SDN Community Contribution

(This is not an official SAP document.)

WDA Tutorial I: Getting Started with Web Dynpro for ABAP

Applies To:

SAP Web AS 7.00

Summary

This tutorial provides a step-by-step guide for developing your first Web Dynpro for ABAP application.

By: Rich Heilman

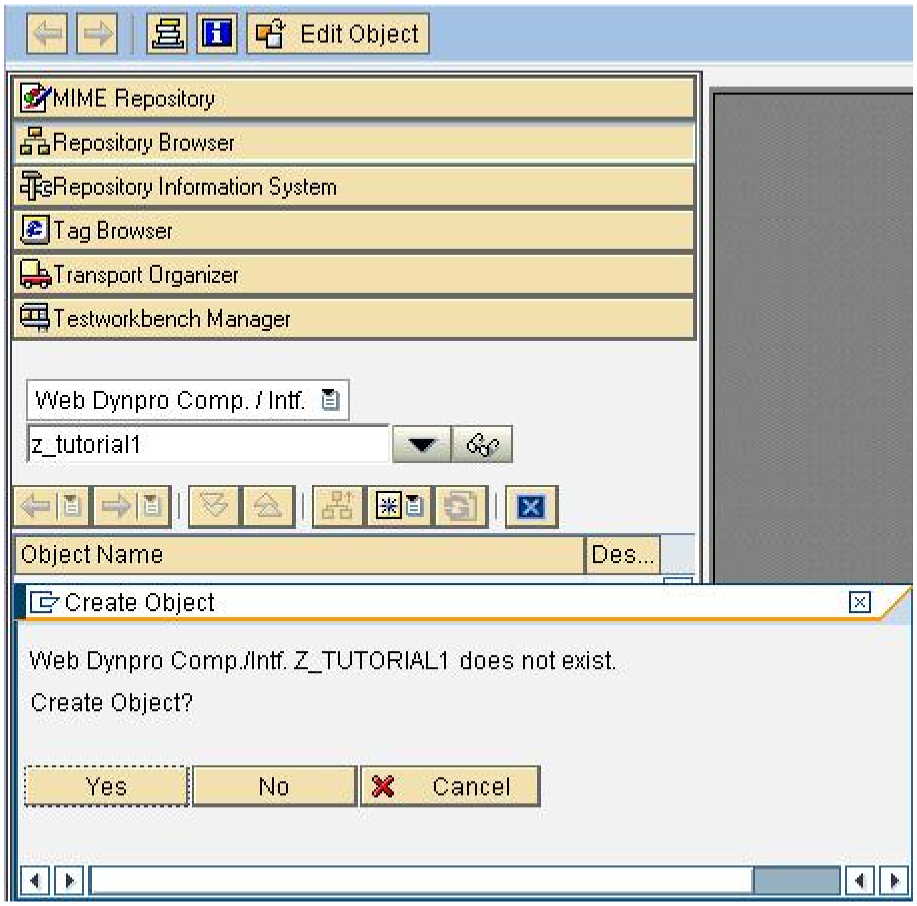
Company: Yorktowne Cabinetry, Inc.

Date: 12/01/2005

Table of Contents

Step 1 – Creating the Web Dynpro for ABAP (WDA) Object

Go to transaction code SE80. This is the ABAP Workbench. In the object list box, choose *Web Dynpro Comp / Inf*. Enter the name of the object as Z_TUTORIAL1 and hit Enter. The system will ask you if you want to create this object. Click Yes.



Web Dynpro Comp. / Intf.

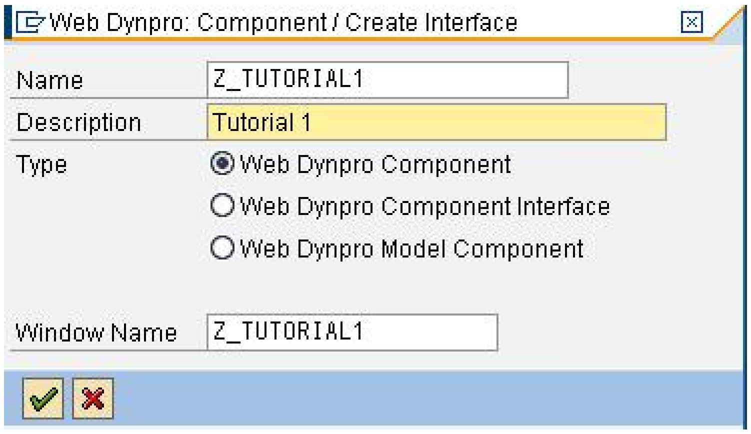
z_tutorial1

Create Object

Web Dynpro Comp./Intf. Z_TUTORIAL1 does not exist.

Create Object?

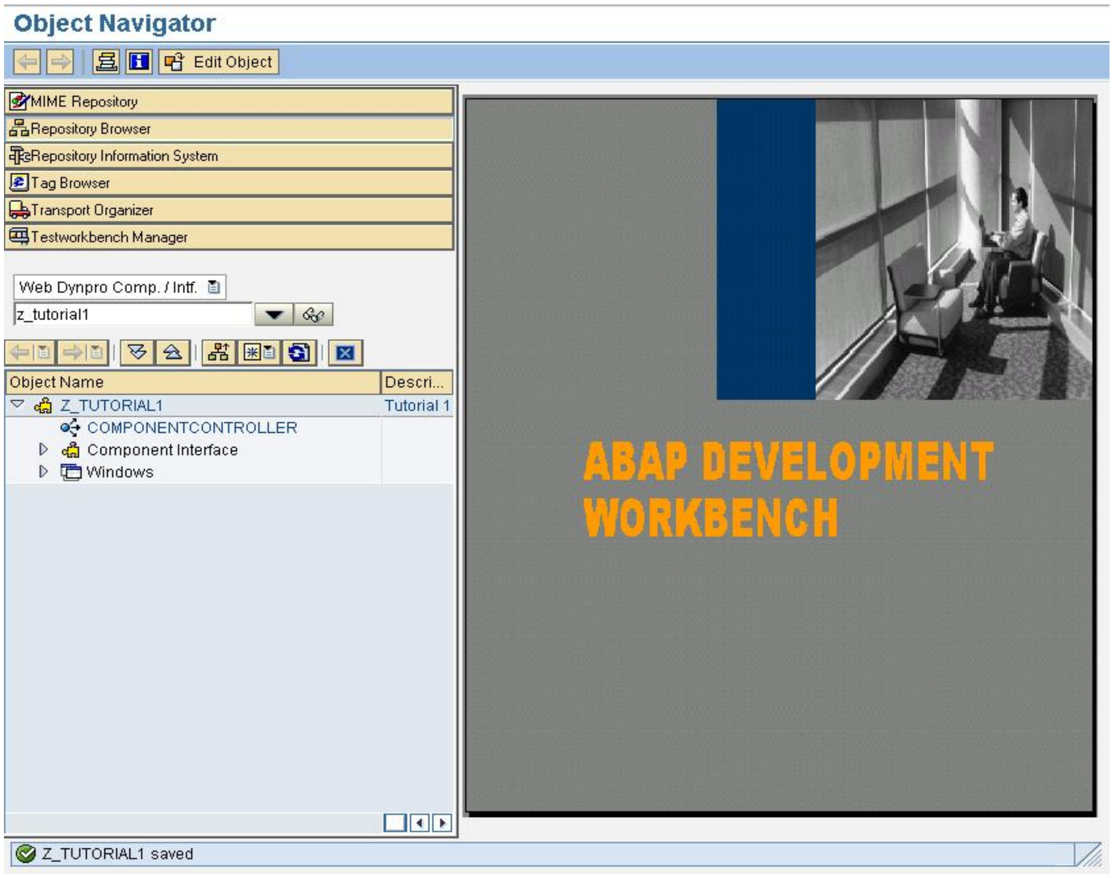
Enter a description for the object and hit Enter. A dialog will appear asking to assign a package. Click *Local Object*.



Web Dynpro: Component / Create Interface

Name	Z_TUTORIAL1
Description	Tutorial 1
Type	● Web Dynpro Component
	○ Web Dynpro Component Interface
	○ Web Dynpro Model Component
Window Name	Z_TUTORIAL1

Now the WDA Object has been created.

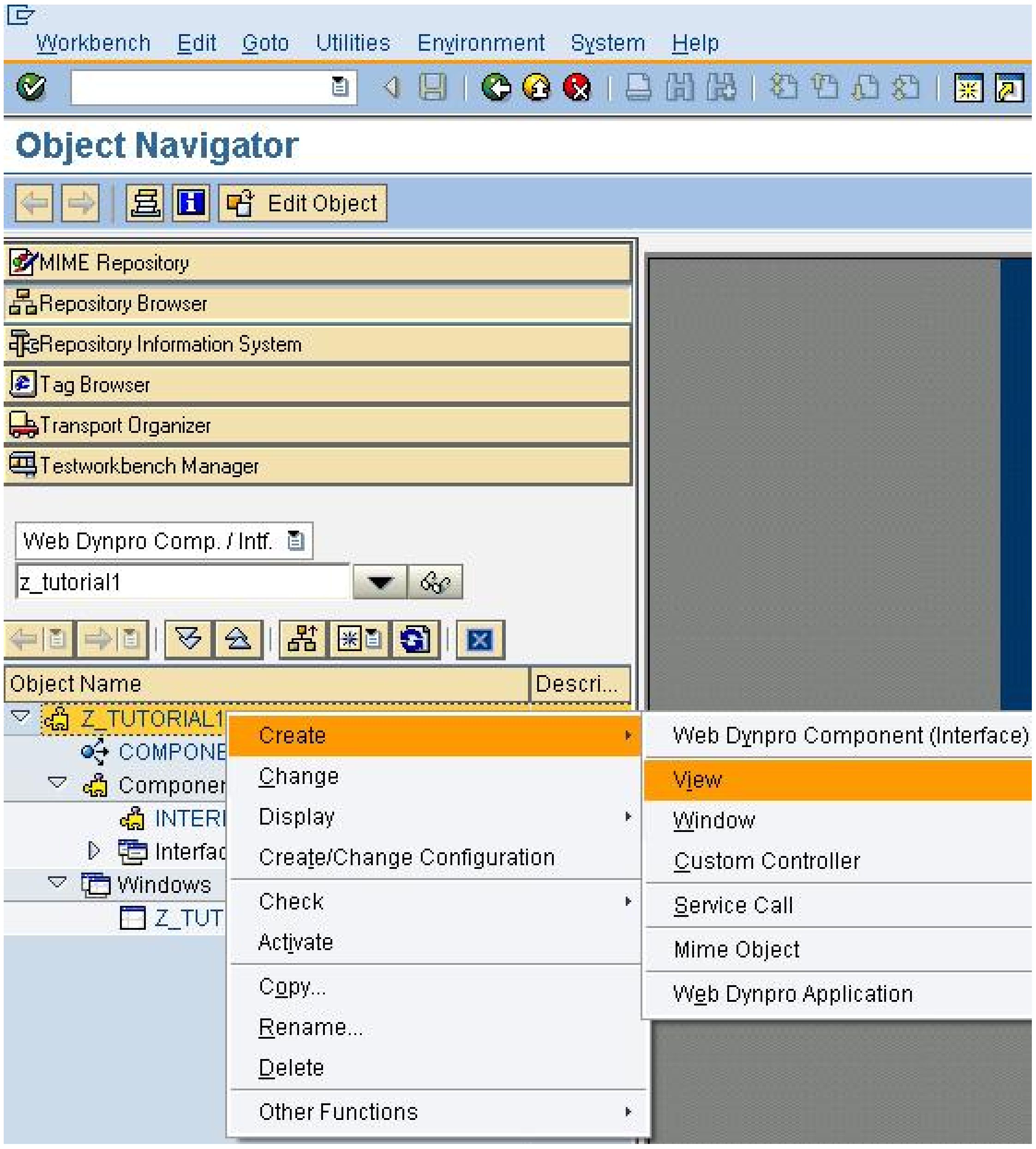


Z_TUTORIAL1 saved

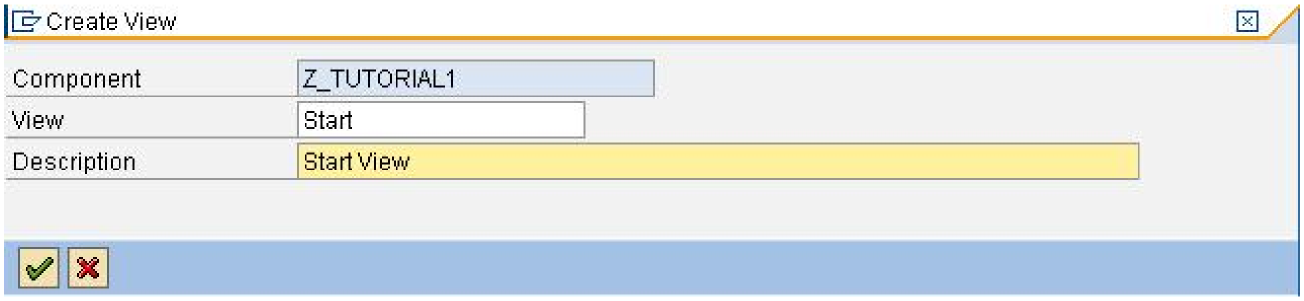
Step 2 – Creating a View

Views contain the UI elements needed for the user to interact with the application. Many different kinds of UI elements are delivered by SAP.

Right click on the WDA Object, choose *Create, View*.

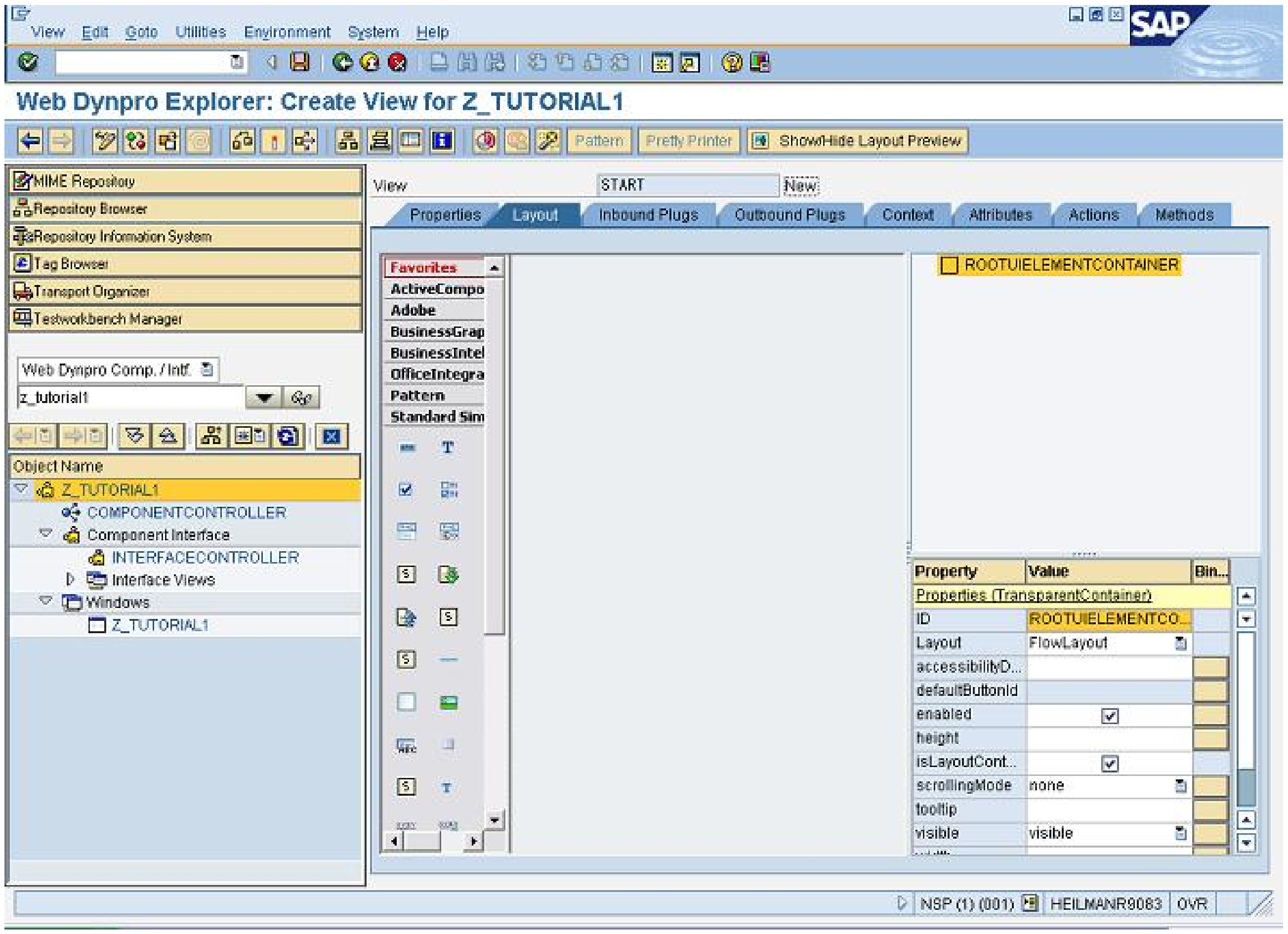


A dialog will appear, enter START as the view name, and “Start View” as the description. Hit Enter.



Component	Z_TUTORIAL1
View	Start
Description	Start View

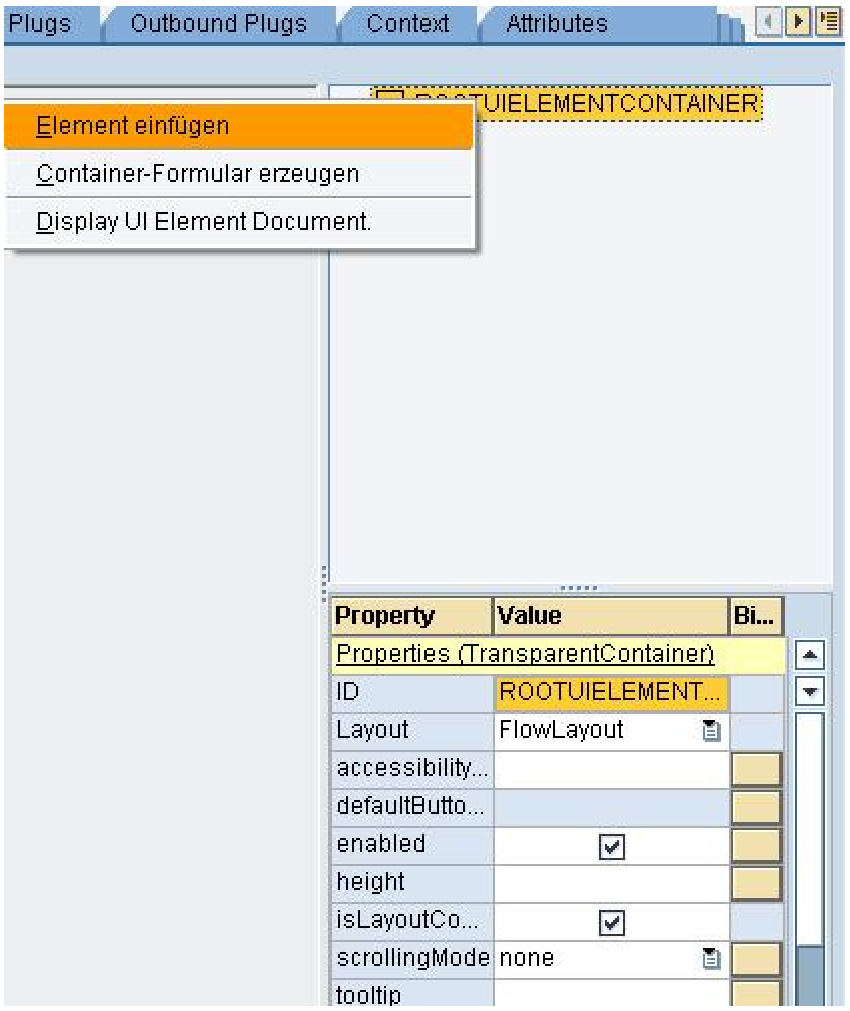
Now the view has been created.



Property	Value	Bin...
Properties (TransparentContainer)		
ID	ROOTUIELEMENTCO...	
Layout	FlowLayout	
accessibilityD...		
defaultButtonId		
enabled	✓	
height		
isLayoutCont...	✓	
scrollingMode	none	
tooltip		
visible	visible	

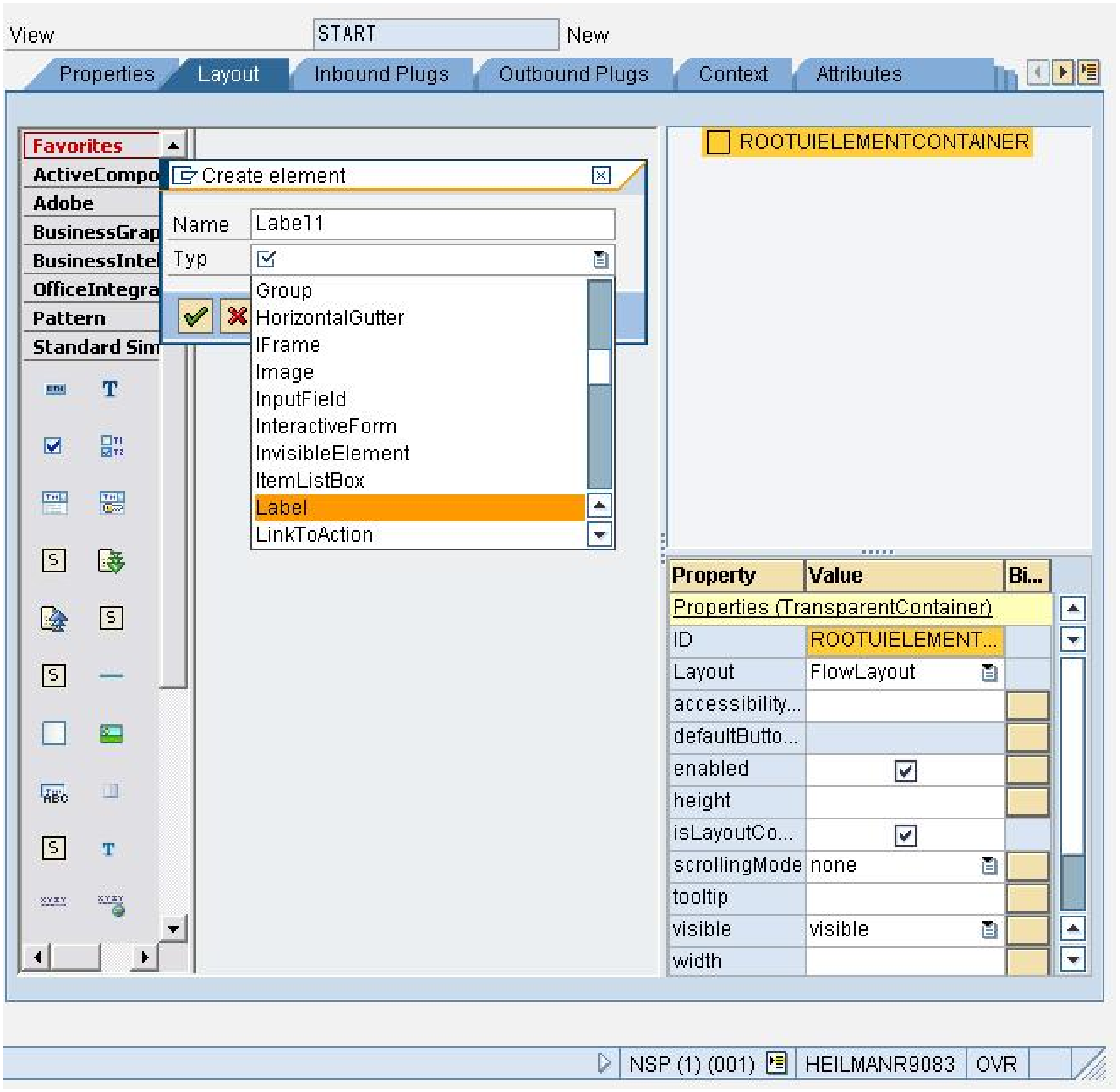
Step 3 – Adding UI Elements to the View

Next, build the user interface for this view. Put a label and an input field on the view. Drag and drop the UI elements from the icons to the view layout. This can also be done by right clicking on the ROOTUIELEMENTCONTAINER and choose *Create Element*.



Property	Value	Bi...
Properties (TransparentContainer)		
ID	ROOTUIELEMENT...	
Layout	FlowLayout	
accessibility...		
defaultButto...		
enabled	✓	
height		
isLayoutCo...	✓	
scrollingMode	none	
tooltip		

A dialog will appear. Enter *Label1* as the name of the element and choose *Label* as the type.



View START New

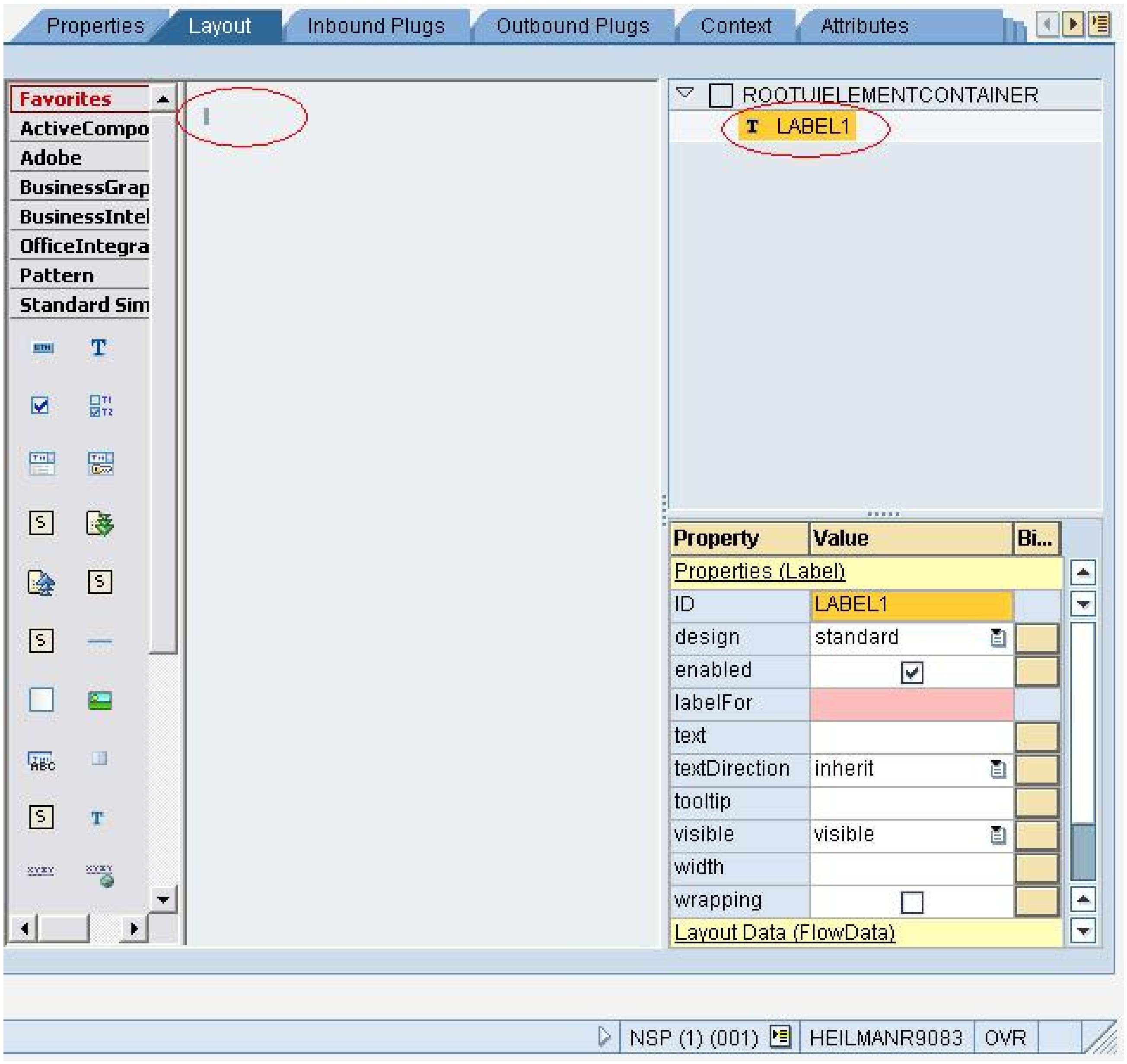
Create element

Name	Label1
Typ	✓

Group
HorizontalGutter
IFrame
Image
InputField
InteractiveForm
InvisibleElement
ItemListBox
Label
LinkToAction

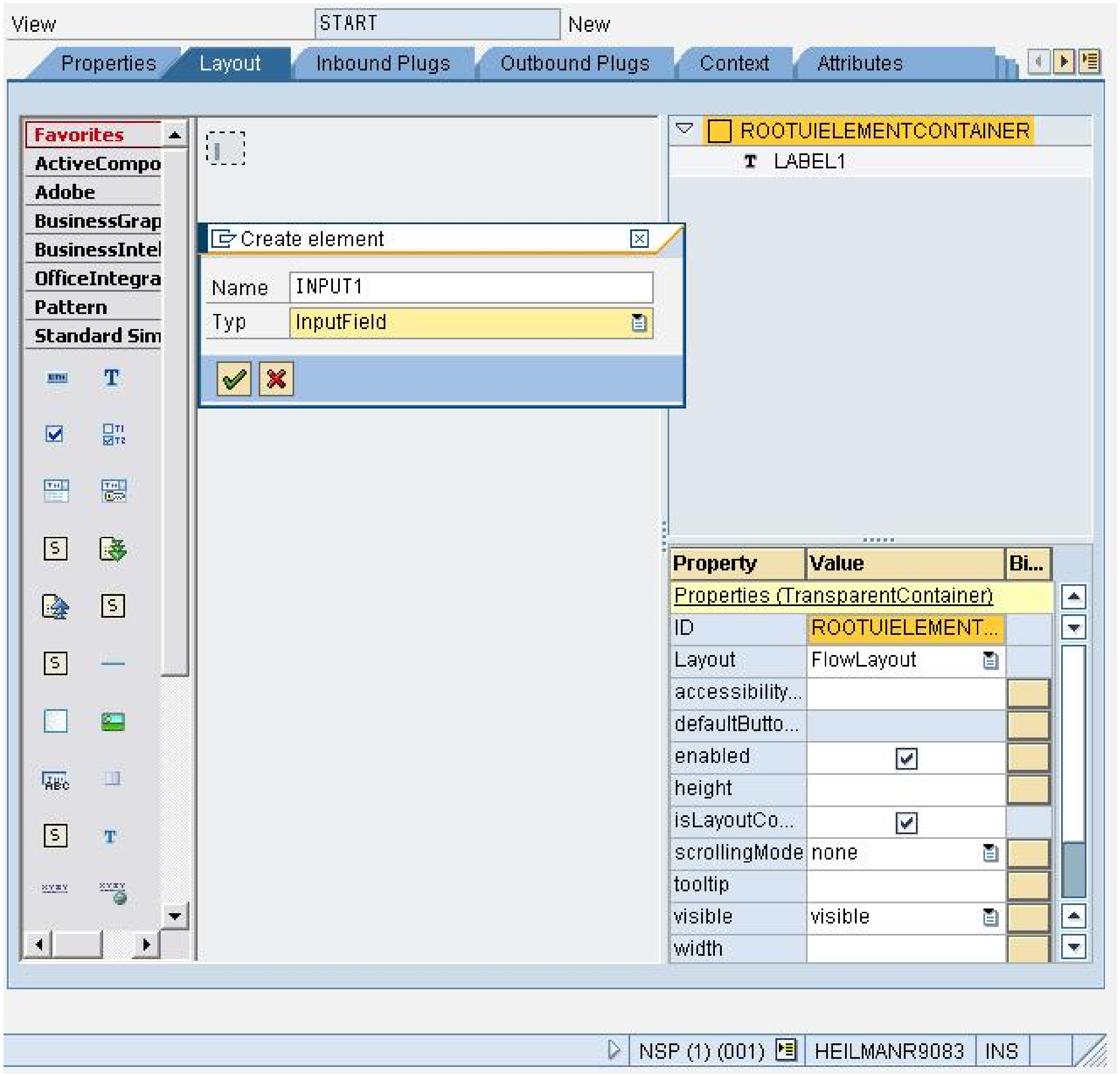
Property	Value	Bi...
Properties (TransparentContainer)		
ID	ROOTUIELEMENT...	
Layout	FlowLayout	
accessibility...		
defaultButto...		
enabled	✓	
height		
isLayoutCo...	✓	
scrollingMode	none	
tooltip		
visible	visible	
width		

Now the label has been created in the view. Notice the layout as well as the element under the ROOTUIELEMENTCONTAINER tree.



Property	Value	Bi...
Properties (Label)		
ID	LABEL1	
design	standard	
enabled	☑	
labelFor		
text		
textDirection	inherit	
tooltip		
visible	visible	
width		
wrapping	☐	
Layout Data (FlowData)		

Create another UI element, this time create an Input field. Right-click on the ROOTUIELEMENTCONTAINER, choose *Create Element*, enter the name of the element as INPUT1 and select the type *Input Field* from the list box.

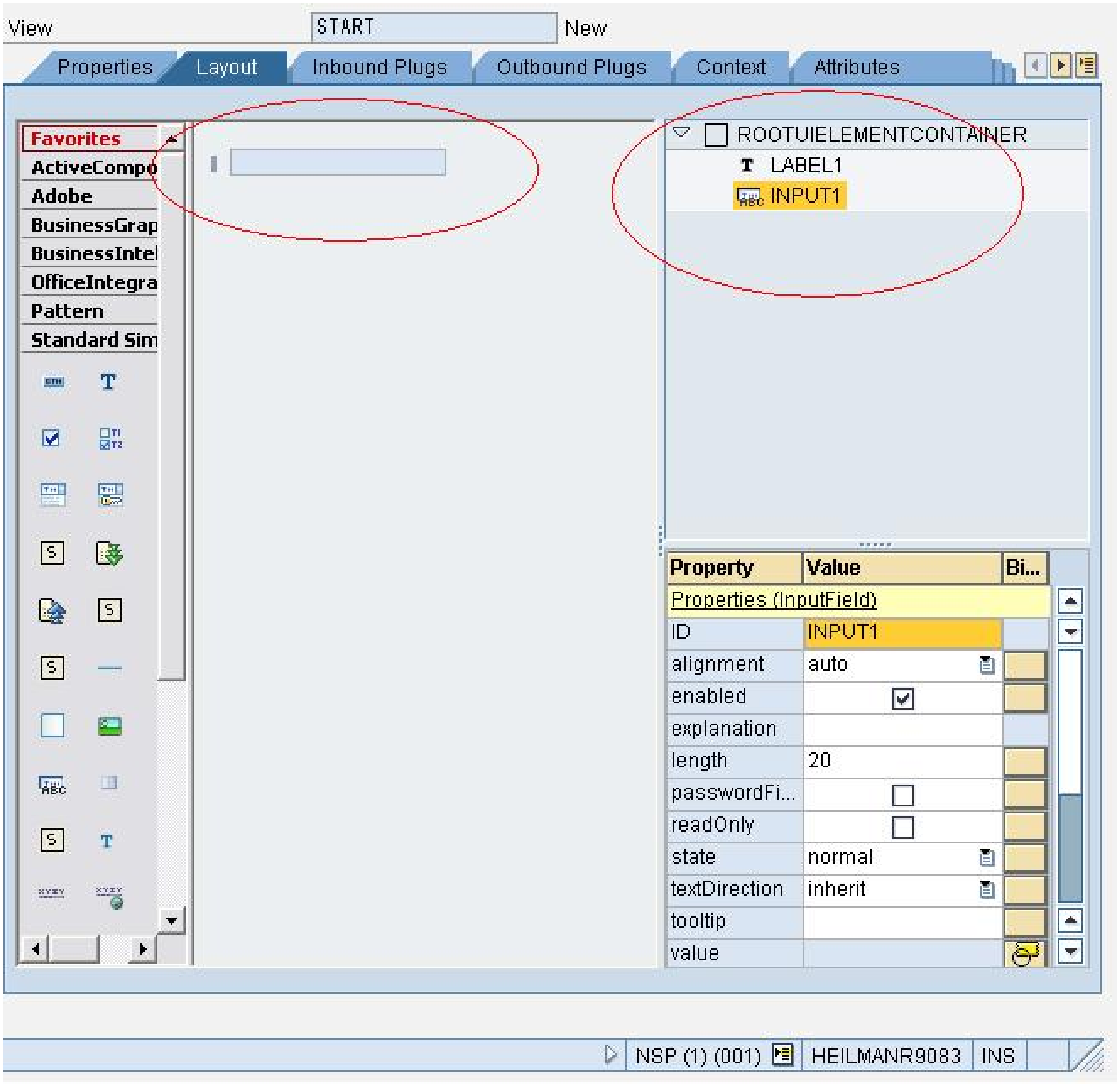


Create element

Name	INPUT1
Typ	InputField

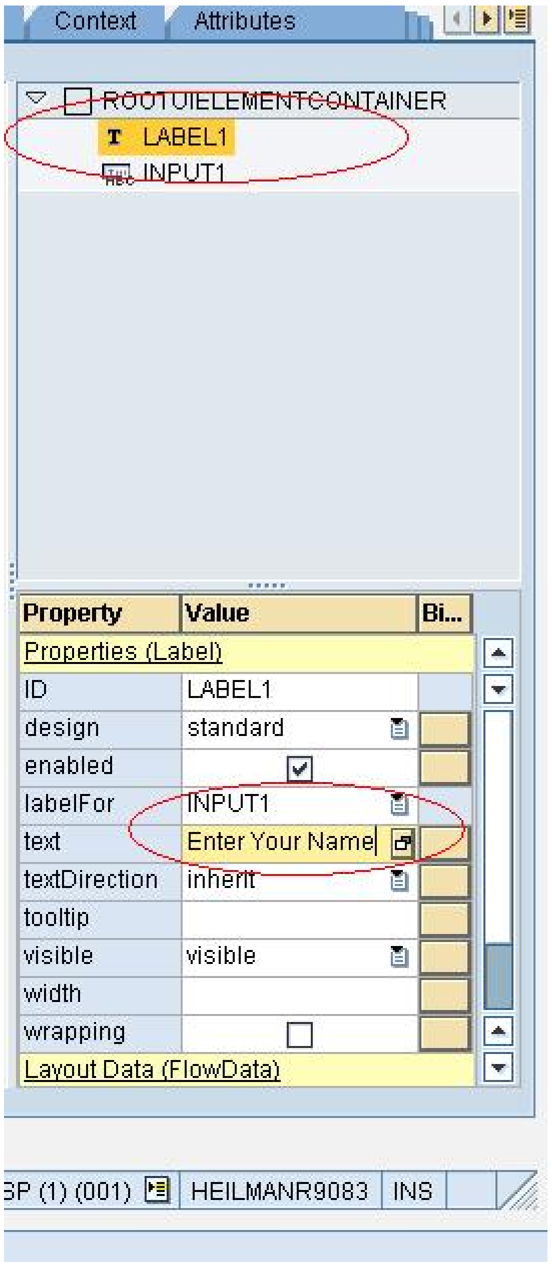
Property	Value	Bi...
Properties (TransparentContainer)		
ID	ROOTUIELEMENT...	
Layout	FlowLayout	
accessibility...		
defaultButto...		
enabled	✓	
height		
isLayoutCo...	✓	
scrollingMode	none	
tooltip		
visible	visible	
width		

Notice the view layout. There is a label as well as an input field.



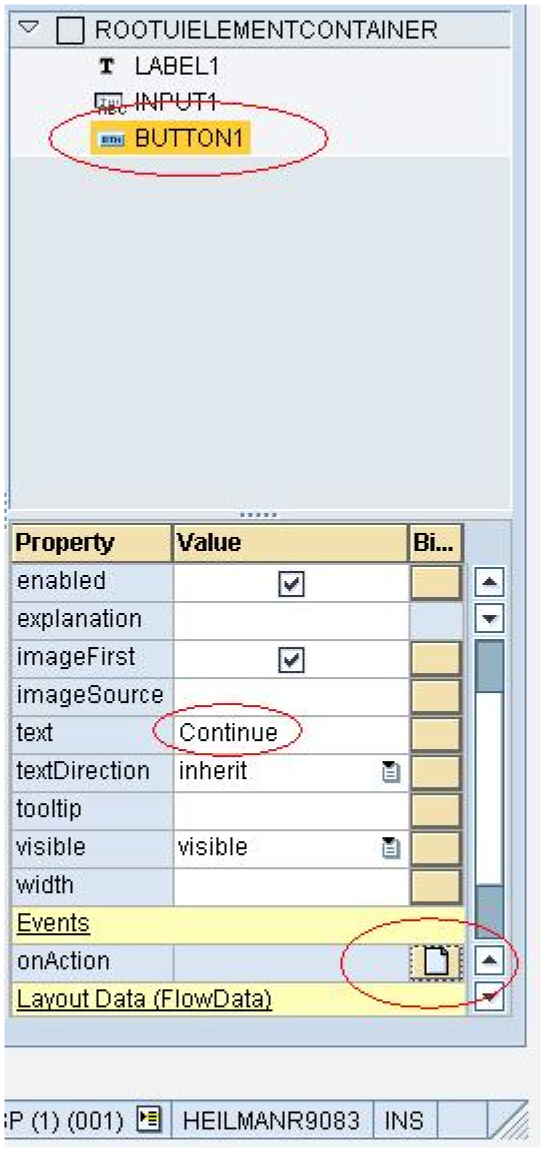
Property	Value	Bi...
Properties (InputField)		
ID	INPUT1	
alignment	auto	
enabled	✓	
explanation		
length	20	
passwordFi...	☐	
readOnly	☐	
state	normal	
textDirection	inherit	
tooltip		
value		

Assign some text for the label. Double-click on the LABEL1 element in the ROOTUIELEMENTCONTAINER tree. The properties of the element will show up below in the Properties box. In the *Label For* property, use the drop down to select the field that the label should be tied to. In this case, choose INPUT1. Also, enter the text for the label as “Enter Your Name”. Hit Enter. The text will now show in the view layout.



Property	Value	Bi...
Properties (Label)		
ID	LABEL1	
design	standard	
enabled	✓	
labelFor	INPUT1	
text	Enter Your Name	
textDirection	inherit	
tooltip		
visible	visible	
width		
wrapping	☐	
Layout Data (FlowData)		

Add a button to the view the same way that you created the other UI elements. Enter BUTTON1 as the name and choose *Button* from the list box and hit enter. Now double click on the BUTTON1 under the ROOTUIELEMENTCONTAINER. Enter the text as “Continue”, and then click the *Create* icon next to *onAction* property under the *Events* section.



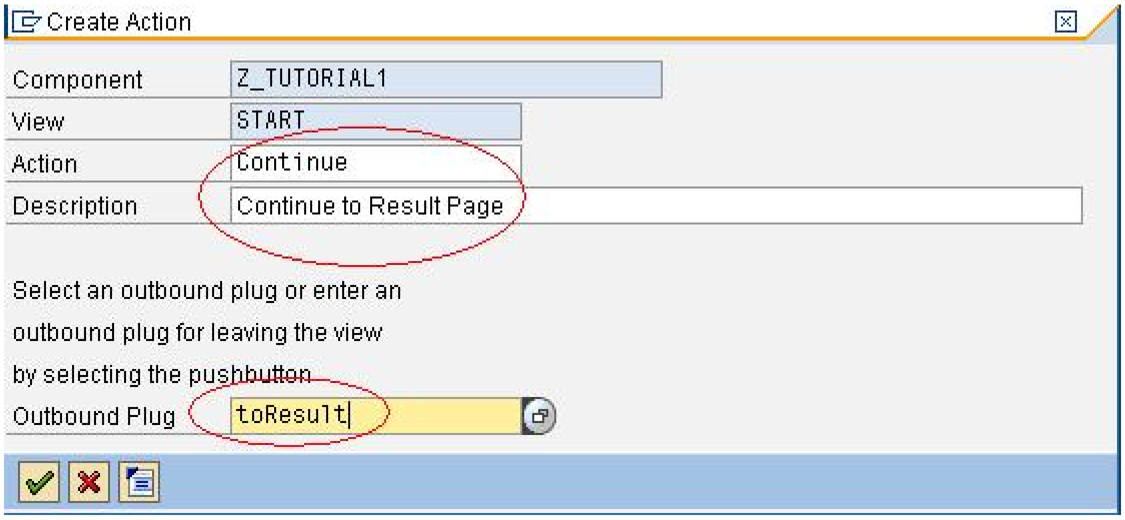
ROOTUIELEMENTCONTAINER
- LABEL1
- INPUT1
- BUTTON1

Property	Value	Bi...
enabled	✓	
explanation		
imageFirst	✓	
imageSource		
text	Continue	
textDirection	inherit	
tooltip		
visible	visible	
width		
Events		
onAction		
Layout Data (FlowData)		

P (1) (001) | HEILMANR9083 | INS

An action must be defined to navigate from one view to the next. When you define this action, an event handler method is generated. The method name is onAction<Action Name>. This event handler method will fire the outbound plug.

Now create the *Action* that will be tied to this button. Give the values shown below and hit enter.



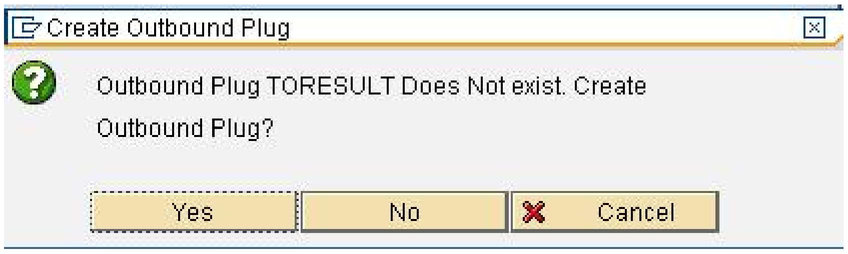
Create Action

Component	Z_TUTORIAL1
View	START
Action	Continue
Description	Continue to Result Page

Select an outbound plug or enter an outbound plug for leaving the view by selecting the pushbutton

Outbound Plug	toResult

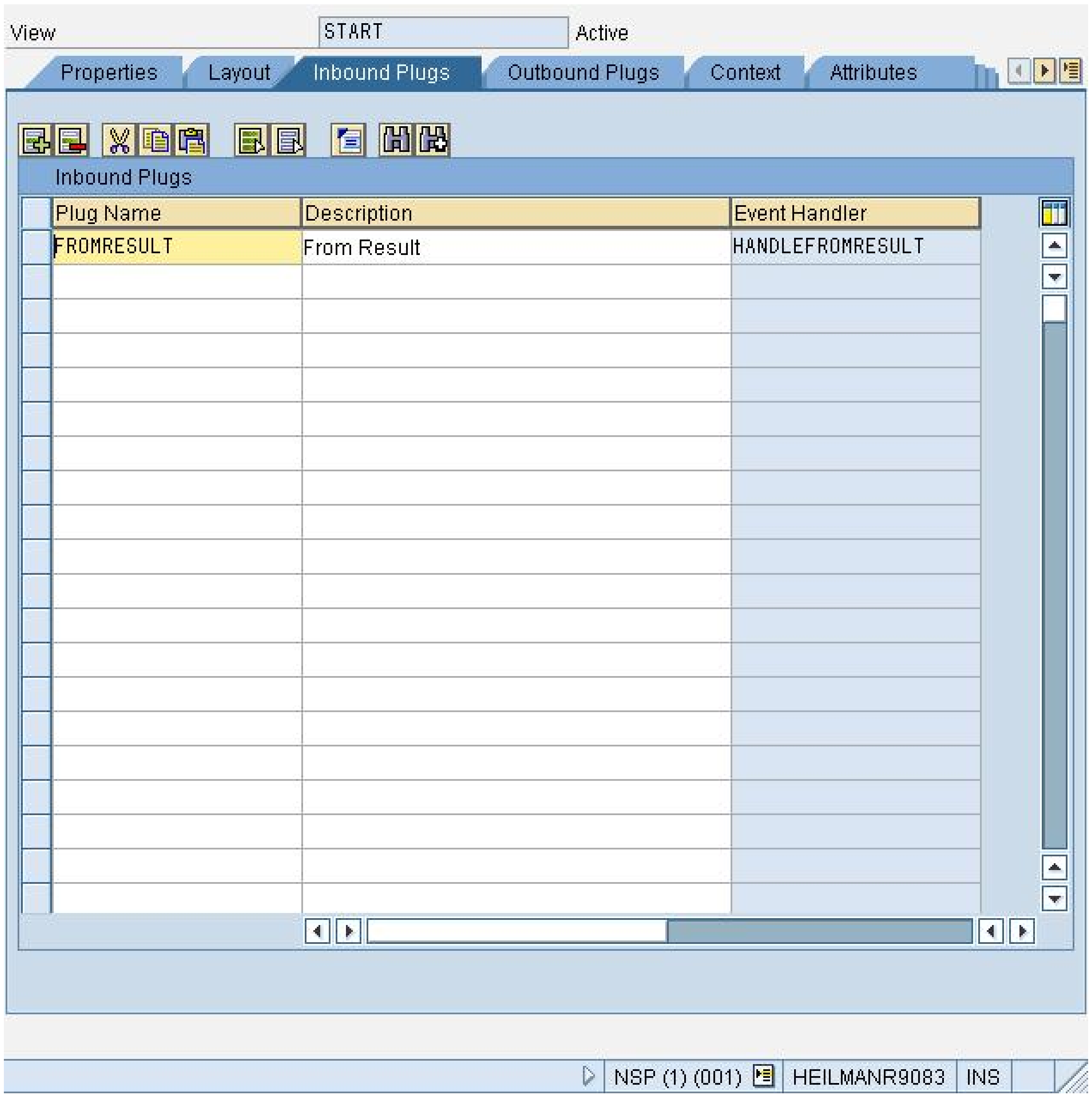
The system will ask about creating the Outbound Plug, click yes.



Create Outbound Plug

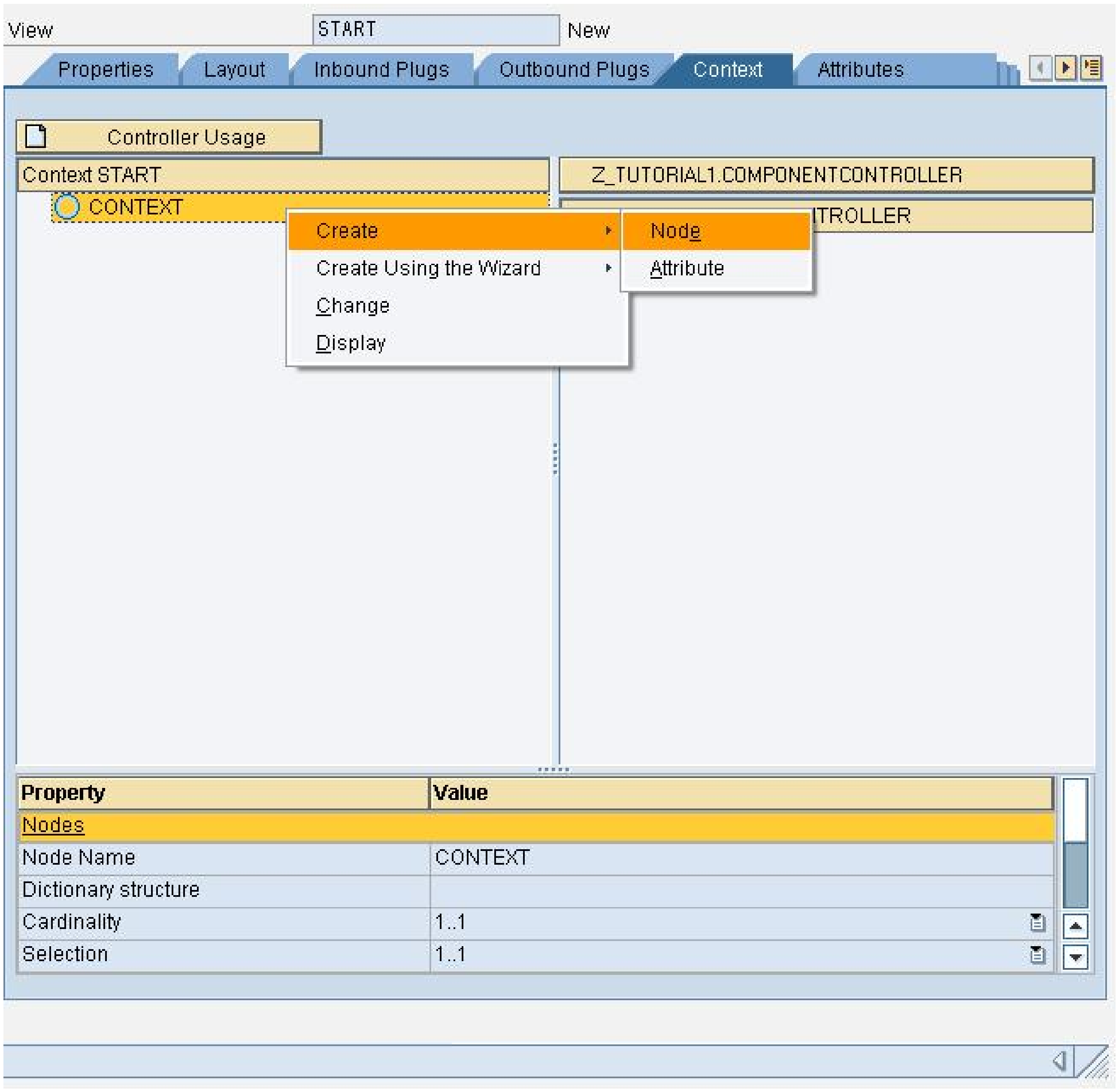
Outbound Plug TORESULT Does Not exist. Create Outbound Plug?

Click on the Inbound Plugs tab and create the Inbound plug FROMRESULT.



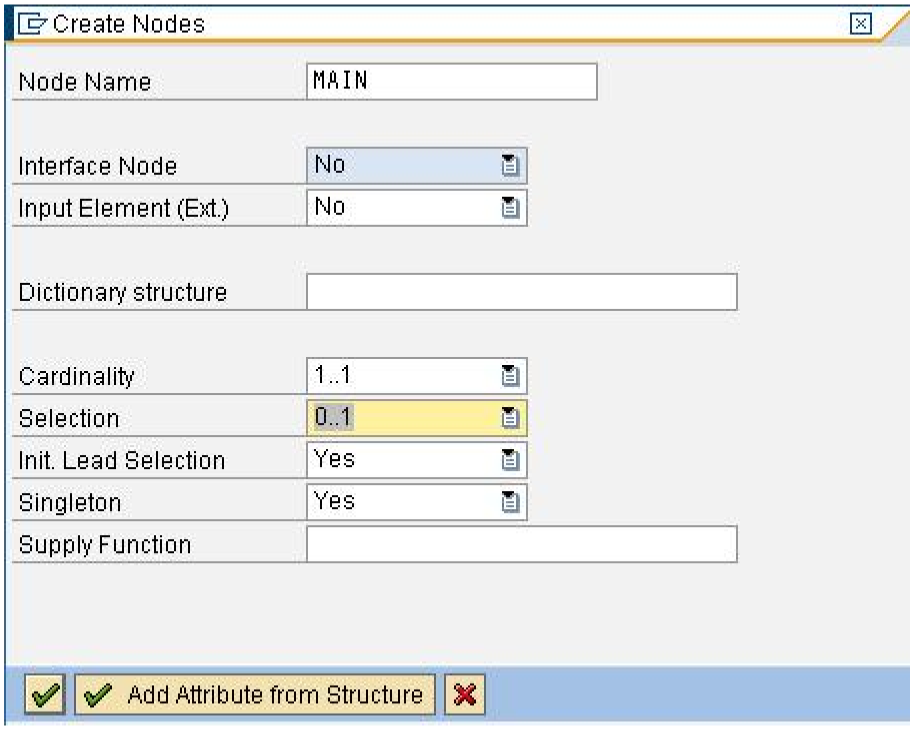
Plug Name	Description	Event Handler
FROMRESULT	From Result	HANDLEFROMRESULT

Create the context of the view. Click on the *Context* tab. First, create the main node. Right-click on the *Context*, choose *Create*, then *Node*.



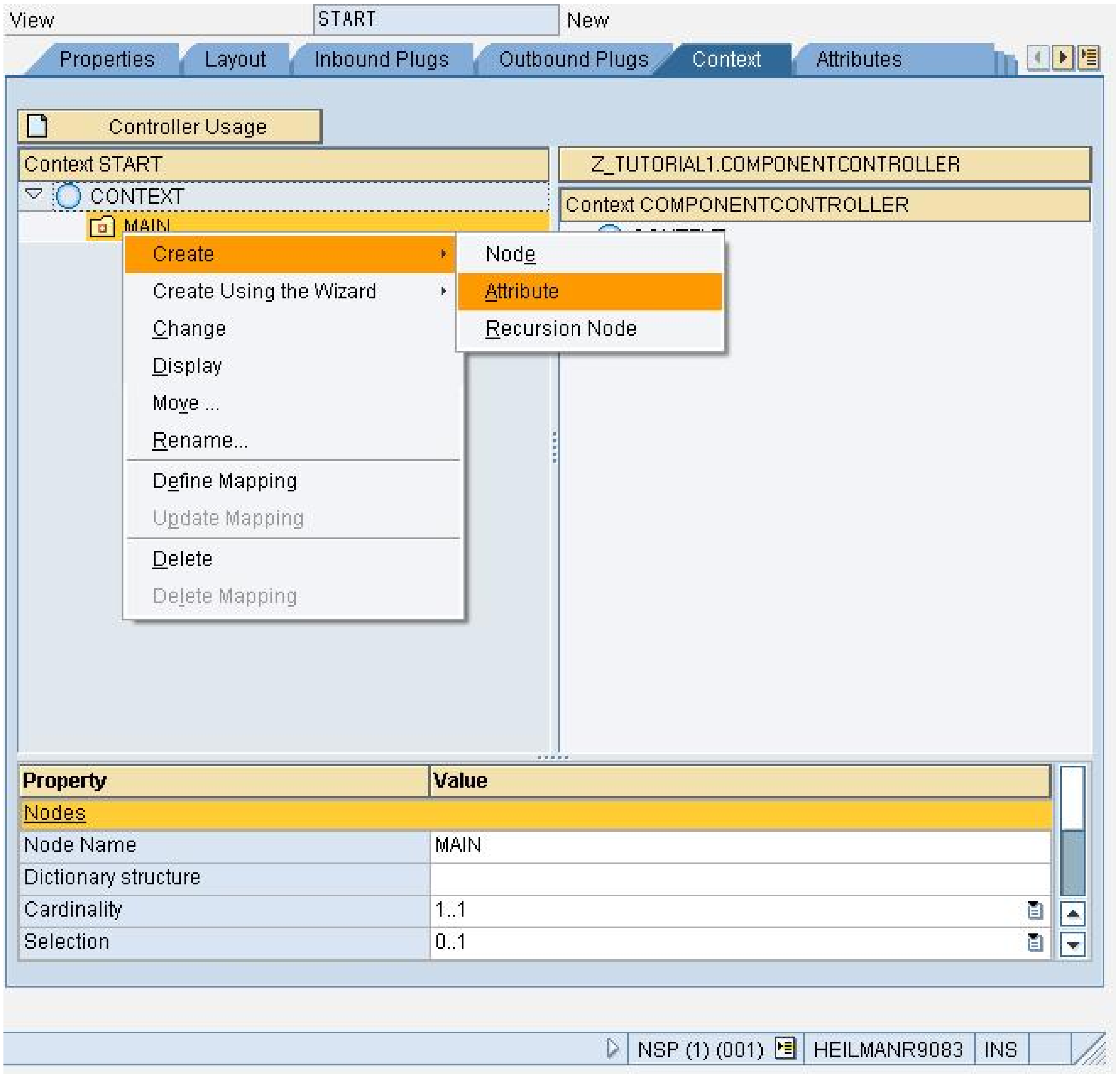
Property	Value
Nodes	
Node Name	CONTEXT
Dictionary structure	
Cardinality	1..1
Selection	1..1

Enter the values as they appear below for Node MAIN.



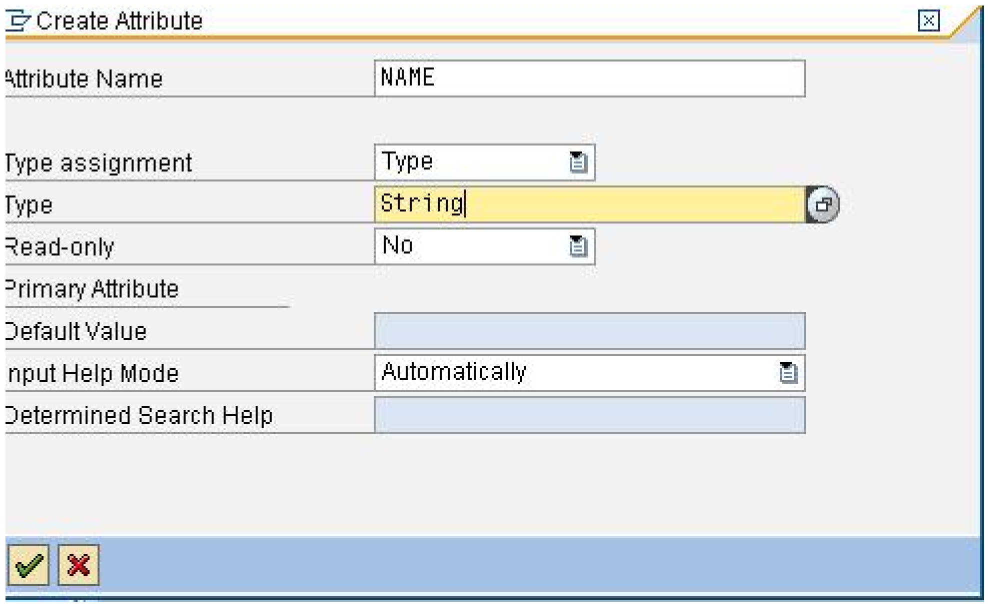
Node Name	MAIN
Interface Node	No
Input Element (Ext.)	No
Dictionary structure	
Cardinality	1..1
Selection	0..1
Init. Lead Selection	Yes
Singleton	Yes
Supply Function	

Next, create an attribute under the Node MAIN. Right-click on the node MAIN and choose *Create*, then *Attribute*.



Property	Value
Nodes	
Node Name	MAIN
Dictionary structure	
Cardinality	1..1
Selection	0..1

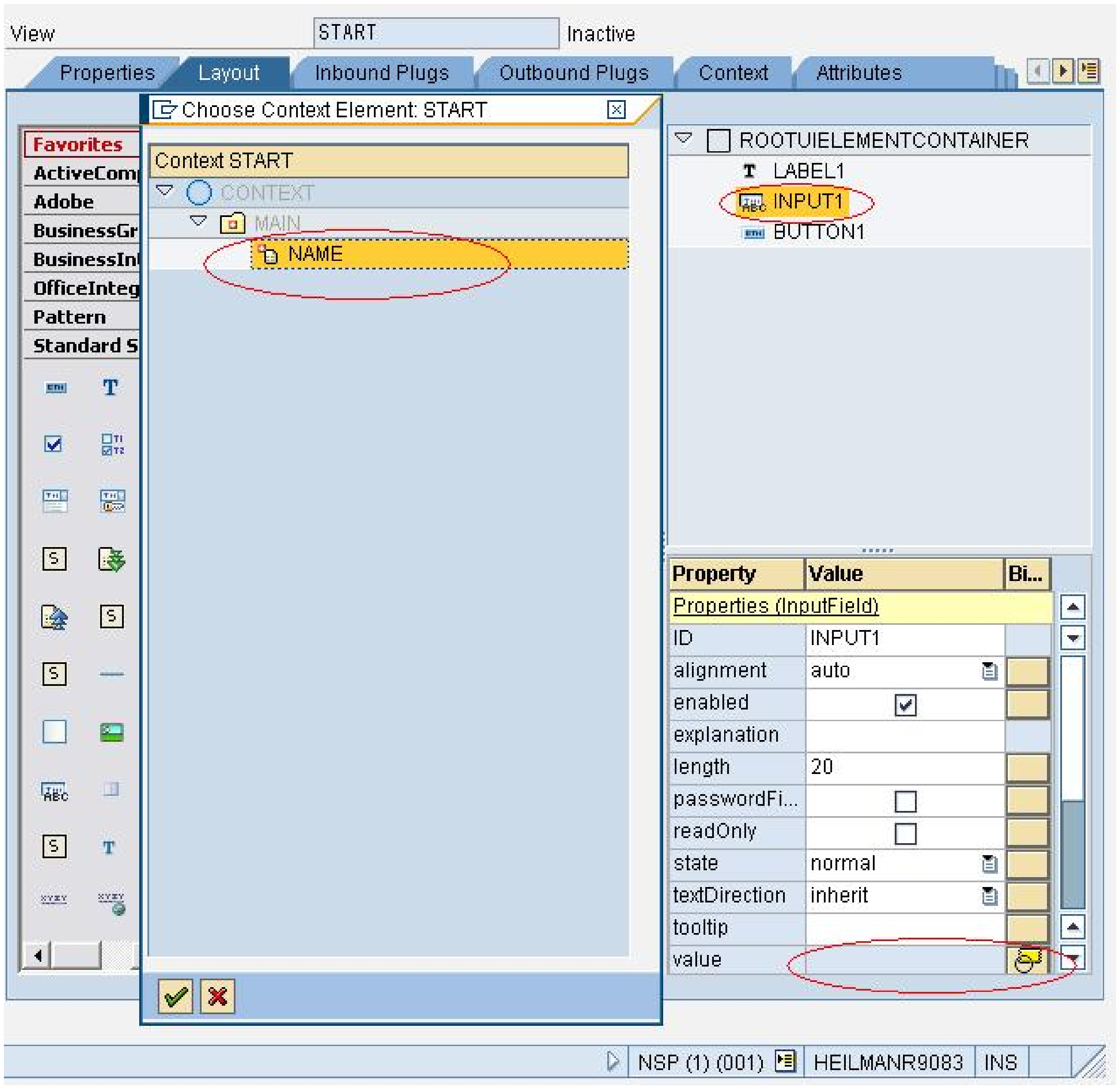
Next, enter the values show below.



Create Attribute

Attribute Name	NAME
Type assignment	Type
Type	String
Read-only	No
Primary Attribute	
Default Value	
Input Help Mode	Automatically
Determined Search Help	

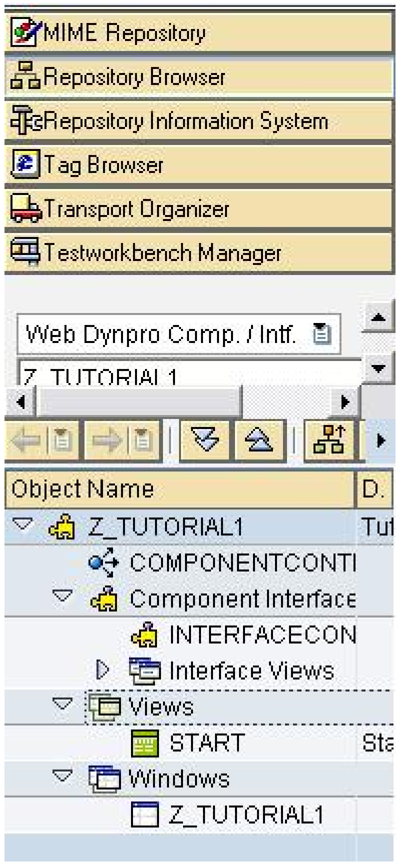
Next, click on the Layout tab of the view, and then double-click the INPUT1 UI element. In the properties box, map the UI element to the view context. Click the icon next to the value property, and choose NAME from the dialog.



Property	Value	Bi...
Properties (InputField)		
ID	INPUT1	
alignment	auto	
enabled	✓	
explanation		
length	20	
passwordFi...	☐	
readOnly	☐	
state	normal	
textDirection	inherit	
tooltip		
value		

Save and activate all objects for this WDA project.

Notice that the View folder has been created in the WDA object tree.

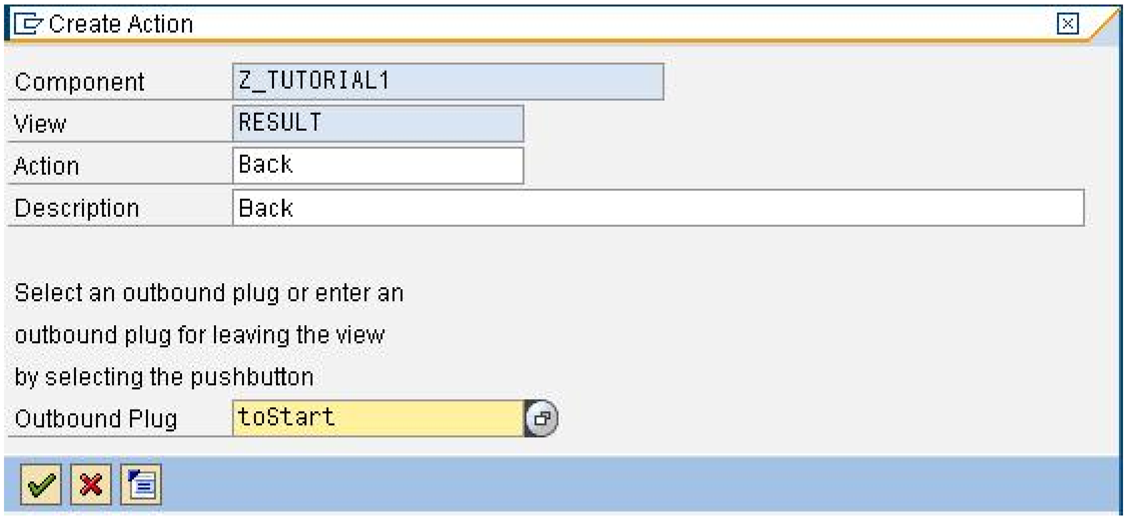


Create the second view called RESULT the same way as the START view. In the result view, create a UI element called TEXT1 with type *TextView*. Also add a Button called BUTTON1, and assign the text as "Back".

Click on the *Context* tab and create the view context. First create the node, call it "MAIN", then create the node attribute under the MAIN node and call it NAME. The NAME attribute should be of type String.

Next, map the TEXT1 UI element to the view context. Click on the Layout Tab. Double-click on the TEXT1 UI element. Click on the box, next to the "text" property. A dialog will appear, choose the NAME attribute from the view context. Hit Enter.

Double-click on the BUTTON1 UI element, in the properties box. Click the *Create* button next to the *onAction* property.   A dialog will appear, enter the values as seen here.

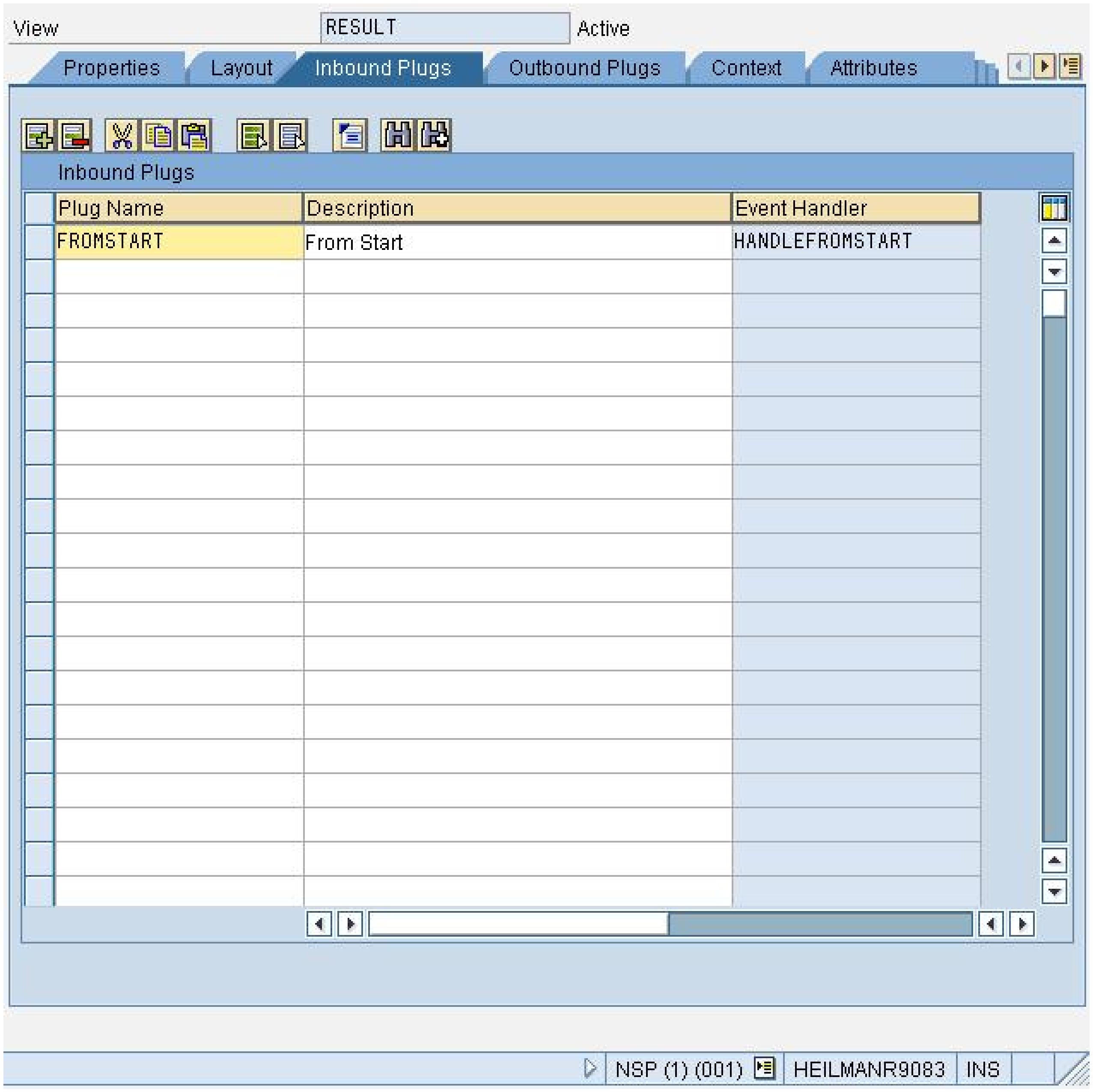


Component	Z_TUTORIAL1
View	RESULT
Action	Back
Description	Back

Select an outbound plug or enter an outbound plug for leaving the view by selecting the pushbutton

Outbound Plug	toStart

Click on the Inbound Plugs tab, and create the FROMSTART inbound plug.

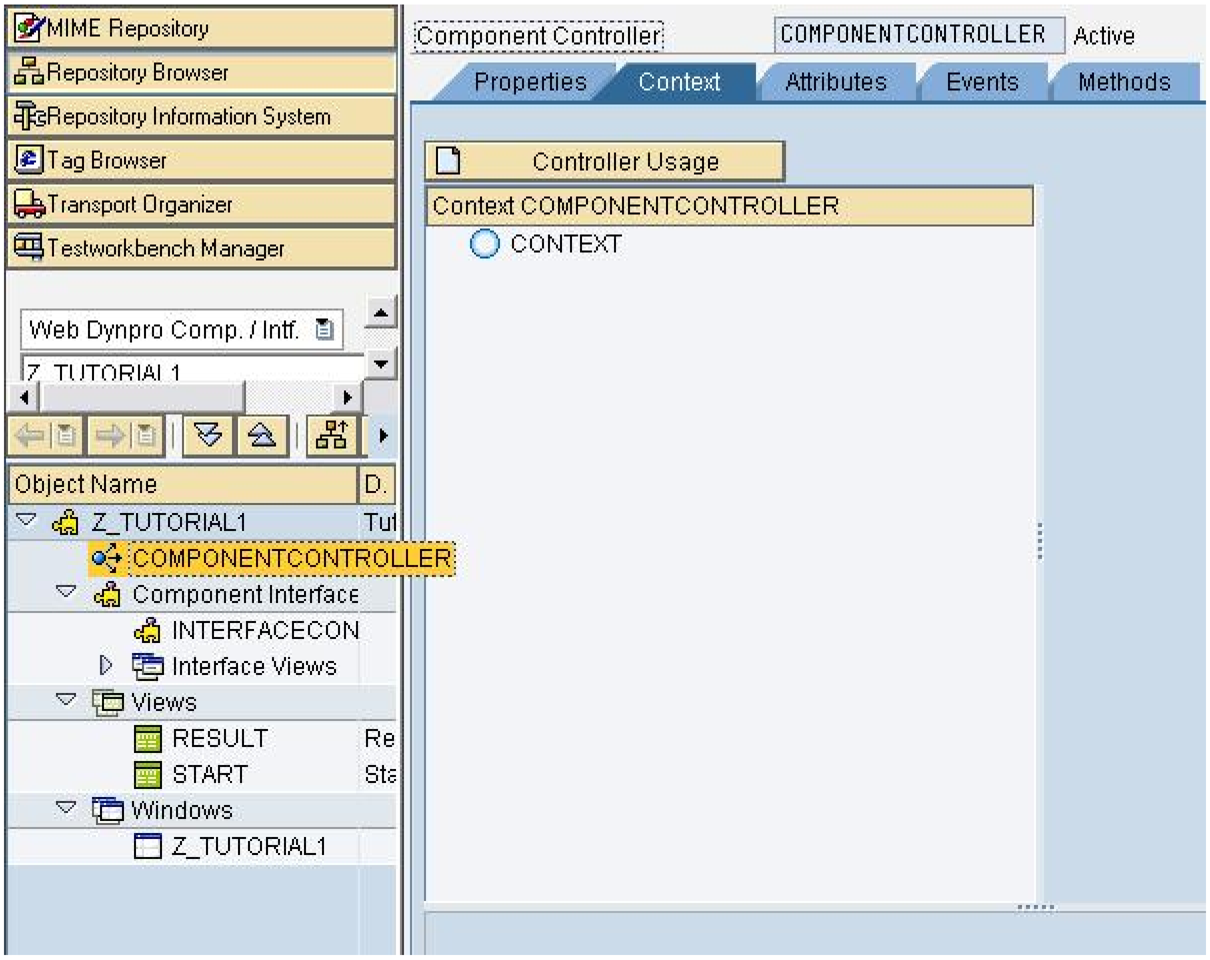


Plug Name	Description	Event Handler
FROMSTART	From Start	HANDLEFROMSTART

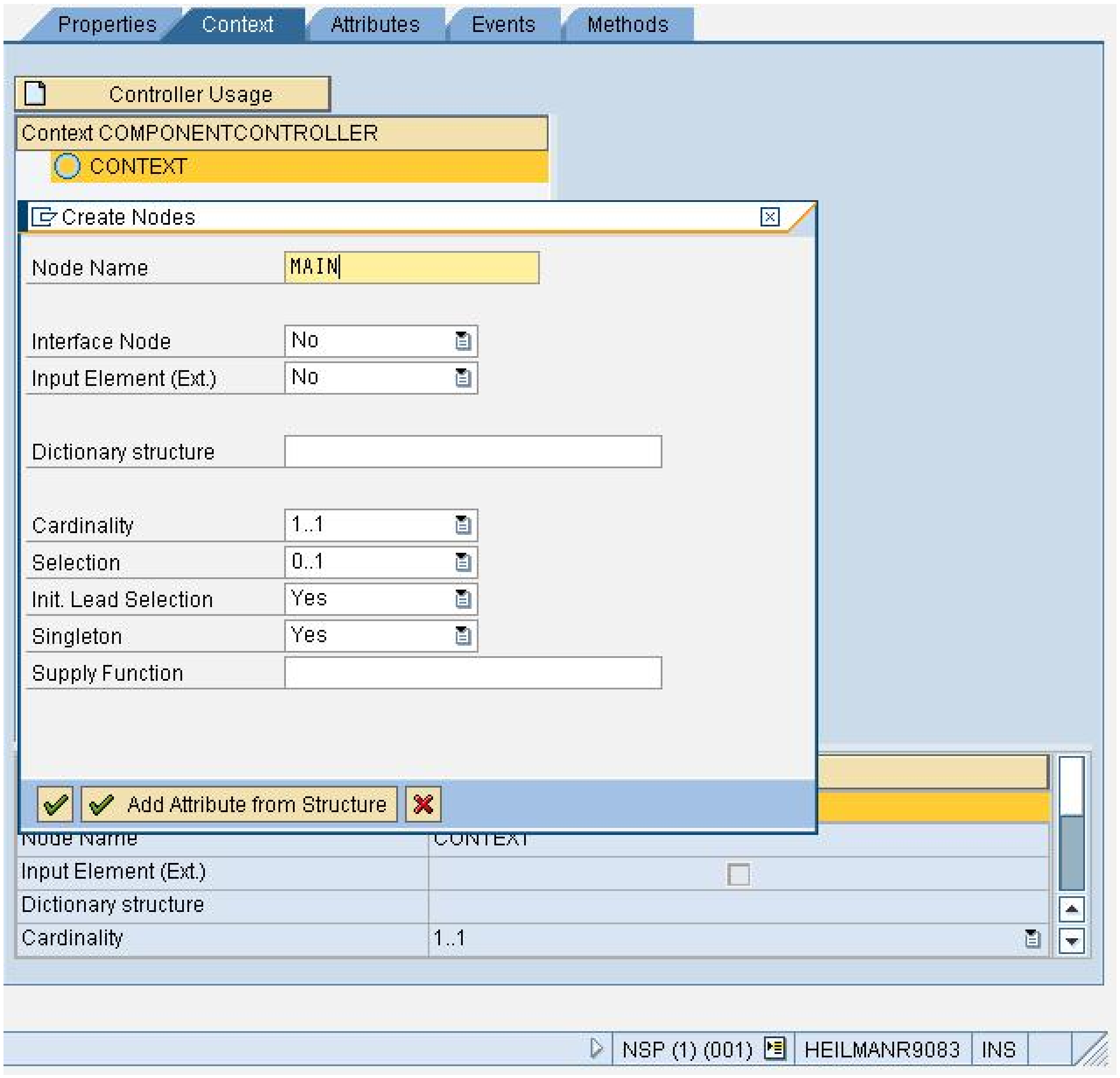
Save and activate your work.

Step 4 – Creating the Component Controller Context

Create the component controller context. This will control the data being passed to and from the views. Double-click on the Component Controller from the object tree on the left.



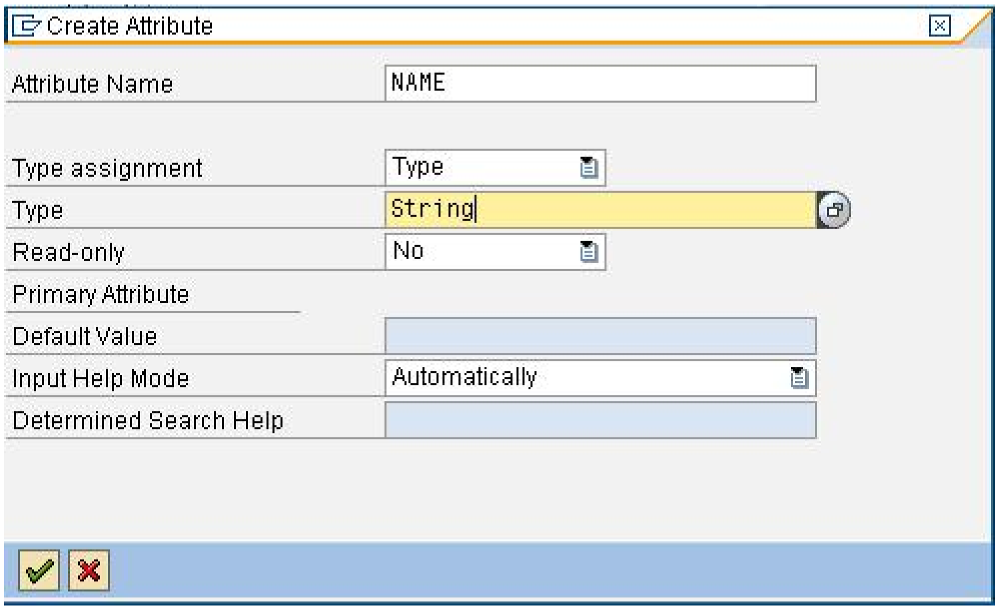
Create the context by right clicking on *Context*, choose *Create*, then *Node*. Enter MAIN for the name of the node and hit enter.



Node Name	MAIN
Interface Node	No
Input Element (Ext.)	No
Dictionary structure	
Cardinality	1..1
Selection	0..1
Init. Lead Selection	Yes
Singleton	Yes
Supply Function	

Node Name	CONTEXT
Input Element (Ext.)	☐
Dictionary structure	
Cardinality	1..1

Create an attribute under the MAIN node. Right-click on MAIN, choose *Create*, and then *Attribute*. Enter NAME as the name of the attribute with type string. Hit enter. Save and activate your work.

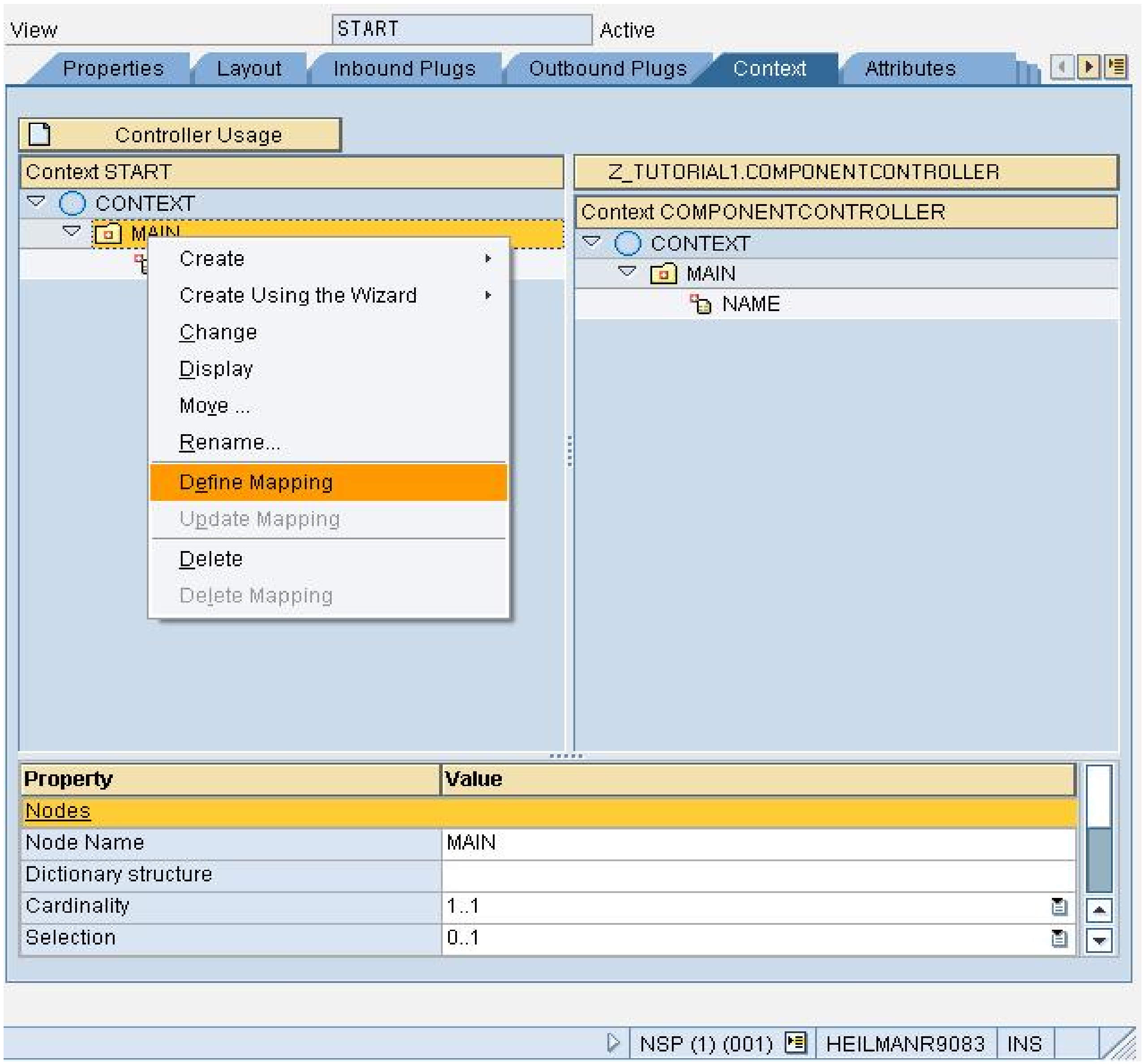


Attribute Name	NAME
Type assignment	Type
Type	String
Read-only	No
Primary Attribute	
Default Value	
Input Help Mode	Automatically
Determined Search Help	

Step 5 – Data Mapping

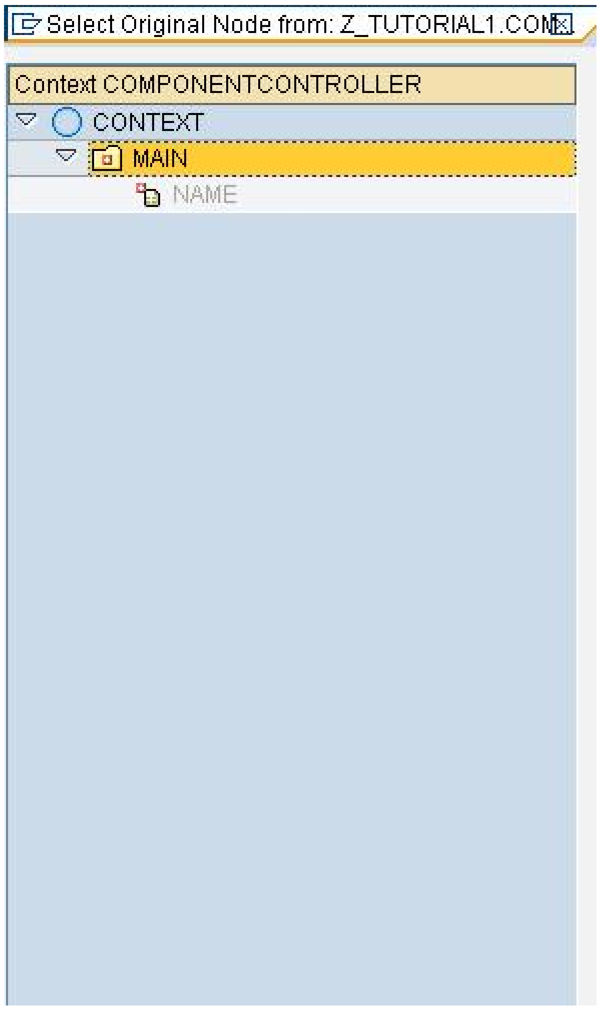
Data mapping is used to link data across multiple views using the component controller.

Map the context of the views to the context of the component controller. Double-click on the START view from the view folder of the object tree to the left. Click on the *Context* tab, right click on the MAIN node of the view controller and choose *Define Mapping*.



Property	Value
Nodes	
Node Name	MAIN
Dictionary structure	
Cardinality	1..1
Selection	0..1

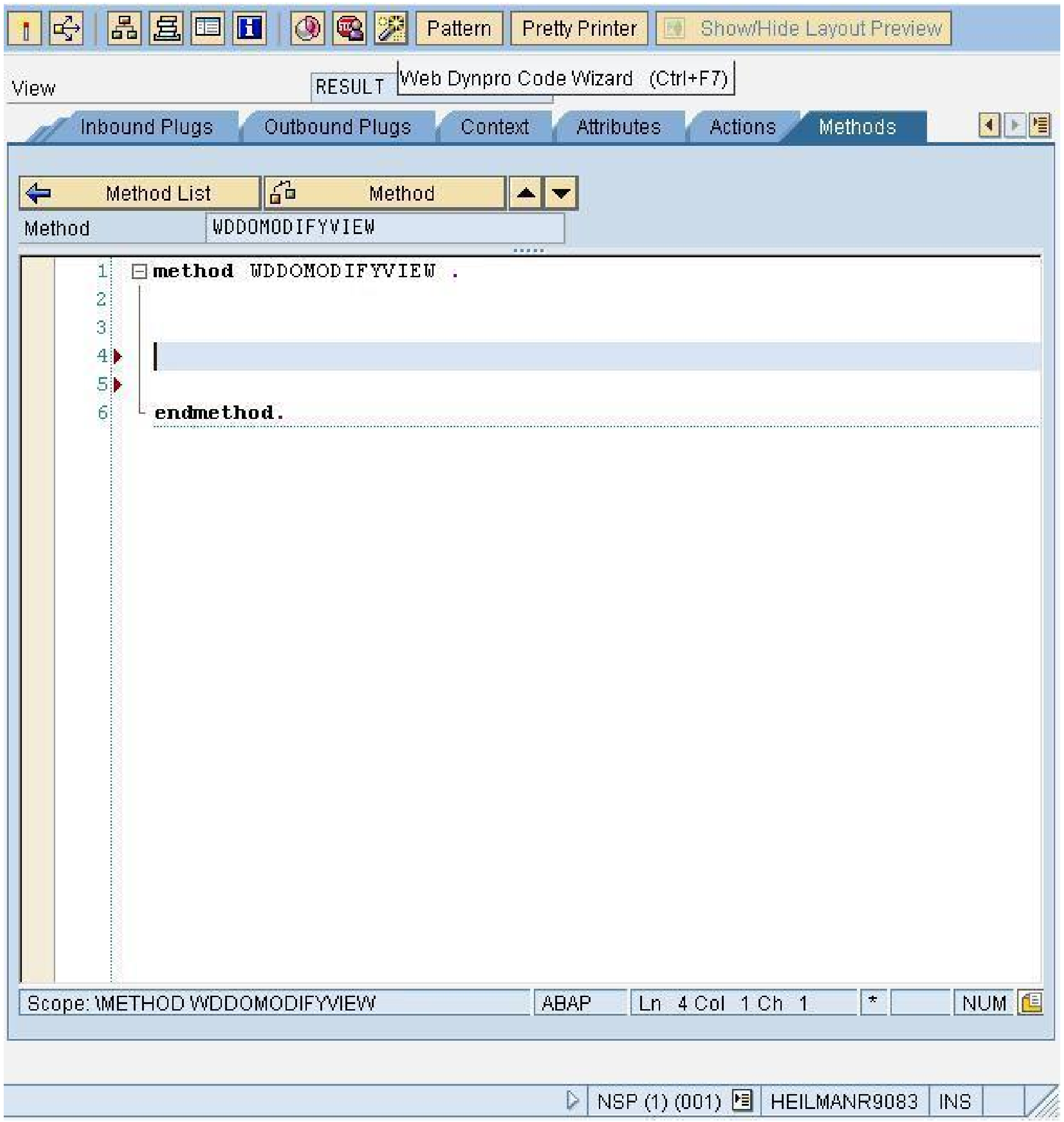
A dialog will appear, choose the MAIN node of the component controller context.


Select Original Node from: Z_TUTORIAL1.COM
Context COMPONENTCONTROLLER
CONTEXT
MAIN
NAME

Do the same for the context of the Result View.

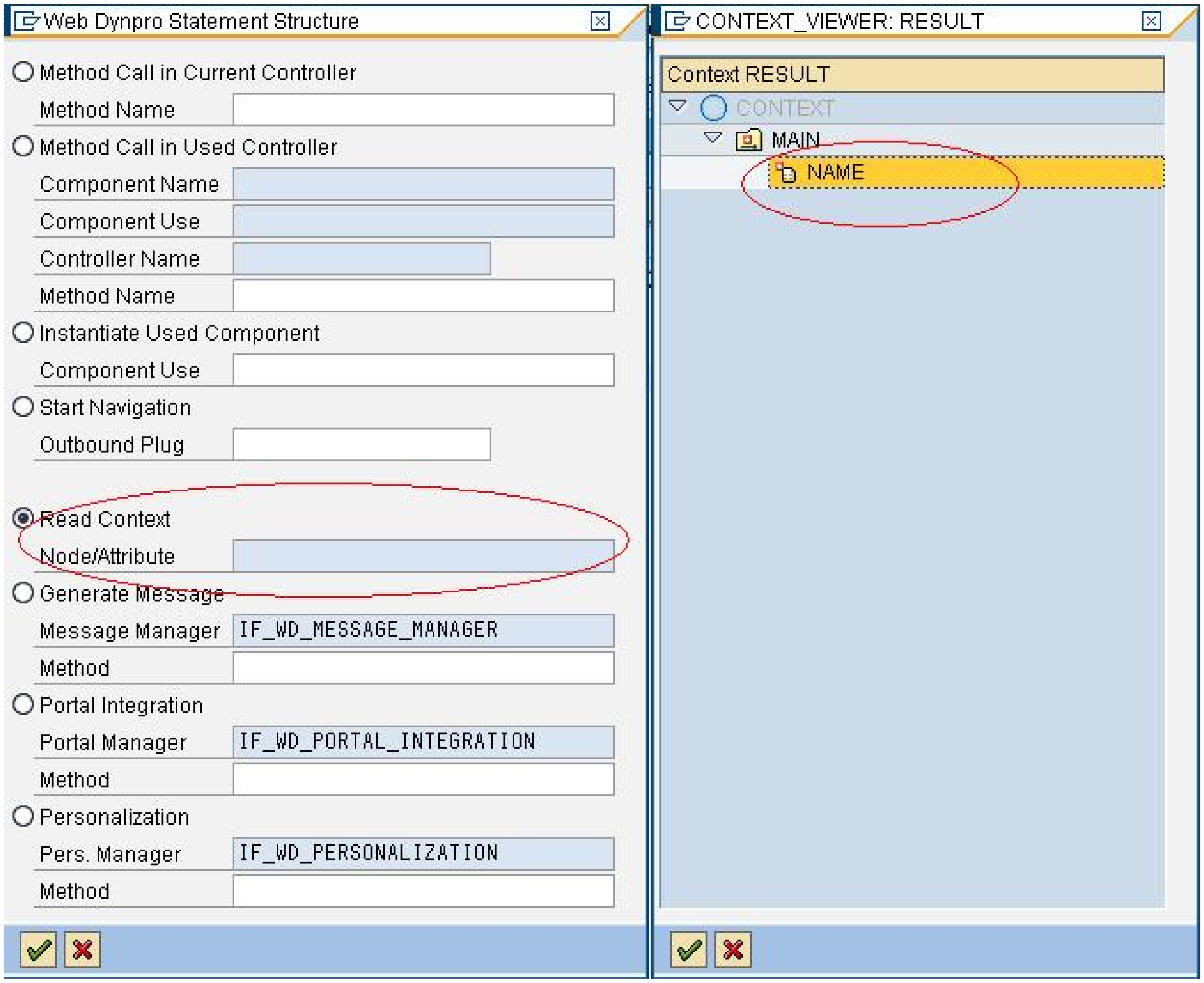
Step 6 – Accessing Data from the Component Controller

Access the data that the user will enter in the START view and prepare it for displaying in the RESULT view. Also clear out the value when leaving the RESULT view. Double click on the RESULT view from the object tree at the left. Click on the *Methods* tab. Double click on the WDDOMODIFYVIEW method. Click on the Web Dynpro Code Wizard on the application toolbar.



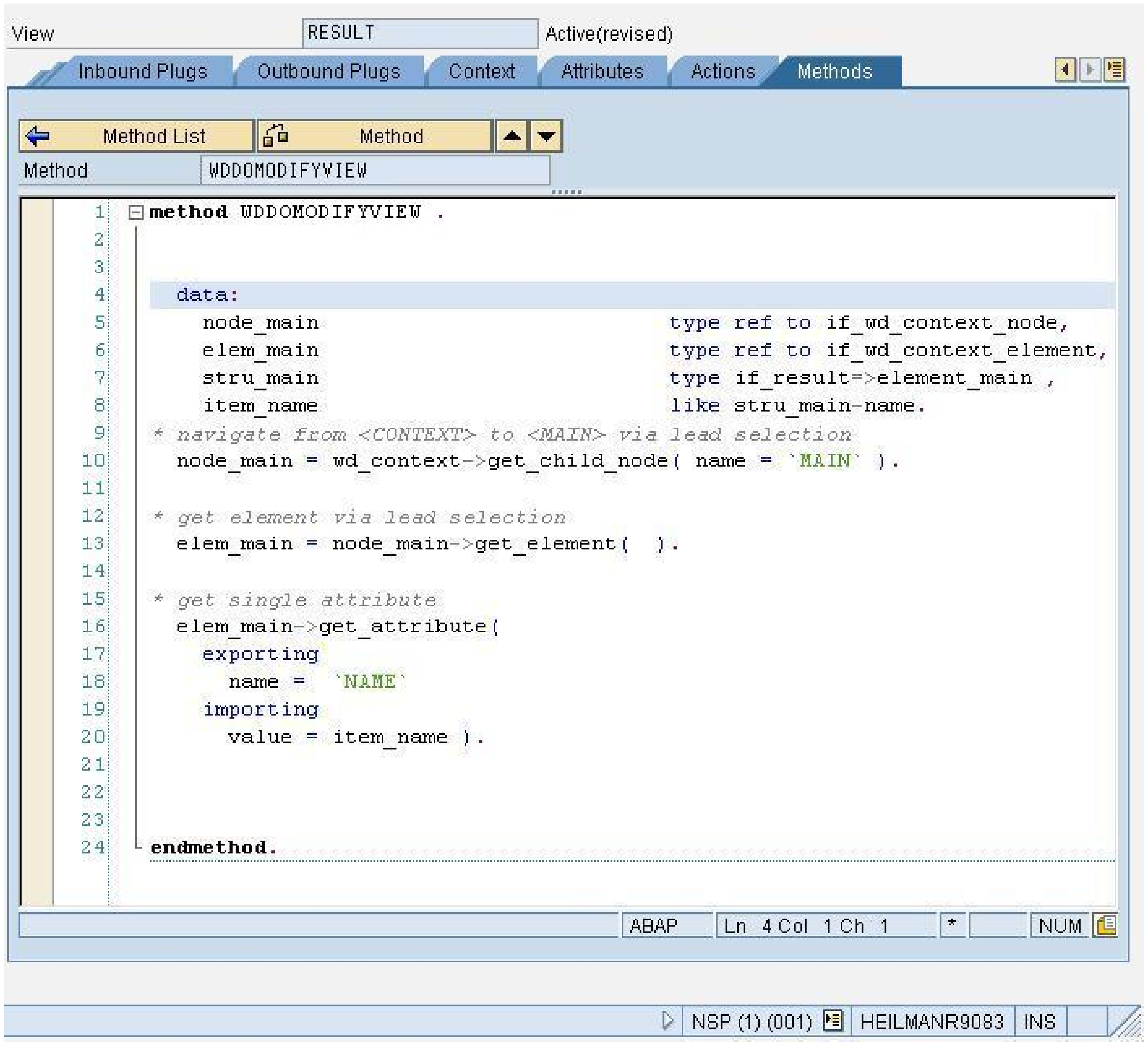
```
method WDDOMODIFYVIEW .



endmethod.
```

A dialog will appear. Click the radio button for *Read Context*. Put the cursor in the field next to it, and press F4. A dialog will appear, select the NAME attribute from the MAIN node. Hit enter to select the attribute, then hit enter again.



○ Method Call in Current Controller	
Method Name	
○ Method Call in Used Controller	
Component Name	
Component Use	
Controller Name	
Method Name	
○ Instantiate Used Component	
Component Use	
○ Start Navigation	
Outbound Plug	
● Read Context	
Node/Attribute	
○ Generate Message	
Message Manager	IF_WD_MESSAGE_MANAGER
Method	
○ Portal Integration	
Portal Manager	IF_WD_PORTAL_INTEGRATION
Method	
○ Personalization	
Pers. Manager	IF_WD_PERSONALIZATION
Method	

The code has been generated.



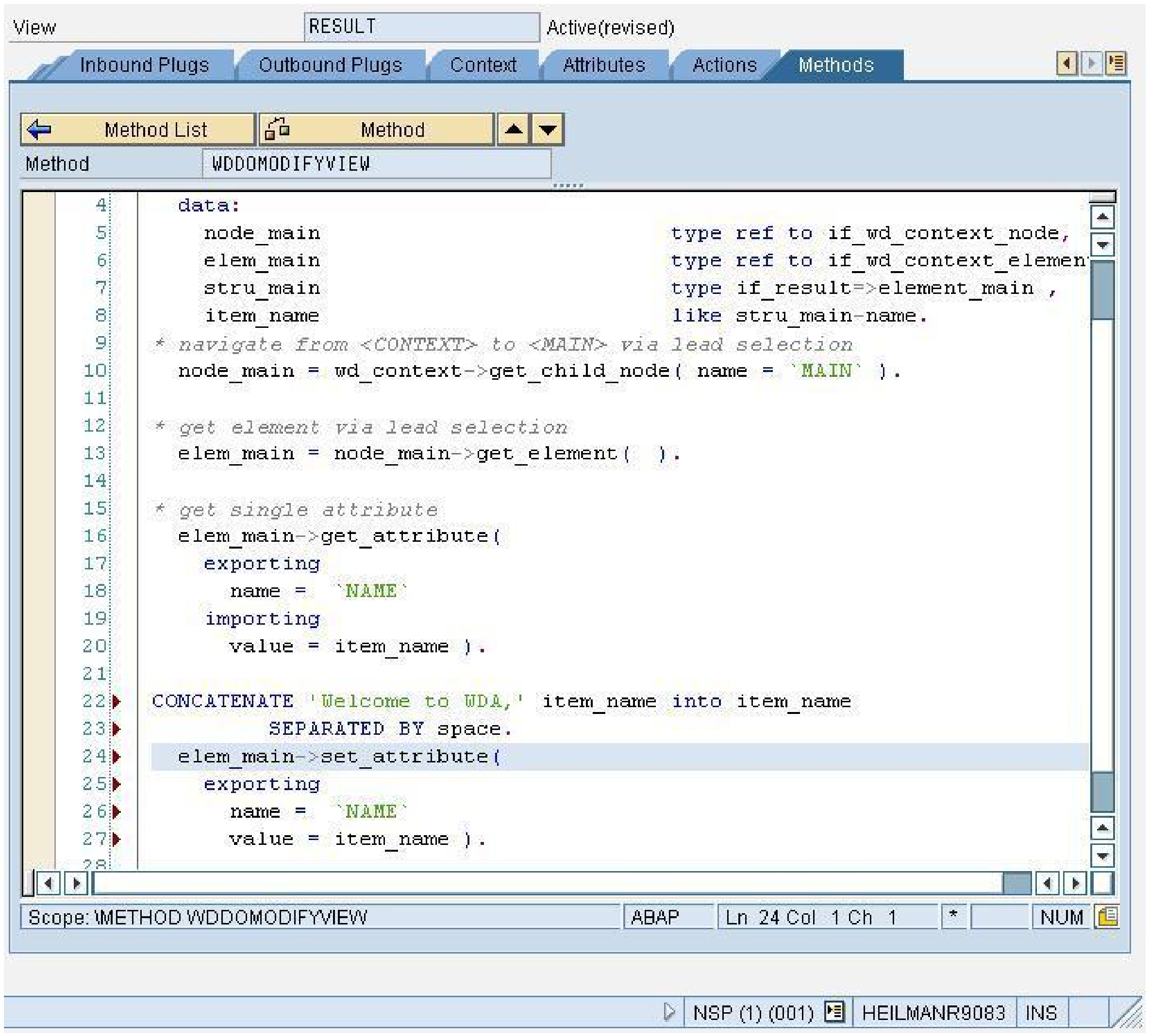
```
method WDDOMODIFYVIEW .


  data:
    node_main                               type ref to if_wd_context_node,
    elem_main                               type ref to if_wd_context_element,
    stru_main                               type if_result=>element_main ,
    item_name                               like stru_main-name.
* navigate from <CONTEXT> to <MAIN> via lead selection
  node_main = wd_context->get_child_node( name = `MAIN` ).

* get element via lead selection
  elem_main = node_main->get_element(  ).

* get single attribute
  elem_main->get_attribute(
    exporting
      name =  `NAME`
    importing
      value = item_name ).



endmethod.
```

Add the following lines of code for manipulating the value entered by the user. Add the words, “Welcome to WDA,” to the value and set it to the context. Add the following code to the end of the method.

```
CONCATENATE 'Welcome to WDA,' item_name into item_name
        SEPARATED BY space.
```

```
  elem_main->set_attribute(
    exporting
      name =  `NAME`
      value = item_name ).
```

The method should now have the following code.



View RESULT Active(revised)

Inbound Plugs | Outbound Plugs | Context | Attributes | Actions | Methods

Method List | Method

Method WDDOMODIFYVIEW

```
  data:
    node_main                               type ref to if_wd_context_node,
    elem_main                               type ref to if_wd_context_elemen
    stru_main                               type if_result=>element_main ,
    item_name                               like stru_main-name.
* navigate from <CONTEXT> to <MAIN> via lead selection
  node_main = wd_context->get_child_node( name = `MAIN` ).

* get element via lead selection
  elem_main = node_main->get_element(  ).

* get single attribute
  elem_main->get_attribute(
    exporting
      name =  `NAME`
    importing
      value = item_name ).

CONCATENATE 'Welcome to WDA,' item_name into item_name
          SEPARATED BY space.
  elem_main->set_attribute(
    exporting
      name =  `NAME`
      value = item_name ).
```

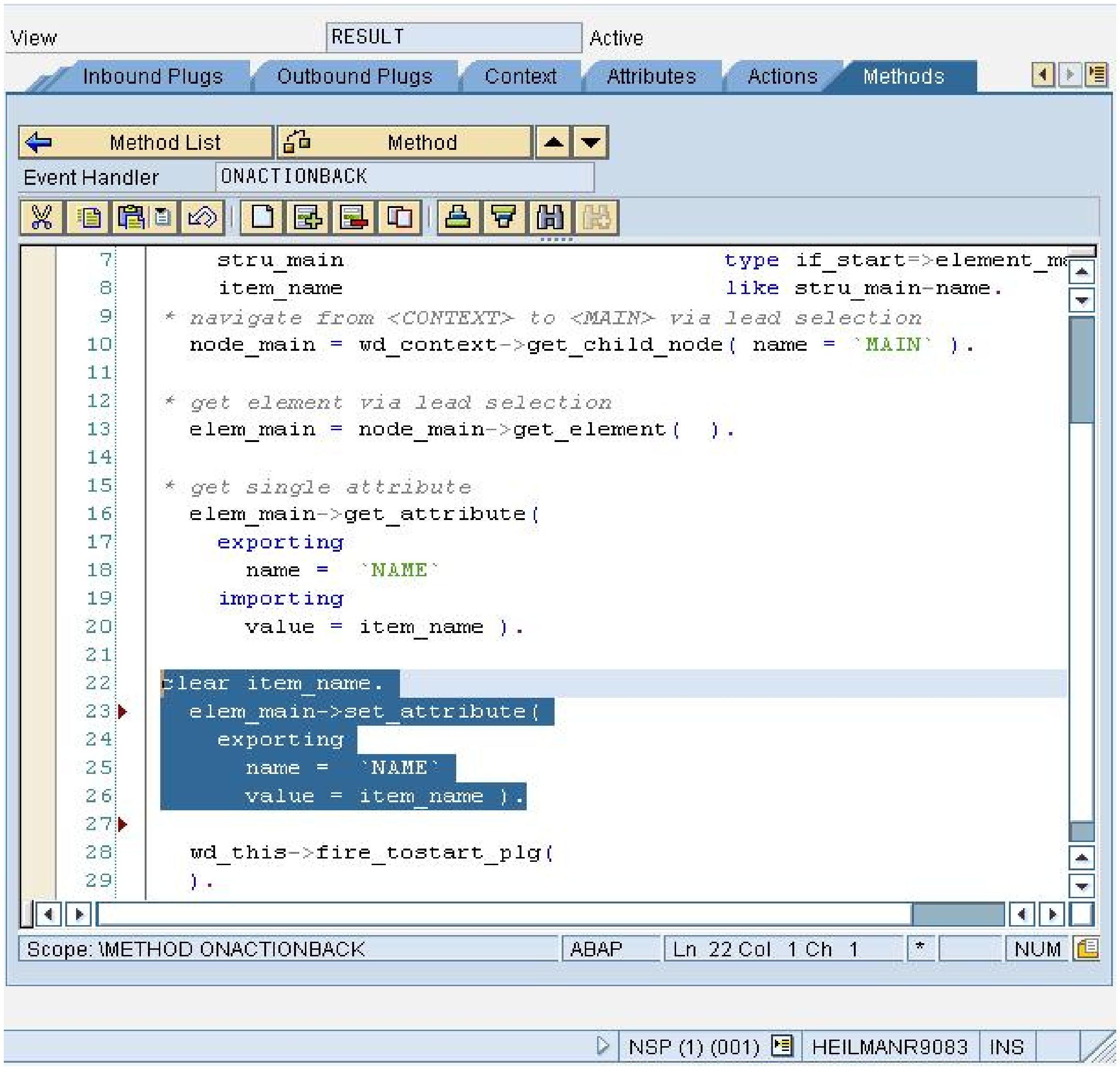
Scope: \METHOD WDDOMODIFYVIEW | ABAP | Ln 24 Col 1 Ch 1 | * | NUM

NSP (1) (001) | HEILMANR9083 | INS

Click on the *Method List.* Double click on the method ONACTIONBACK. Use the wizard to generate the code for reading the context. Add the following code before the method call *fire_to_startplug*.

```
clear item_name.

  elem_main->set_attribute(
    exporting
      name =  `NAME`
      value = item_name ).
```

The code should now look like this. Save and activate.



```
    stru_main                              type if_start=>element_ma
    item_name                              like stru_main-name.
* navigate from <CONTEXT> to <MAIN> via lead selection
  node_main = wd_context->get_child_node( name = `MAIN` ).

* get element via lead selection
  elem_main = node_main->get_element(  ).

* get single attribute
  elem_main->get_attribute(
    exporting
      name =  `NAME`
    importing
      value = item_name ).

clear item_name.
  elem_main->set_attribute(
    exporting
      name =  `NAME`
      value = item_name ).

  wd_this->fire_tostart_plg(
  ).
```

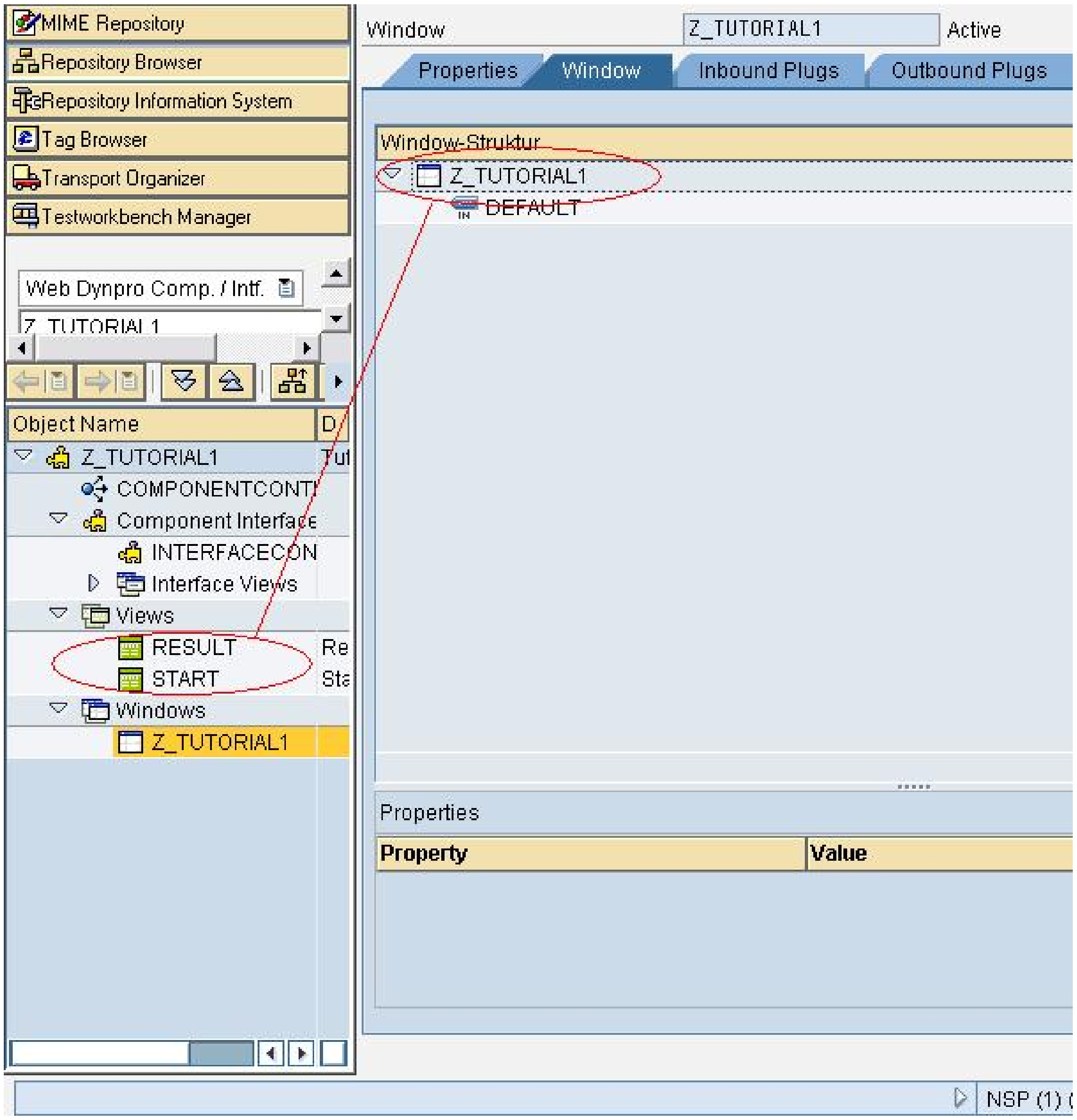
Scope: \METHOD ONACTIONBACK | ABAP | Ln 22 Col 1 Ch 1 | NUM

NSP (1) (001) | HEILMANR9083 | INS

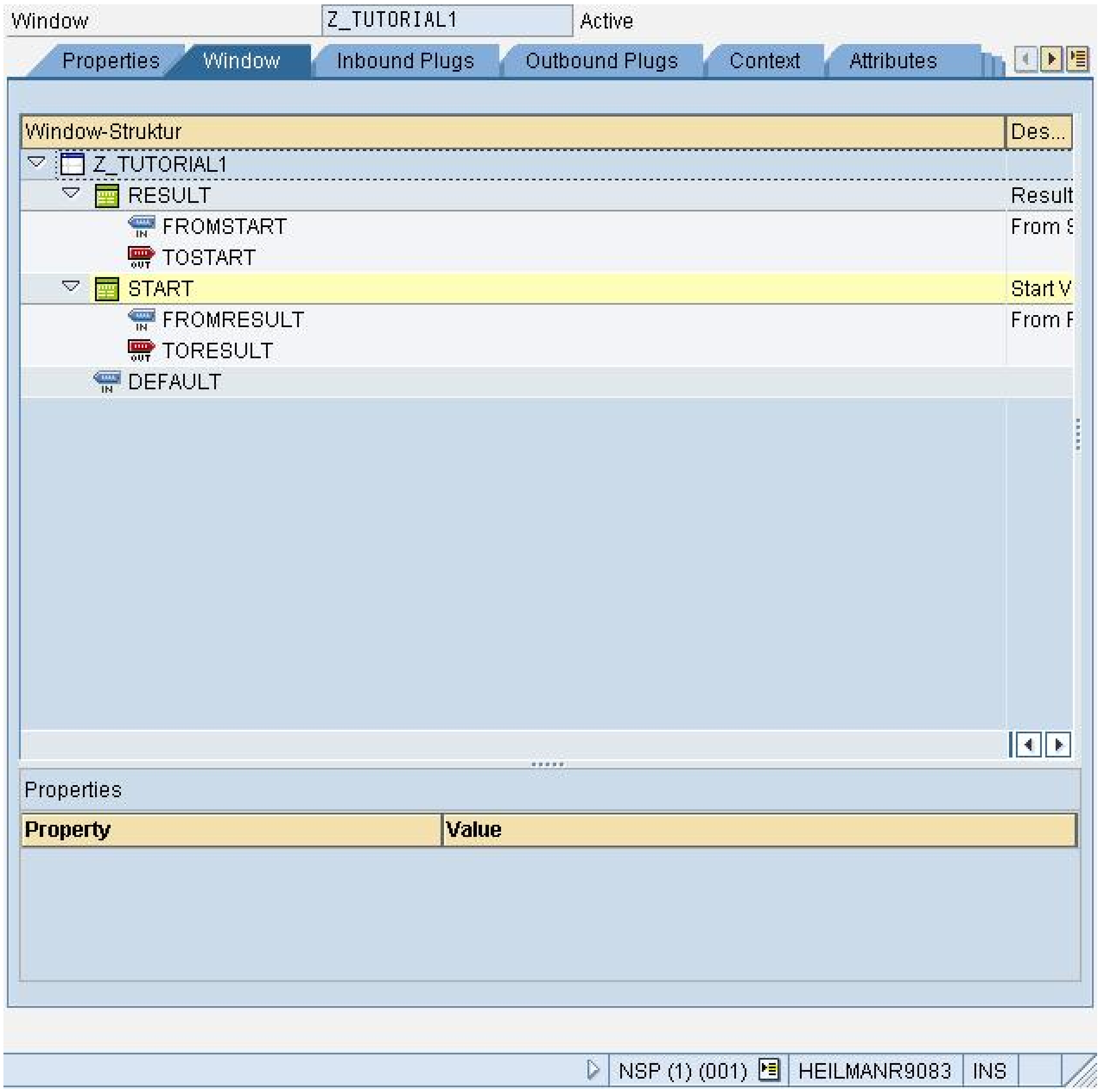
Step 7 – Define the Navigational Schema

Navigational Schemas allow you to define the navigational flow between views. Defining inbound and outbound plugs provide entry and exit points. The navigational links define the sequence in which the views are displayed.

Double-click the Window Z_TUTORIAL1 from the object tree at the left. Drag and drop each view into the Window.



The views have been moved into the window. Open the sub objects to reveal the plugs.

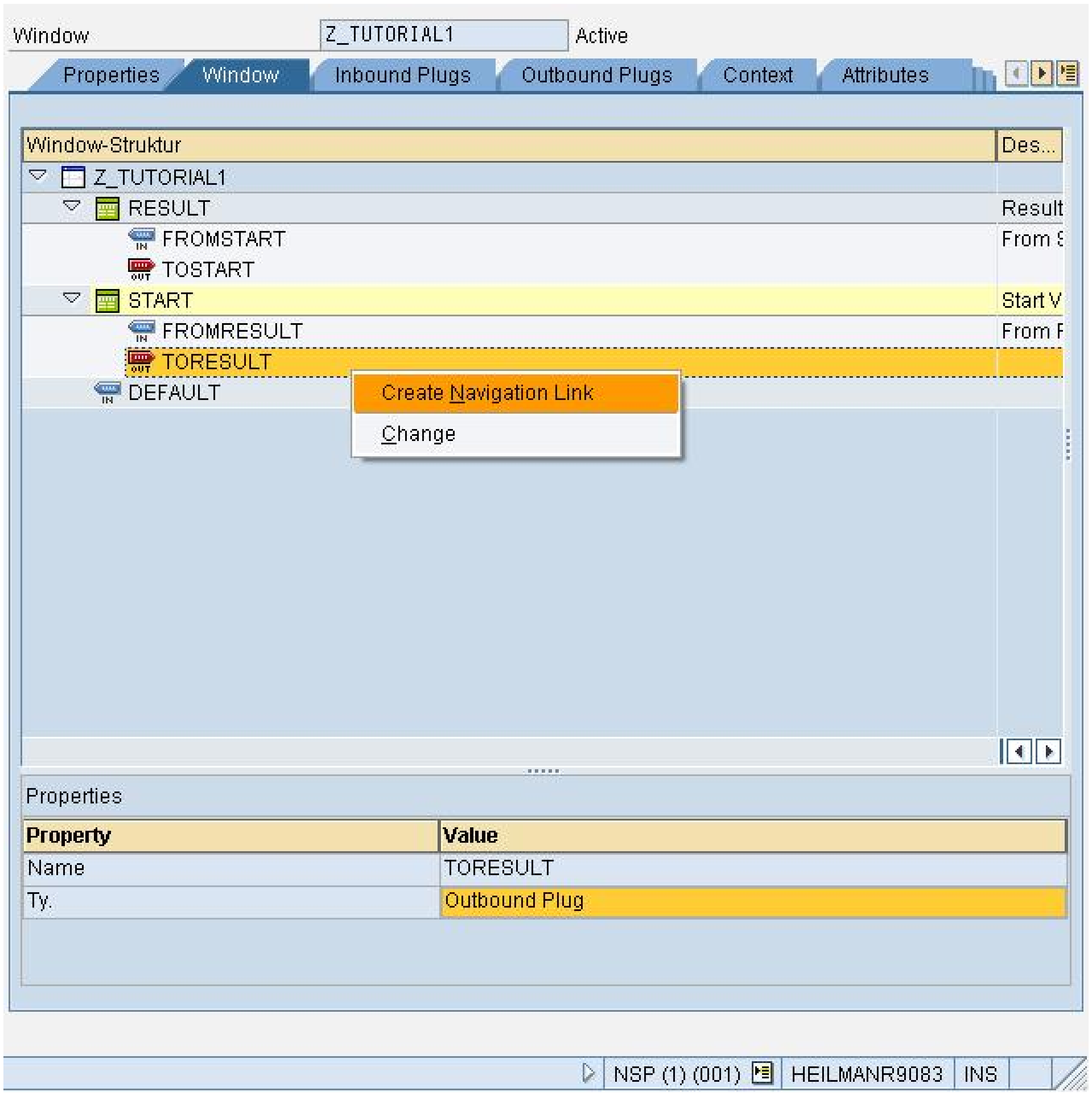

Window Z_TUTORIAL1 Active

Window-Struktur	Des...
Z_TUTORIAL1	
RESULT	Result
FROMSTART	From S
TOSTART	
START	Start V
FROMRESULT	From F
TORESULT	
DEFAULT	

Properties

Property	Value

Create the navigation links. Right-click on the TORESULT outbound plug of the START view and choose *Create Navigational Link*.

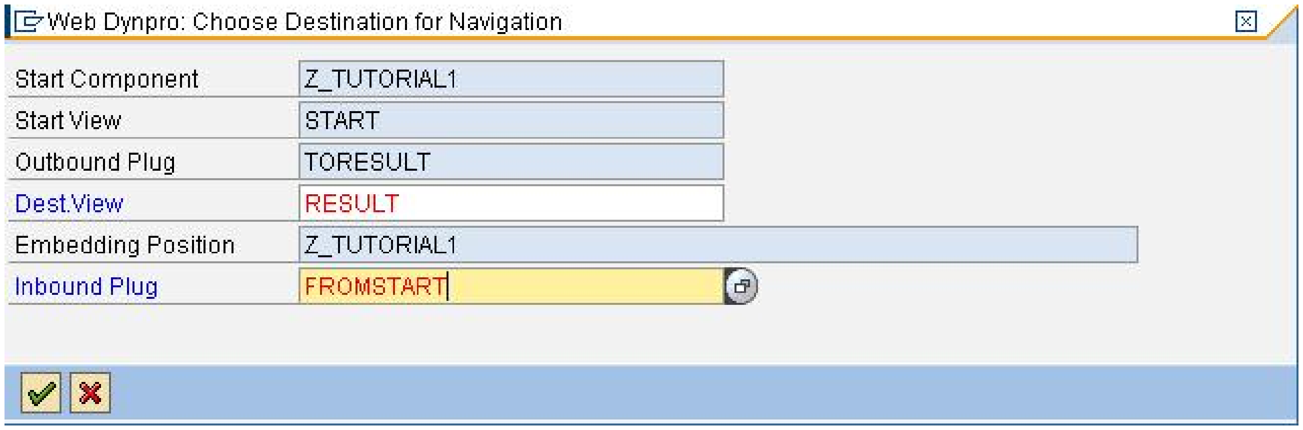


Window-Struktur	Des...
Z_TUTORIAL1	
RESULT	Result
FROMSTART	From S
TOSTART	
START	Start V
FROMRESULT	From F
TORESULT	
DEFAULT	

Properties

Property	Value
Name	TORESULT
Ty.	Outbound Plug

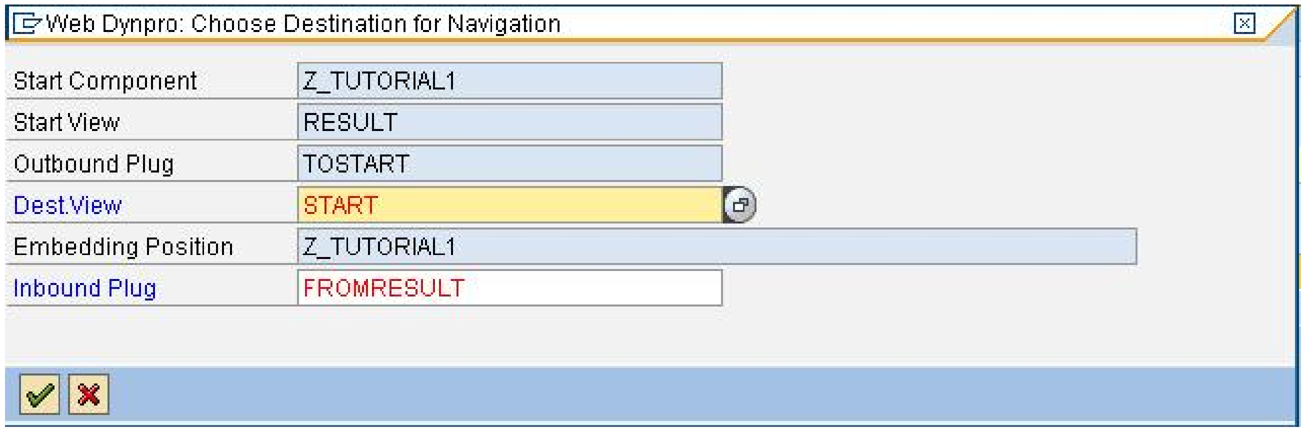
A dialog will appear. Enter the values as seen here. Hit enter.



Web Dynpro: Choose Destination for Navigation

Start Component	Z_TUTORIAL1
Start View	START
Outbound Plug	TORESULT
Dest.View	RESULT
Embedding Position	Z_TUTORIAL1
Inbound Plug	FROMSTART

Right-click on the TOSTART outbound plug of the RESULT view and choose *Create Navigational Link*. A dialog will appear, enter the values as seen here and hit enter.



Web Dynpro: Choose Destination for Navigation

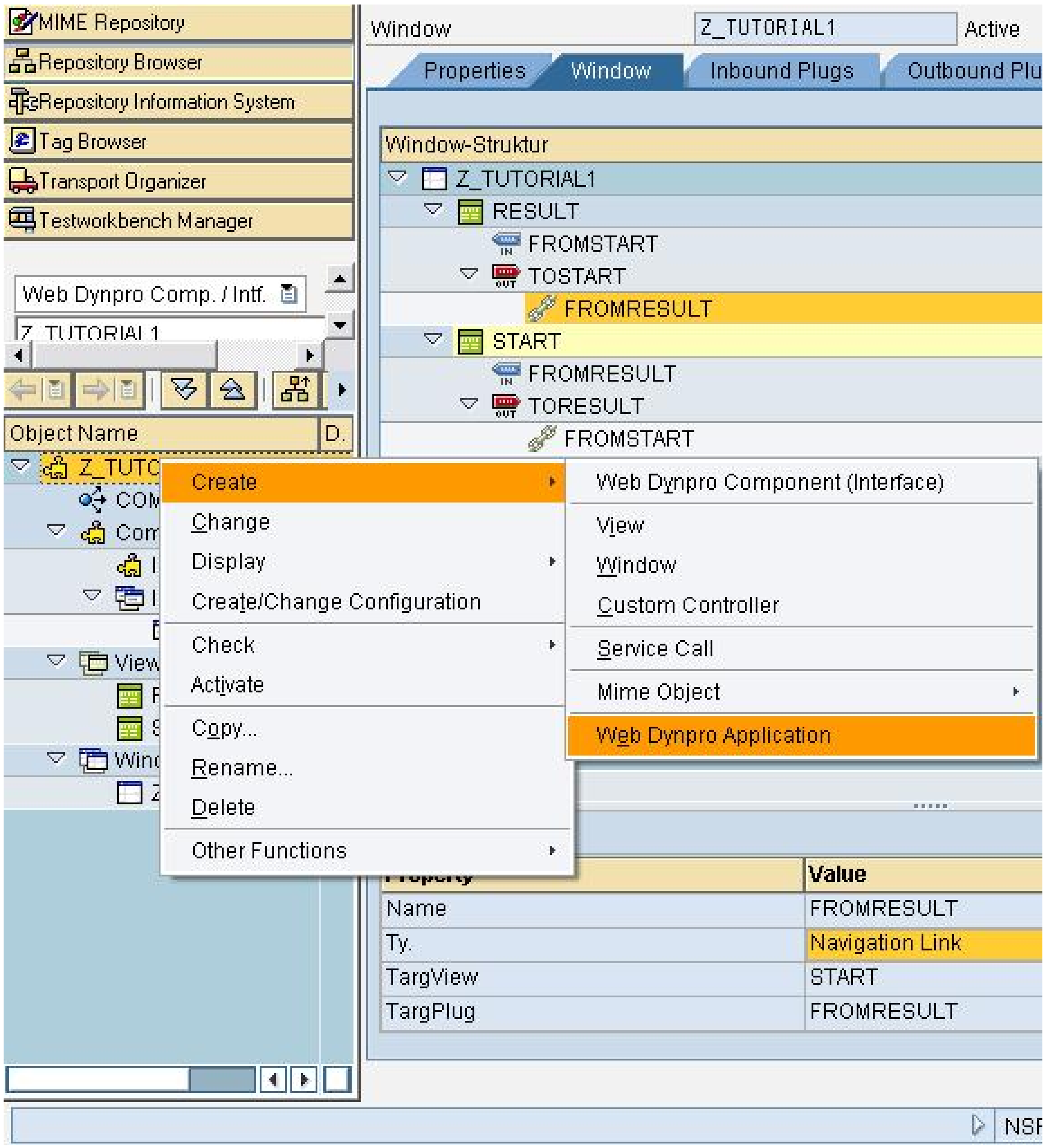
Start Component	Z_TUTORIAL1
Start View	RESULT
Outbound Plug	TOSTART
Dest.View	START
Embedding Position	Z_TUTORIAL1
Inbound Plug	FROMRESULT

Save and activate.

Step 8 – Creating the Application

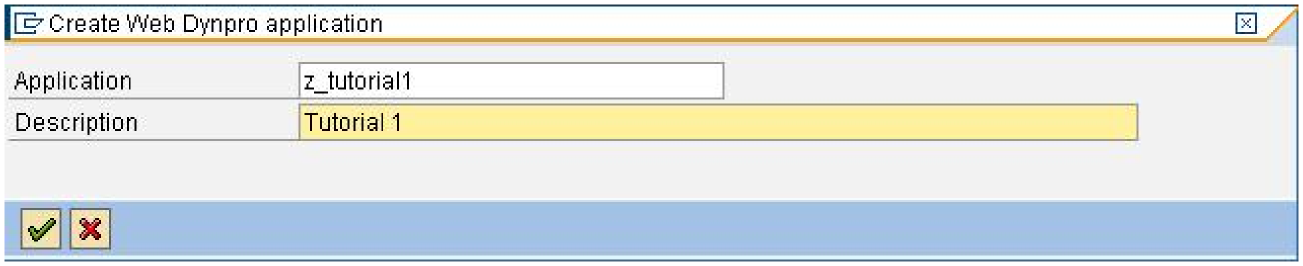
The application is the object that allows the WDA application to be addressed and displayed in the browser. When the application is created, a URL is automatically generated.

Create the application by right-clicking the WDA Object from the object tree at the left.



Property	Value
Name	FROMRESULT
Ty.	Navigation Link
TargView	START
TargPlug	FROMRESULT

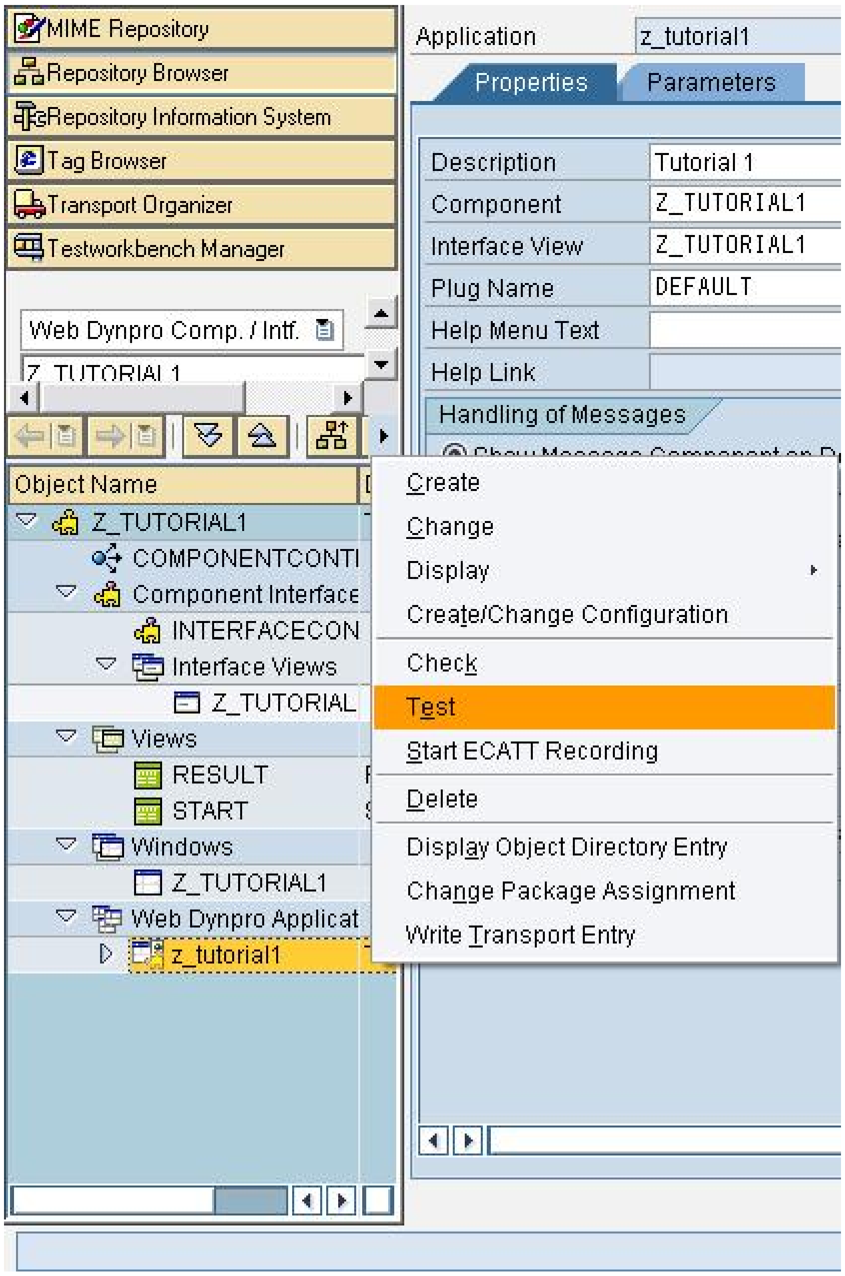
A dialog will appear. Enter the values as seen here and hit enter. After the application has been created, then click save.



Application	z_tutorial1
Description	Tutorial 1

Step 9 – Testing the WDA Application

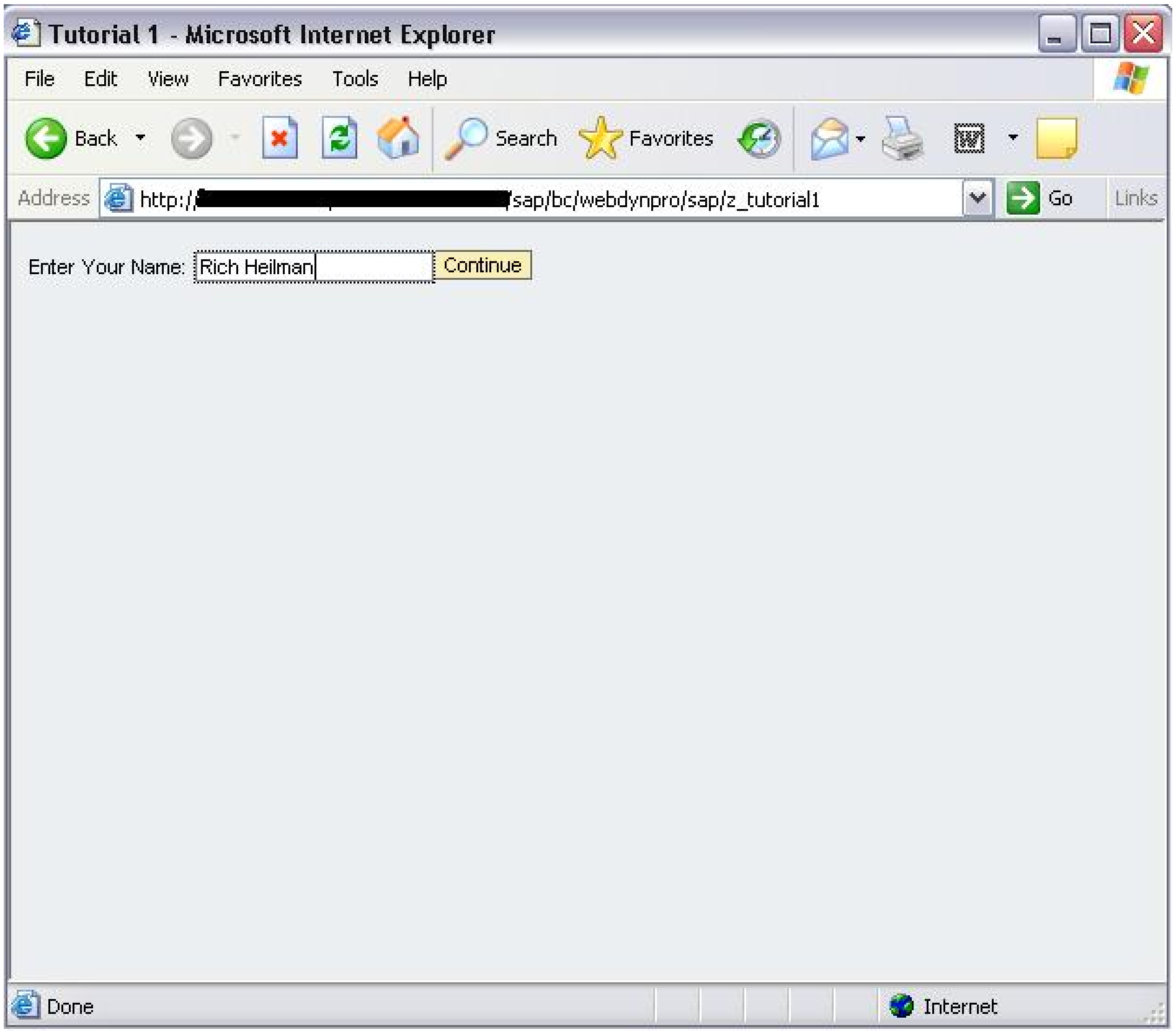
Test the application by right clicking on the application name under the Web Dynpro Application folder in the object tree to the left. Choose *Test*.



Application: z_tutorial1

Description	Tutorial 1
Component	Z_TUTORIAL1
Interface View	Z_TUTORIAL1
Plug Name	DEFAULT
Help Menu Text	
Help Link	

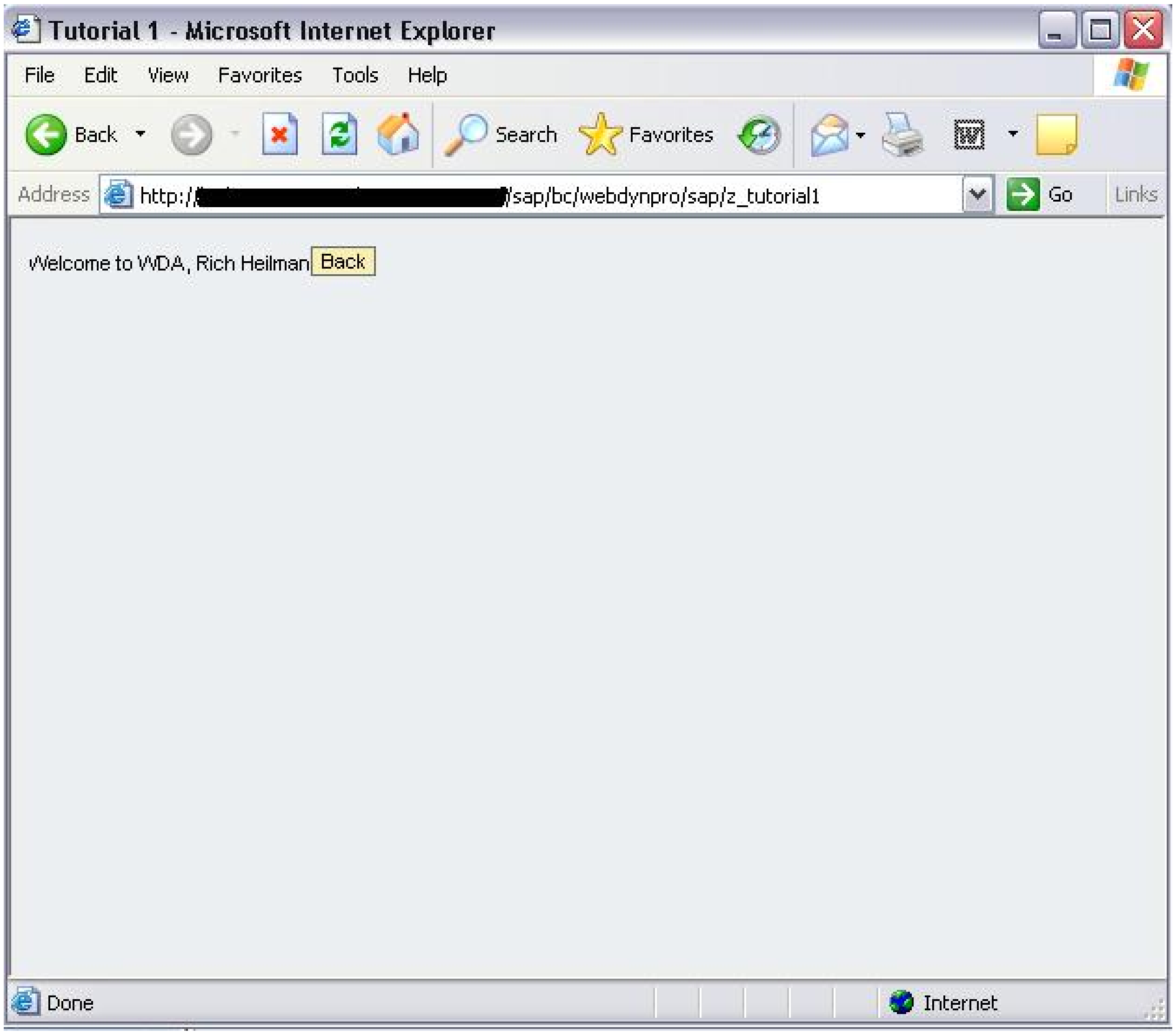
A browser will open with the application running inside. Enter your name and click the *Continue* button.


Tutorial 1 - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Address
Go
Links
Continue
Done
Internet

Address: http://[redacted]/sap/bc/webdynpro/sap/z_tutorial1

Enter Your Name: Rich Heilman

The next view will be displayed with the message. Click the Back button to go back to the first view.


Tutorial 1 - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://[redacted]/sap/bc/webdynpro/sap/z_tutorial1 Go Links
Back
Done
Internet

Welcome to WDA, Rich Heilman

Congratulations, you have created your first Web Dynpro for ABAP application.

Author Bio

Rich Heilman is an ABAP/J2EE Software Engineer/Analyst for Yorktowne Cabinetry, Inc. based in Red Lion, Pennsylvania, USA. He has a total of nine years experience in the IT industry. He has spent the past five years studying ABAP and Java.