service provides methods to use the portal navigation and gives access to the parameters. The WDPortalNavigation service is a generic part of the Web Dynpro runtime.

### Absolute Page Navigation

The WDPortalNavigation service provides `navigateAbsolute()` methods for absolute page navigation. The methods have the following parameters:

- **navigationTarget**
  Specifies the absolute target URL (i.e. the URL to the iView or page acting as navigation destination). This absolute target URL points to the location of the iView or page in the Portal Content Directory (PCD) structure. You have to use the prefix ROLES://.

  **Absolute target URL example:**

- **mode**
  Specifies if the defined navigation target is displayed in the same browser window or in a new one. The following modes are available:

  - **WDPortalNavigationMode.SHOW_INPLACE**
    The navigation destination is displayed in place, meaning, in the same browser window. The Top Level Navigation (TLN) and the Detailed Navigation (DTN) are updated accordingly.

    You can use the BACK/FORWARD functionality of the page header to navigate back.

  - **WDPortalNavigationMode.SHOW_EXTERNAL**
    The navigation destination is displayed in a new browser window that has no portal frame. Only the specified iView or page is displayed.

  - **WDPortalNavigationMode.SHOW_EXTERNAL_PORTAL**
    The navigation destination is displayed in a new browser window with the standard portal frame (containing TLN and DTN). The TLN and DTN are updated accordingly. The new browser window clears the navigation history so you cannot use the BACK/FORWARD functionality of the page header to navigate.

- **historyMode**
  The history mode defines how the navigation step is visible in the navigation history as part of the page header.
The following options are available:

- **No history**
  The navigation step is not visible in the navigation history. You cannot use the BACK/FORWARD functionality to navigate back.

- **No duplications**
  The navigation step is visible in the navigation history. If there are several navigation steps to the same navigation destination, they are only visible once in the navigation history.

- **Allow duplications**
  The navigation step is visible in the navigation history. If there are several navigation steps to the same navigation destination, but with different parameters, they are visible as different entries in the navigation history.
  
  This option is useful when you navigate to an iView displaying details of a customer. The customer ID can be a parameter. This results in different entries in the navigation history.

  Using the **History Title** parameter as described below, you can define the title of the entry in the navigation history, like “Details for Customer 4711” and “Details for Customer 007” to explain that the same navigation destination is used but with different parameters.

- **targetTitle**
  The targetTitle defines the title of the entry in the navigation history.

- **contextURL**
  Defines the “navigation context” in which the navigation destination should be displayed. The context URL is useful if your specified navigation destination is not visible in TLN or DTN. If you trigger navigation steps to an invisible navigation destination the TLN and DTN is not adjusted. To update the TDN and DTN in this case, you can use the context URL.
  
  The context URL has the same format as the absolute target URL.

- **windowFeatures**
  If you start the navigation destination in a new browser window, you can define the used window features, like the window size, the window position, or the visible button bars.

- **windowName**
  If you start the navigation destination in a new browser window, you can define the name of this new browser window.

- **launcherParameters**
  You can define any parameters for a navigation step. When navigating to SAP-based portal content, you can define parameters that should only be passed to the portal component, the AppIntegrator, to launch the specific SAP content, like a Web Dynpro application, a BSP application or an IAC-based on the Internet Transaction Server (ITS).
  
  Example how to specify launcher parameters:
Depending on the defined parameter values you have to encode the parameter values before assembling the `launcherParameters` string.

- **businessParameters**
  If you want to define parameters that are passed to the SAP content itself, you have to define them as business parameters.
  Example:

  ```
  CustomerID=4711&DisplayMode=Edit
  ```

  Depending on the defined parameter values you have to encode the parameter values before assembling the `businessParameters` string.

- **useSAPLauncher**
  Boolean value. If your navigation destination is content based on SAP technology (i.e. BSP, ITS, BI, or Web Dynpro Java or ABAP) you have to set this flag to “true”.

- **postParameters**
  Boolean value. If you have several parameters it can be necessary to post the parameters because of URL length restrictions. Setting this flag to “true” will post the parameters.

---

**Relative Page Navigation**

Using an absolute target URL can cause problems if you move the navigation destination. When you navigate from one page to another page stored in the same content folder, the relative navigation will work even when the entire folder has been moved.

For the relative page navigation the `WDPortalNavigation` service offers the `navigateRelative()` methods. It has following parameters:

- **baseUrl**
  This is the starting point of the relative navigation destination. The base target URL has the same format as the absolute target URL of an absolute navigation.

  Example of a base URL:

  ```
  ```

- **levelsUp**
  You have to define how many levels in the navigation hierarchy you have to go up for the relative navigation.

- **path**
  The relative navigation path describes the path to the navigation destination relative to the defined base target URL and the defined number of levels up.

All other (optional) parameters are the same as for the absolute page navigation [page 17].
Enterprise Portal Object Based Navigation (OBN)

Purpose
With the standard portal navigation you define a constant URL. With the OBN you can define an "operation" of a business object.

Example:
You define that you want to trigger the "Display" operation of a "Customer" business object. The specific iView or page that is used to realize or implement this operation is configured within the portal platform and could be role-specific or even user-specific.

For more information about OBN, please refer to the portal OBN documentation.

Like the portal navigation, the Web Dynpro integration of the OBN feature is very similar to the portal eventing integration. The Web Dynpro runtime offers a specific WDPortalNavigation service to define the necessary parameters.

Triggering Object Based Navigation

The WDPortalNavigation service allows access any page or object-based navigation functionality and parameters from a Web Dynpro application. The WDPortalNavigation service is a generic part of the Web Dynpro Runtime.

Triggering Object-Based Navigation

The WDPortalNavigation service provides the navigateToObject() methods with following parameters:

- system
  You have to specify the system (alias) to which the business object is assigned. This is a mandatory parameter.
- businessobjType
  You have to define the business object using this mandatory parameter.

Optional parameters are:

- objValue
  Normally there are many different instances for one business object.
  Example:
  A business object called Customer. To specify which customer should be used for the object navigation step, you have to specify the object value which is the customer ID.
- operation
  Specifies which operation should be used for the object navigation step.
- objValueName

Portal Integration of Web Dynpro Applications 20
The specified object value is passed as a single URL parameter to the object navigation step. The default name of the parameter is `ObjectValue`. Other parameter names can be specified.

- **businessParameters**
  
  Additional parameters can be specified.
  
  Example:
  
  ```
  Mode=Edit&ShowHeader=false.
  ```

**Parameters for the object-based navigation destination.**

- **forwardOBNMetaData**
  
  With this parameter you can get more information about the current object navigation step.
  
  Example:
  
  An application implements different operations of a business object.
  
  To do that the application has to know which operation was triggered by the object navigation step.
  
  Following parameters are possible:
  
  - `obn.system`
    
    The system that the business object is assigned.
  
  - `obn.bo_type`
    
    The business object itself.
  
  - `obn.operation`
    
    The triggered operation. If the default operation is triggered, the value is `_default_`.

---

**Using the IUserObjectBasedNavigation Service**

With the `WDPortalNavigation` service you can trigger an object-based navigation for a operation of a business object. To create a more user friendly user interface you can use the `IUserObjectBasedNavigation` service.

**Defining the Necessary Reference**

The `IUserObjectBasedNavigation` service is provided by the portal platform. You have to define a specific sharing reference in the portal application descriptor file of the Web Dynpro application using this service. The following screenshot demonstrates the definition of this sharing reference.
For more information about how to access a portal service see [Accessing an Enterprise Portal Service][25].

**Operations**

This final target URL depends on the configuration for the current role or user. As a result it is possible that for the currently logged in user and his role, there is no target. Such an operation is invalid. A valid operation is an operation with a defined destination, meaning, there is an iView or page in one of the user roles implementing the requested operation).

**Checking a Target**

The following code example demonstrates how to check if an operation is valid:

```java
// Get a reference to the OBN service
IUserObjectBasedNavigation obnService = (IUserObjectBasedNavigation) WDPortalUtils.getServiceReference(IUserObjectBasedNavigation.KEY);

// Get the current user
IUser user = null;
try {
    user = WDClientUser.getCurrentUser().getSAPUser();
} catch (WDUMException e) {
    wdComponentAPI.getMessageManager().reportException("Failed to get current user: ", true);
}

// Define the system and the business object
String system = "MySystem";
String bo = "customer";

// Call the service
boolean hasValidDefaultOperation = obnService.isTargetExist(system, bo, user);
```

You can use this function to make sure you render the right UI element depending on the configuration for the current user, or to enable or disable a LinkToAction UI element that is used to trigger the object-based navigation for the default operation.
Checking a Target for an Operation

The following code example shows how to check if a certain operation has a valid target:

```java
// Get a reference to the OBN service
IUserObjectBasedNavigation obnService = (IUserObjectBasedNavigation) WDPortalUtils.getServiceReference(IUserObjectBasedNavigation.KEY);

// Get the current user
IUser user = null;
try {
    user = WDClientUser.getCurrentUser().getSAPUser();
} catch (WDUMException e) {
    wdComponentAPI.getMessageManager().reportException("Failed to get current user: " + e.getLocalizedMessage(), true);
}

// Define the system, the business object and the operation
String system = "MySystem";
String bo = "customer";
String operation = "Display";

// Call the service
boolean operationHasValidTarget = isTargetExistsForOperation(system, bo, operation, user);
```

Getting a List of Valid Operations

The IUserObjectBasedNavigation service can be used to get the list of valid operations for a business object.

Example:
// Get a reference to the OBN service
IUserObjectBasedNavigation obnService = (IUserObjectBasedNavigation) WDPortalUtils.getServiceReference(IUserObjectBasedNavigation.KEY);

// Get the current user
IUser user = null;
try {
    user = WDClientUser.getCurrentUser().getSAPUser();
} catch (WDUMException e) {
    wdComponentAPI.getMessageManager().reportException(
        "Failed to get current user: " + e.getLocalizedMessage(), true);
}

// Define the system and the business object
String system = "MySystem";
String bo = "customer";

// Get the list of valid operations
List operations = obn.getTargets(system, bo, user);

// Fill dynamically a context node with the operation information
// The list can be displayed in a DropDownByIndex control.
IMyTestView.IOperationsElement newOperation = null;
for (Iterator iter = operations.iterator(); iter.hasNext();) {
    IOBNTarget target = (IOBNTarget) iter.next();
    newOperation = wdContext.nodeOperations().createOperationsElement();
    newOperation.setCaption(target.getOperationFriendlyName());
    newOperation.setName(target.getOperationName());
    wdContext.nodeOperations().addElement(newOperation);
}

Using a Web Dynpro iView as a Target

To make sure that the forwarding of parameters works for the Web Dynpro iView you have to change the Javascript code that is used by the object-based navigation to define the object value manipulation.

For every operation where your Web Dynpro iView is the target (or implementation), you have to define the following Javascript code:

    return \'DynamicParameter\'=\' + objValue;

Example:
Accessing an Enterprise Portal Service

Purpose

Portal services can be used by portal applications, for example, Enterprise Portal Client Framework (EPCF) or the connector framework.

Referencing a Portal Service

Every portal service used in a Web Dynpro application has to be defined as sharing reference. To do so, open the “Properties” dialog of your Web Dynpro development component or Web Dynpro Eclipse project when you work with DTR.

The sharing reference must be defined as follows:

```
PORTAL:<Vendor name>/<Full qualified name of the portal service>
```

Example:
After defining the sharing reference the jar file containing the service has to be added to the classpath of the project in the IDE.

⚠️ Do not store the jar file of the service in the lib folder of your development component or Eclipse project. This will cause classloader problems at runtime.

**Gaining Access to the Portal Service**

The following code example demonstrates how to get access the IUserObjectBasedNavigation portal service:

```java
IUserObjectBasedNavigation obnService = (IUserObjectBasedNavigation)
    WDPortalUtils.getServiceReference(IUserObjectBasedNavigation.KEY);
```

If there is no more than one Web Dynpro controller using the same portal service, you must store this portal service reference in a certain context node, for example, in the component controller context, or any other custom controller context. All controllers using this service must have access using context mapping to the specified context node.
Restrictions
A portal service can be used only when the portal platform is installed and the Web Dynpro application and the portal platform run on the same machine.

Even if you use a complete SAP NetWeaver installation (including the portal platform), you always have to use the local portal service. It is not recommended to use your Web Dynpro-based portal content within a global or federated portal scenario.

Using the Work Protect Mode

Purpose
To avoid losing unsaved data when navigating from one iView to another, the SAP Enterprise Portal provides the work-protect mode. Having unsaved data in an application is also called “dirty state”.

Using the Work-Protect Mode
A Web Dynpro application can use the work-protect mode in three levels:

- None
  The work-protect mode is not used by the Web Dynpro application.

- Application Only (Default)
  Only the Web Dynpro application itself decides if the application has unsaved data. Using the specific work-protect mode Java wrapper class, the Web Dynpro application developer can define the “dirty state” of the application. The “dirty state” is therefore only defined on the server-side. In this level data still can be lost when it was not transported to the server.

- Standard
  The Web Dynpro application and the Web Dynpro HTML Client check the application state. Therefore both the application developer and the Web Dynpro HTML Client check the “dirty state” of a Web Dynpro application. The Web Dynpro HTML Client makes sure that no user input data that is typed in but not transported to the server is lost, by setting the “dirty state” of the application in the portal as soon as the user makes an input.

Example: Defining the work-protect mode level.

```java
// Define the needed level. The level can be switched during runtime, for example be switched for different views.
WDPortalWorkProtectMode.setApplicationDirtyControl(WDApplicationIsDirtyMode.NONE);
WDPortalWorkProtectMode.setApplicationDirtyControl(WDApplicationIsDirtyMode.APPLICATION_ONLY);
WDPortalWorkProtectMode.setApplicationDirtyControl(WDApplicationIsDirtyMode.STANDARD);
```

Example: Define the “dirty state” of an application:
// Set the "dirty state" to YES, i.e. the application is dirty and
// a navigation should be launched in a new window to make sure that
// no data is lost
WDPortalWorkProtectMode.setApplicationDirty(true);

// Set the "dirty state" to FALSE, i.e. the application state is "clean"
// and therefore the navigation is launched in place and the
// running application is left.
WDPortalWorkProtectMode.setApplicationDirty(true);

### Restrictions

For SAP NetWeaver '04 Stack 09 the following restriction applies:

- The work-protect mode works only for the Web Dynpro HTML Client. All other (rich) clients do not support the work-protect mode.

---

#### Defining the Theme for Web Dynpro Applications

**Purpose**

The Web Dynpro application uses automatically the currently selected portal and any user or role-based personalization of the theme.

However, if the Web Dynpro application and the portal are running on different systems with different releases, problems with incompatible style sheets can occur when the portal system has an older release than the system running the Web Dynpro application. This causes errors in the behavior of some UI elements of the Web Dynpro application because the Web Dynpro UI elements also depend on the used themes/style sheets.

Therefore, it is possible to configure the Web Dynpro runtime to use the defined Web Dynpro theme instead of the portal theme. The Web Dynpro application is then rendered in a different theme as the portal, but there is no dependency of the style sheets anymore.

With the Visual Admin tool you can adjust the setting to prevent the Web Dynpro runtime from using the portal theme. Choose **Configuration Adapter service** and navigate to:

```
webdynpro → sap.com → tc~wd~dispwda → Propertiesheet default.
```

You have to open this default property sheet and change one property value called:

```
sap.useWebDynproStyleSheet.
```

The possible values are:

- **true**
  
  The Web Dynpro application does not use the portal theme but the theme defined by the Web Dynpro runtime.

- **false**

  The Web Dynpro application uses the theme defined by the portal. This is the default for SAP NetWeaver '04 Stack 09.

The following screenshot shows the editor for changing this property value.
Running as a Standalone Application

If the Web Dynpro application runs as a standalone application (not in the SAP Enterprise Portal), or if the Web Dynpro runtime is configured to use the Web Dynpro theme, you have two options to define the theme used by a Web Dynpro application.

Defining the Default Theme

The default theme of the Web Dynpro runtime is changed in the Configuration Adapter service of the Visual Admin tool. Navigate to:

```
webdynpro → sap.com → tc~wd~dispwda → Propertiesheet default.
```

Select the property `sap.theme.default` of the Propertiesheet. As value, you have to specify a valid, full qualified URL referring the used theme in the form:

```
http://<Your host>:<Your port>/<Your path to the theme>/<Theme name>
```

Example:

```
service/themes/portal/sap_chrome.
```

Defining an Application-Specific Theme

If you would like to define a theme only for one specific application, you can use the URL parameter `sap-cssurl` to define the theme. This overrides the default settings. The URL specified has to be valid, full qualified and encoded.