Web Dynpro Tutorial: Hello World

Summary
This is the Web Dynpro Java version of a "Hello World" tutorial. Typically used to enable the developer to do the first steps within a programming language or development environment.

Author: Chris Whealy
Company: SAP AG
Created on: 13 September 2007

Author Bio
Chris Whealy has more than 20 years experience on Computer Science. He has worked with SAP software since 1993 and specifically with web based interfaces since 1996. In late 2002, Chris turned his attention to Web Dynpro and by October 2003, was able to put his knowledge into practice when he started work as the lead technical consultant for a large Web Dynpro based project. Chris is the author of the SAP Press book "Inside Web Dynpro for Java" published in November 2004 and both the SAP training courses on Web Dynpro for Java (JA310 and JA312)
# Table of Contents

**Summary** ................................................................. 1

**Author Bio** ............................................................. 1

**Development Objectives** .......................................... 3

**Graphics and screen shots** ........................................... 3

**Result** ........................................................................ 3

**Prerequisites** ............................................................ 3

**Development Steps** .................................................... 3

**Building, Deploying, and Running** .................................. 18

**Copyright** .................................................................... 20
**Development Objectives**

The first exercise is a version of the popular “Hello World” program.

You will write a very simple Web Dynpro application that places the text “Welcome to Web Dynpro” on the user’s browser.

At this early stage, the application will simply place some text on the screen and not be interactive.

**Graphics and screen shots**

Certain screen shots shown in this document may have been cropped in order to reduce the amount of empty image space and the overall document size. Only those areas of the screen that carry relevant information have been included in screen shots.

Therefore, it is entirely possible that the image you see on the printed page is different in size from the corresponding screen seen in the NetWeaver Developer Studio.

**Result**

After this exercise is complete, you should see the result shown on the left.

![Image of screen shot](image)

**Prerequisites**

You have the SAP NetWeaver 7.0 Java Server installed and running

You have the SAP NetWeaver Developer Studio (NWDS) installed and running.

Within the NWDS, you have selected the Web Dynpro Perspective (Window → Open Perspective → Other → Web Dynpro).

**Development Steps**

Create a new Local Development Component of type Web Dynpro.
According to the principles of SAP’s Component Model hierarchy, whenever you create a Development Component, it must be assigned to one (and only one) Software Component. Since it is very possible that any given installation of the SAP Java Server has not had its NetWeaver Development Infrastructure configured, SAP has supplied a dummy Software Component to which your Development Component can be assigned. This is always called “MyComponents”.

In the next screen, you need to enter the following values:

The name of your Development Component. The name can be divided into different sections separated with a forward slash character. This is not the same as the Java package name, but should be structured in a similar manner.

The Caption is optional, but useful when looking at long lists of Development Components.
Most importantly, you must decide which type of Development Component you want to create. In this case, it is a Web Dynpro Development Component.

![Image of Development Component creation](image)

Figure 3: Create the Development Component of type Web Dynpro

After you have pressed **Next** and then **Finish**, you will be returned to the Web Dynpro perspective of NWDS. Expand the Development Component hierarchy to reveal the items under the "Web Dynpro" branch.
All Web Dynpro development is based on the unit of software known as the “Web Dynpro Component”. It is unfortunate that the word “Component” is being reused here, but do not confuse a “Development Component of type Web Dynpro”, with a “Web Dynpro Component”. These are two different software entities!

The next thing you need to do is to create a Web Dynpro Component. In Web Dynpro development, the component is your unit of development and your unit of reuse. Therefore, if you want to create a functional Web Dynpro program, you must create at least one Web Dynpro Component.

Right mouse click on the tree item “Web Dynpro Components” and select “Create Web Dynpro Component”.

You will now see a pop-up window into which you must supply the details of your new component. These value are:

The name of the component. In this case, this will be “Ex1Comp”.

The naming convention here is to use the desired component name, followed by the suffix “Comp”. If this naming convention is followed, then you will make life easier for yourself when you come to read the generated Java coding.

Notice that as soon as you enter a component name (outlined at the top in red), the window and the component names are constructed automatically.

The Java package name. In this case com.sap.tutorial although you could use any package name you like here.

The Window name. By default, the name here will be the same as the component name. For purposes of clarity though, it is a good idea to replace the “Comp” suffix in the window name with the word “Window”. That way, you will not confuse the component with its window.

Renaming the window is not a necessity, but does prove to be helpful when doing more advanced coding.

Also, the View name defaults to the component name followed by the word “View”. Again, for reasons of clarity, it is often useful to remove the word “Comp” leaving just “Ex1View”.
As soon as you create a new component, various files are created by the NWDS. Some of these are XML files that contain general declarative information about the Web Dynpro component, and other files are Java source code files. Either way, the principle here is that as a result of declarations you make (I.E. the creation of a Web Dynpro component), the NWDS then generates the relevant source code for you.

This is a fundamental principle in Web Dynpro development.

You will notice that two diagram views have opened in the top right of your screen. If you just accepted the default name for your Window back in step 0, then the two tab strips at the top of the diagram view will both contain “Ex1Comp” because the Web Dynpro component and its window both share the same name. This is not illegal, but can be confusing if you’re new to Web Dynpro.
Since the exercise we are performing here is a very simple one, we can close the diagram view by clicking on the X in the top right corner of the Diagram View window.

Expand the “Web Dynpro Components” branch of the hierarchy, then expand the component name “Ex1Comp” and finally expand the “Views” branch.

Double click on the view name and you will see the View Editor open.
In the bottom left of the screen, you will see the Outline View. This is the view in which all the UI elements belonging to a particular view are represented as a hierarchy. The top level UI element in any view layout is always called “RootUIElementContainer” and it is always of type “TransparentContainer”. You will notice that there is already a TextView UI element underneath the root UI element that displays the name of the view.

Right mouse click on the RootUIElementContainer and select “Insert Child” from the pop-up menu.
Now scroll down the list of UI elements and select **TextView**.

Figure 10: Insert a new child UI element

Figure 11: Add a new TextView UI element

Now notice two things that have happened:
The UI element tree shows that a **TextView** UI element has been added:
The layout view does not look any different. So you might be thinking “Why can’t I see the UI element I’ve just added?”

Figure 12: The new UI element in the hierarchy

Figure 13: Where's the new UI element gone
The reason why you can't see the UI element you've just added is that before a TextView UI can become visible on the screen, you have to give it some text to display.¹

Make sure the new TextView UI element is selected in the Outline View, and then select the Properties tab from the tabstrip the bottom right of the screen.

¹ Doh!
Click on the “text” property field highlighted above and enter the text “Welcome to Web Dynpro” and press enter. Now the TextView UI element becomes visible.

Figure 15: The TextView is now visible – at the wrong location!
The only problem now is that the text “Welcome to Web Dynpro” appears on the same line as the text “Ex1View”. We would like to put this down on the line below. To do this, we need to change the way all the UI elements are arranged in this particular view.

Select the RootUIElementContainer in the Outline view and locate its “layout” property. By default, this will be set to “FlowLayout”. Select the value MatrixLayout from the drop down list. Now that you have changed layout manager for the entire view, all the UI elements in the hierarchy will now inherit a set of properties that relate to your choice of layout manager.

![Figure 16](image)

If you now select the TextView UI element you just added, you will see that there are a new group of extra properties. These have appeared because we changed the layout manager of the parent container UI element.

Select the new TextView UI element again and change its “LayoutData” property to MatrixHeadData. This does two things:

It forces the UI element to occupy the first column of a new row.

A new set of properties will appear that are specific to the MatrixHeadData setting. We do not need to make any changes to these new properties.
If we left our development at this point, we would have a perfectly functional Web Dynpro component – but with one vital thing missing! We have no way of invoking its functionality. This is where the last step comes in – creating an Application.

Right mouse click on the “Applications” branch under the “Web Dynpro” menu, and select “Create Application”. The purpose of an application is to provide a browser with an entry point into the functionality contained within your Web Dynpro Component.
In the pop-up window, enter the following:

The name of the application. The naming convention here is to add the suffix “App” to your desired application name. In this case, this is “Ex1App”.

Enter the Java package name. In this case com.sap.tutorial.
Press next.

In this case we have wrote the Web Dynpro component before the application was created; therefore, choose the “Use existing component” option and press Next.

In this particular case, we only have one Web Dynpro component to choose from, so none of the options on this screen can be adjusted. So just press finish.

It is perfectly possible however, to build a business process using multiple Web Dynpro components. In this case you would need to nominate one of those components to become the starting point for your functionality.
Building, Deploying, and Running

We are now ready to build the entire Web Dynpro application and deploy it to the SAP Java Server. This is done (most easily) by expanding the Applications branch of the hierarchy to show the application we have just created, and then right mouse clicking on it to display a pop-up menu.
Figure 22: Select “Deploy New Archive and Run” from the pop-up menu

This option will perform three tasks one after the other:
- It will build all the Web Dynpro Components that have changed since the last build.
- It will create an Enterprise Archive (.ear) file
- It will deploy the .ear file to the SAP Java Server.

When you deploy an application for the first time since starting the NetWeaver Design Studio, you will be asked for the Software Deployment Manager password. The default value is sdm.

Your browser should now show the following display:

Figure 23: The final result