

Version 1.0

Date 27.06.2016

Status Completed

Number 1

Initial version by Igor Muntoreanu

1) Homework (item14 from the Practical Guide):

- a. Create a button in the V\_DETAILS to return to the V\_SELECT view.

Processed to Number 2)

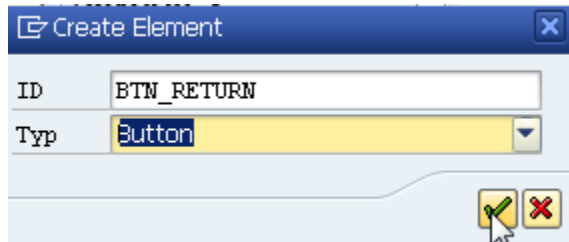
- b. Create a validation if the entry data is valid. If not display a POP UP and freeze in the V\_SELECT view.

Processed to Number 3)

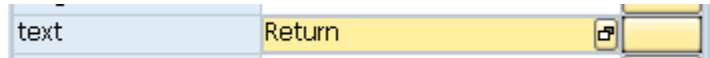
2) Return Button to get back to the initial view.

- a. Navigate to the V\_DETAILS view

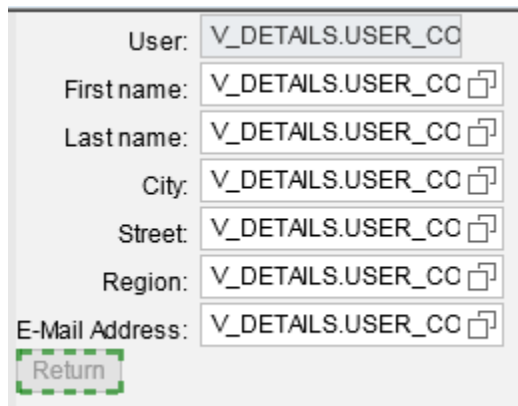
- b. Create a new button in the ROOTUIELEMENTCONTAINER:



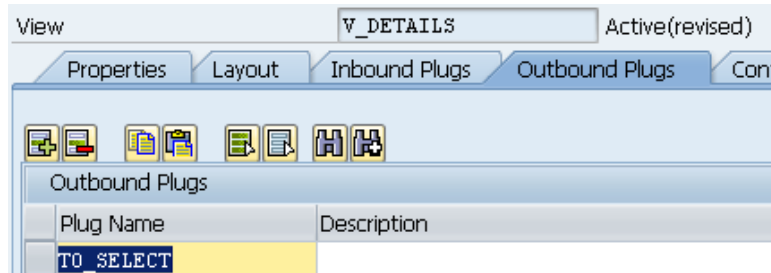
- c. Name the new button as "RETURN":



- d. Your view will be like this now:

A screenshot of the V\_DETAILS view form. It contains several input fields: 'User', 'First name', 'Last name', 'City', 'Street', 'Region', and 'E-Mail Address'. Each field has the value 'V\_DETAILS.USER\_CC' and a small square icon to its right. At the bottom left, there is a 'Return' button with a green dashed border.

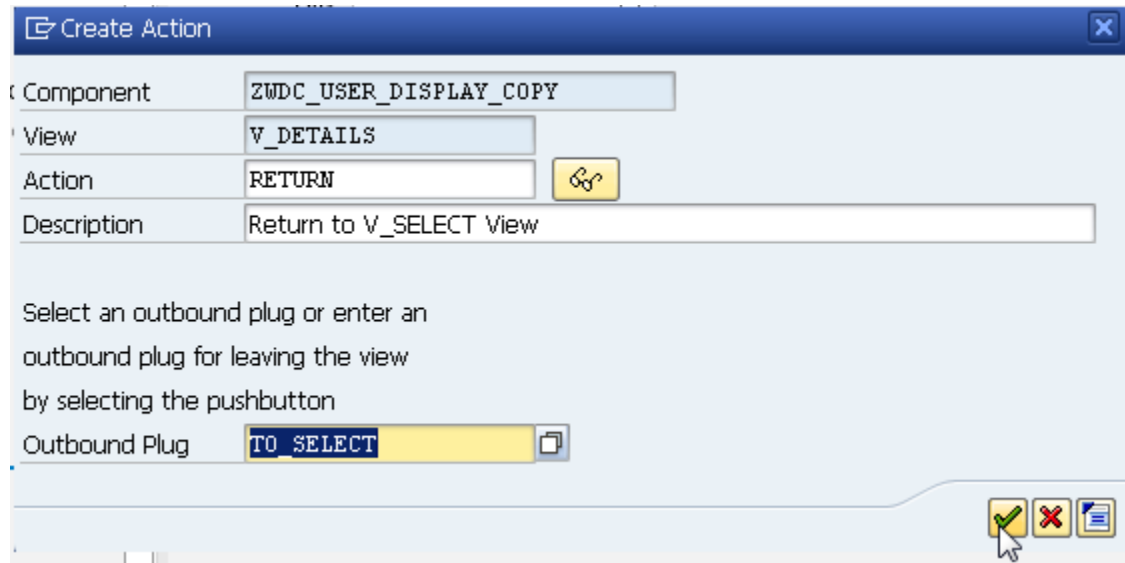
- e. Make sure that you have the Outbound Plug like below:



- f. Create an Action to the new button:



Action: RETURN  
Outbound Plug: TO\_SELECT  
Confirm/Enter



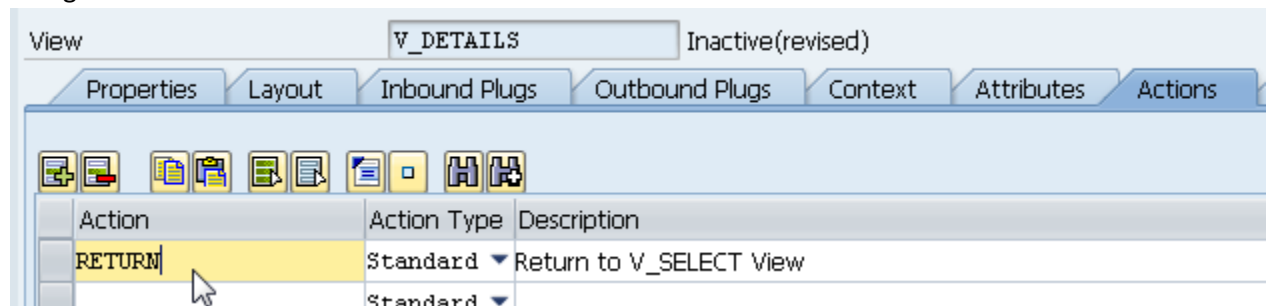
The 'Create Action' dialog box is shown with the following fields:

- Component: ZWDC\_USER\_DISPLAY\_COPY
- View: V\_DETAILS
- Action: RETURN
- Description: Return to V\_SELECT View

Below the fields, there is a text instruction: "Select an outbound plug or enter an outbound plug for leaving the view by selecting the pushbutton". Below this instruction, the 'Outbound Plug' field is set to TO\_SELECT.

At the bottom right, there are three icons: a green checkmark, a red X, and a document icon.

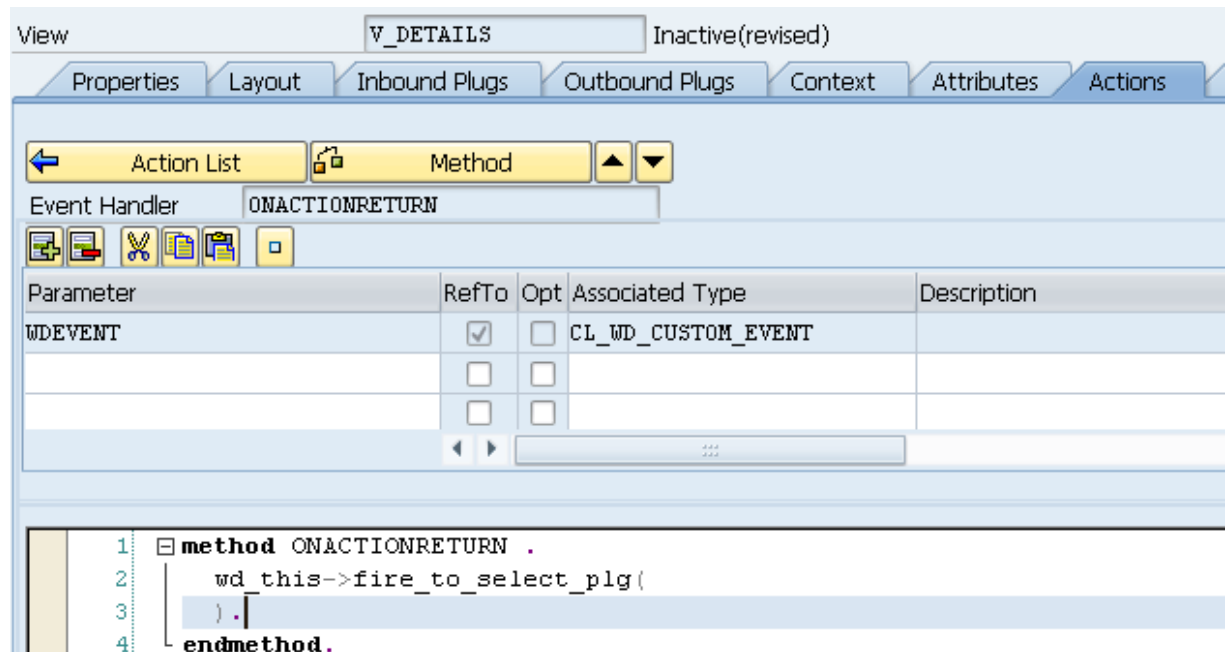
- g. Navigate to the Action tab and click twice in the new action created:



The screenshot shows the 'V\_DETAILS' view with the 'Actions' tab selected. The 'Action' list contains one entry:

Action	Action Type	Description
RETURN	Standard	Return to V_SELECT View

As you can see the pre-defined code was already generated:



The screenshot shows the 'V\_DETAILS' view with the 'Actions' tab selected. The 'Event Handler' is set to ONACTIONRETURN. The 'Parameter' list is as follows:

Parameter	RefTo	Opt	Associated Type	Description
WDEVENT	<input checked="" type="checkbox"/>	<input type="checkbox"/>	CL_WD_CUSTOM_EVENT	
	<input type="checkbox"/>	<input type="checkbox"/>		
	<input type="checkbox"/>	<input type="checkbox"/>		

Below the parameter list, the pre-defined code is shown:

```
1 method ONACTIONRETURN .  
2   wd_this->fire_to_select_plg(  
3     ).  
4 endmethod.
```

- h. Go to the window and check the navigation configuration the V\_DETAILS view should be like this:

The screenshot shows the WDA configuration interface. The 'Window' tab is active, and the 'W\_MAIN' window is selected. The 'Window Structure' pane shows the hierarchy: W\_MAIN > V\_DETAILS > FROM\_SELECT > TO\_SELECT > FROM\_DETAILS. The 'FROM\_DETAILS' view is highlighted in yellow. A red box highlights the 'TO\_SELECT' view, and a red arrow points from it to the 'Target View' property in the 'Properties' pane. The 'Properties' pane shows the following configuration for the 'FROM\_DETAILS' view:

Property	Value
Name	FROM_DETAILS
Type	Navigation Link
Target View	V_SELECT
TargPlug	FROM_DETAILS
Component of Target View	ZWDC_USER_DISPLAY_COPY
Embedding Position of Target	

- i. Activate all the objects and run your WDA application

The screenshot shows a user login form with the following fields and buttons:

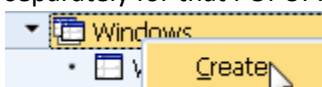
- \*User:
- Search

The screenshot shows a user registration form with the following fields and buttons:

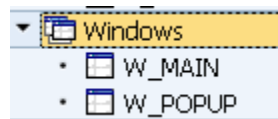
- User:
- First name:
- Last name:
- City:
- Street:
- Region:
- E-Mail Address:
- 

### 3) POPUP to validate the entered user

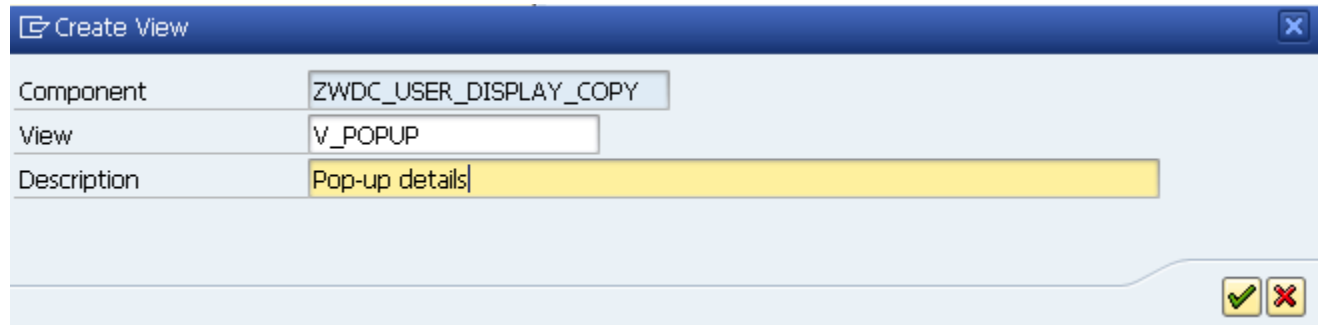
- a. Before anything, if we are going to create a new POPUP it will be necessary to create a WINDOW separately for that POPUP. Right click on the window and create a new one:



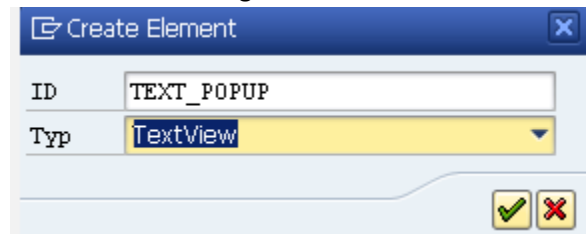
Result:



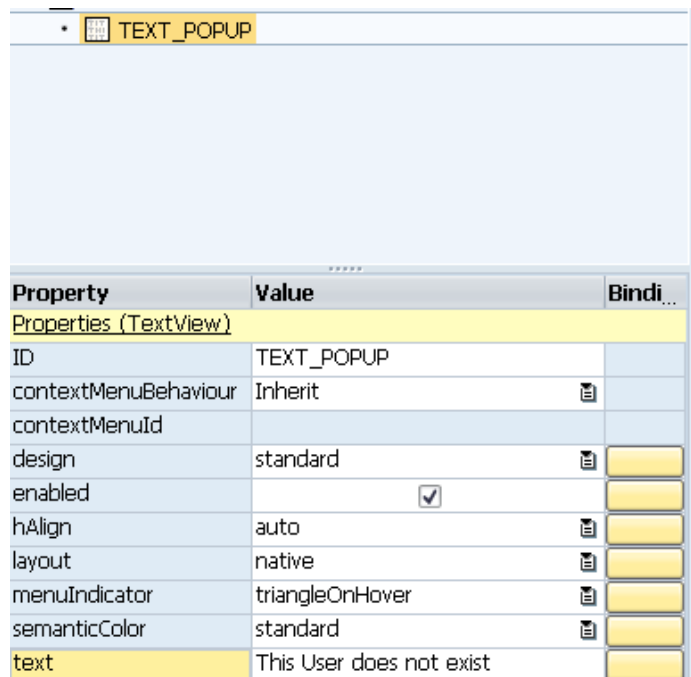
- b. Also, create a VIEW for this new window:



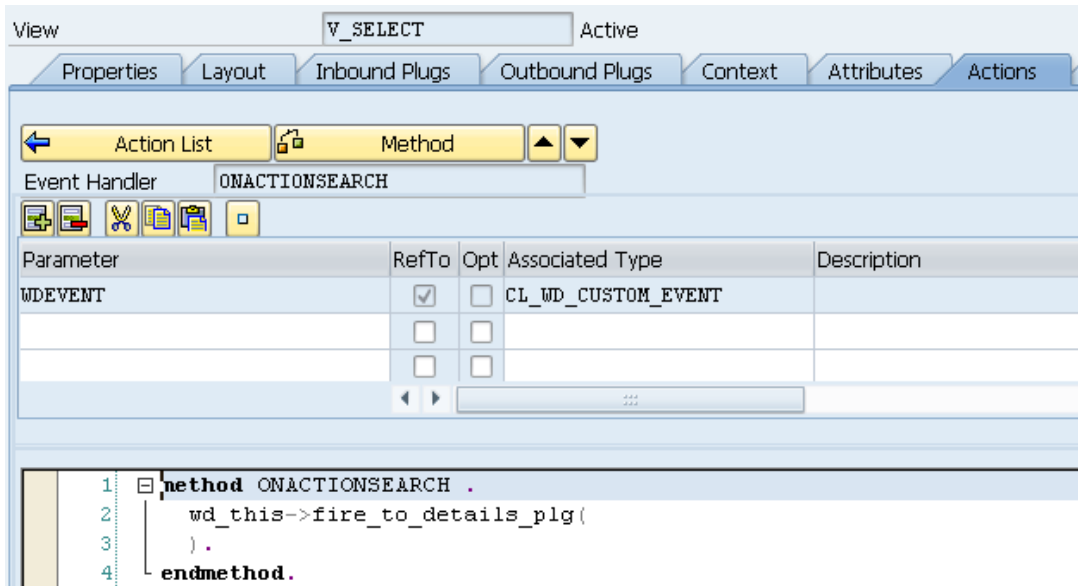
Create the following element in the ROOTUIELEMENTCONTAINER:




Give a simple text that will be shown in the new POPUP window:



- c. Navigate to the SEARCH action in the V\_SELECT View. (Search action is the action of the V\_SELECT Search Button).



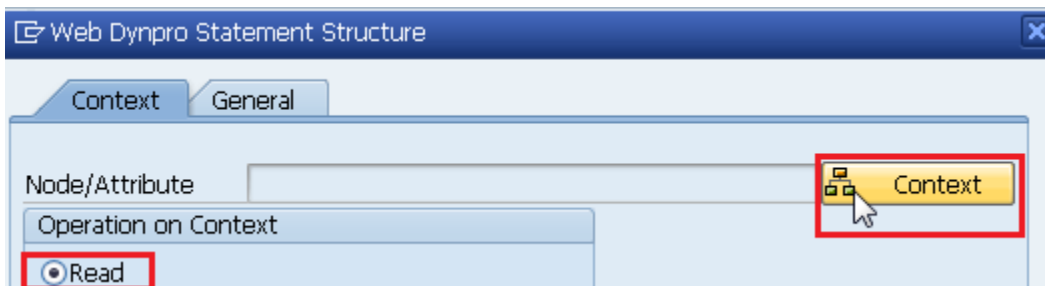
- d. Now, let's do some code. Click at the Wizard Toll 

The idea here is to check the entered value that the user put in the input field:

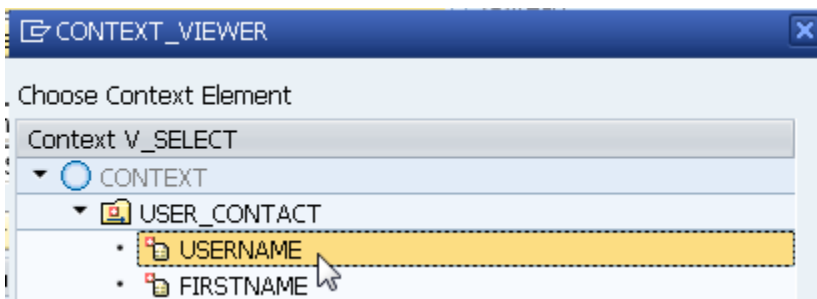


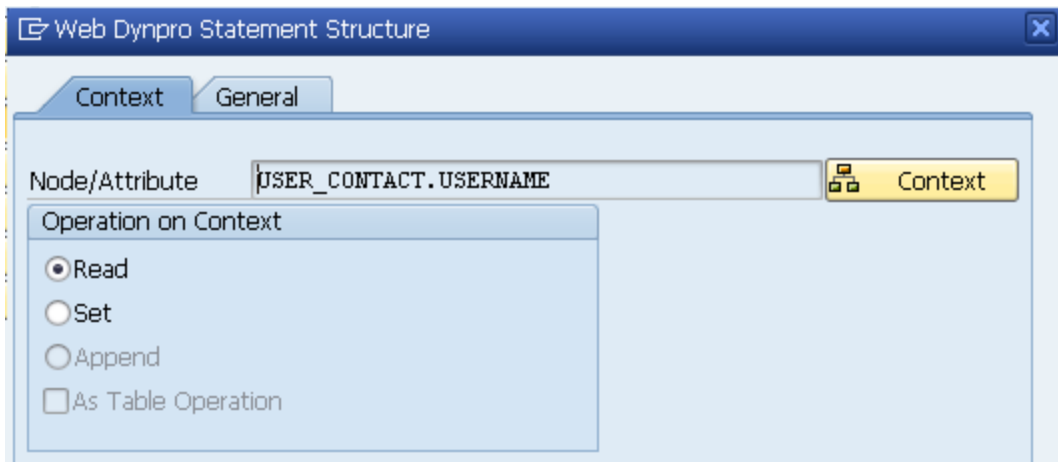
If it is Wrong, a popup will be displayed

Getting back to the Wizard, do:



Choose the USERNAME and press ENTER:





Press ENTER

Some CODE will be generated automatically:

```

1  method ONACTIONSEARCH .
2
3      DATA lo_nd_user_contact TYPE REF TO if_wd_context_node.
4
5      DATA lo_el_user_contact TYPE REF TO if_wd_context_element.
6      DATA ls_user_contact TYPE wd_this->element_user_contact.
7      DATA lv_username TYPE wd_this->element_user_contact-username.
8
9      * navigate from <CONTEXT> to <USER_CONTACT> via lead selection
10     lo_nd_user_contact = wd_context->get_child_node( name = wd_this->wdctx_user_contact ).
11
12     * @TODO handle non existant child
13     * IF lo_nd_user_contact IS INITIAL.
14     *     ENDIF.
15
16     * get element via lead selection
17     lo_el_user_contact = lo_nd_user_contact->get_element( ).
18     * @TODO handle not set lead selection
19     IF lo_el_user_contact IS INITIAL.
20     ENDIF.
21
22     * get single attribute
23     lo_el_user_contact->get_attribute(
24         EXPORTING
25             name = `USERNAME`
26         IMPORTING
27             value = lv_username ).
28
29
30
31     wd_this->fire_to_details_plg(
32     ).

```


By using the variable lv\_username, check if the user exists by using the BAPI:  
"BAPI\_USER\_EXISTENCE\_CHECK"

Your code should look like this now:

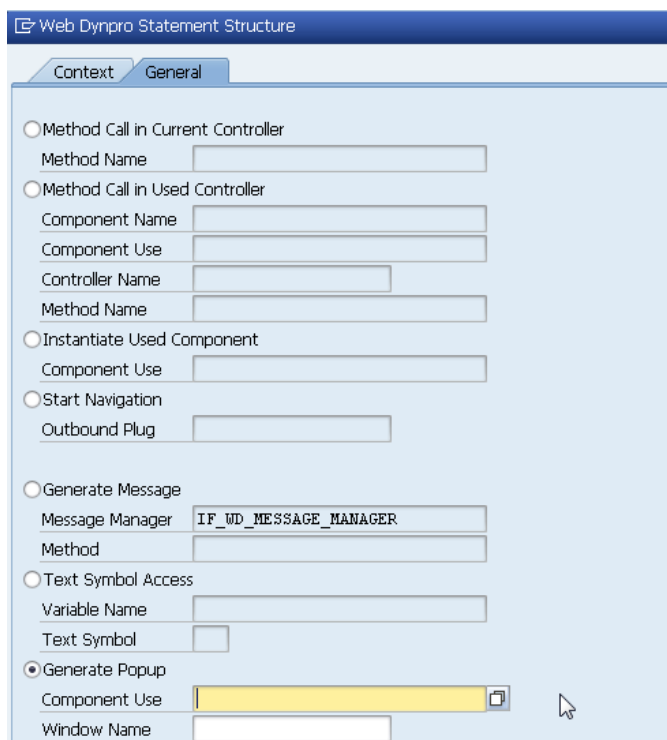
```

23 ~ get single attribute
24 lo_el_user_contact->get_attribute(
25     EXPORTING
26         name = `USERNAME`
27     IMPORTING
28         value = lv_username ).
29
30 IF lv_username IS NOT INITIAL.
31
32     CALL FUNCTION 'BAPI_USER_EXISTENCE_CHECK'
33     EXPORTING
34         username = lv_username
35     IMPORTING
36         return = ls_bapiret2.
37
38 IF ls_bapiret2-id = '01' AND ls_bapiret2-number = 124. " If the user does
39
40     ENDIF.
41
42 ENDIF.
43
44
45 wd_this->fire_to_details_plg(
46 ).
47 ENDMETHOD.

```

Put the cursor in the line 39 and click at the Wizard Toll .

Navigate until the GENERAL tab und choose the popup option like showed below:

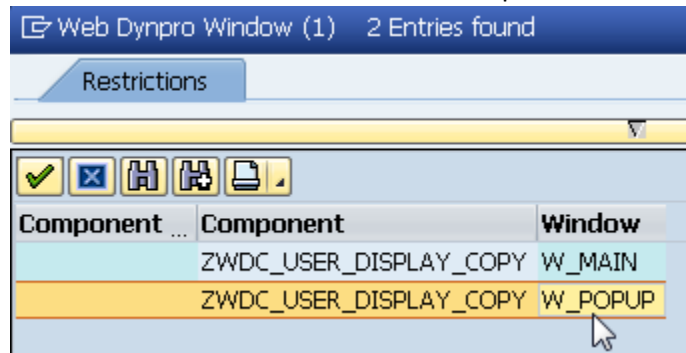


The image shows the 'Web Dynpro Statement Structure' dialog box with the 'General' tab selected. The 'Generate Popup' option is chosen, and the 'Component Use' field is highlighted in yellow.

Option	Field 1	Field 2	Field 3	Field 4
<input type="radio"/> Method Call in Current Controller	Method Name			
<input type="radio"/> Method Call in Used Controller	Component Name		Component Use	
	Controller Name			
	Method Name			
<input type="radio"/> Instantiate Used Component	Component Use			
<input type="radio"/> Start Navigation	Outbound Plug			
<input type="radio"/> Generate Message	Message Manager	IF_WD_MESSAGE_MANAGER	Method	
<input type="radio"/> Text Symbol Access	Variable Name		Text Symbol	
<input checked="" type="radio"/> Generate Popup	Component Use			
	Window Name			



Press the match-code and select our component



Press OK:

☒ Generate Popup

Component Use

Window Name

☐ Portal Integration

Portal Manager

Method

☐ Personalization

Pers. Manager

Method

Now your code should be like shown below:

**METHOD** onactionsearch .

```
DATA lo_nd_user_contact TYPE REF TO if_wd_context_node.

DATA lo_el_user_contact TYPE REF TO if_wd_context_element.
DATA ls_user_contact TYPE wd_this->element_user_contact.
DATA ls_bapiret2 TYPE bapiret2.
DATA lv_username TYPE wd_this->element_user_contact-username.

*   navigate from <CONTEXT> to <USER_CONTACT> via lead selection
lo_nd_user_contact = wd_context->get_child_node( name = wd_this->wdctx_user_contact ).

*   @TODO handle non existant child
*   IF lo_nd_user_contact IS INITIAL.
*   ENDIF.

*   get element via lead selection
lo_el_user_contact = lo_nd_user_contact->get_element( ).
*   @TODO handle not set lead selection
IF lo_el_user_contact IS INITIAL.
```

```

ENDIF.

*   get single attribute
lo_el_user_contact->get_attribute(
  EXPORTING
    name = `USERNAME`
  IMPORTING
    value = lv_username ).

IF lv_username IS NOT INITIAL.

  CALL FUNCTION 'BAPI_USER_EXISTENCE_CHECK'
    EXPORTING
      username = lv_username
    IMPORTING
      return    = ls_bapiret2.

  IF ls_bapiret2-id = '01' AND ls_bapiret2-
number = 124. " If the user does not exist

    DATA lo_window_manager TYPE REF TO if_wd_window_manager.
    DATA lo_api_component  TYPE REF TO if_wd_component.
    DATA lo_window         TYPE REF TO if_wd_window.
    DATA lt_buttons        TYPE wdr_popup_button_list.
    DATA ls_canc_action    TYPE wdr_popup_button_action.

    lo_api_component      = wd_comp_controller->wd_get_api( ).
    lo_window_manager     = lo_api_component->get_window_manager( ).
    *   create the cancel icon, but without any action handler
    ls_canc_action-action_name = '*'.
    *   Simple example, see docu of method create_and_open_popup for details
    lt_buttons              = lo_window_manager->get_buttons_ok(
    *   default_button      = if_wd_window=>co_button_ok
    ).

    lo_window              = lo_window_manager->create_and_open_popup(
      window_name          = 'W_POPUP'

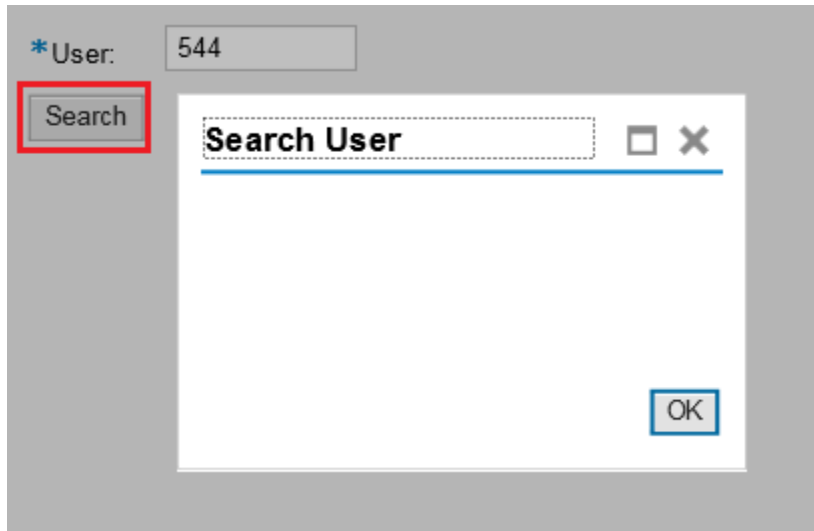
    *   title              =
      message_type          = if_wd_window=>co_msg_type_none
      message_display_mode = if_wd_window=>co_msg_display_mode_selected
    *   is_resizable        = ABAP_TRUE
      buttons               = lt_buttons
      cancel_action         = ls_canc_action
    ).

  ELSE.
    wd_this->fire_to_details_plg( ).
  ENDIF.
ELSE.
  wd_this->fire_to_details_plg( ).
ENDIF.

ENDMETHOD.

```

- e. Execute your WDA, put an invalid user and press SEARCH:



As we can see a new POPUP is appearing when the user is invalid. However this popup is very poor of information, lets improve it a little bit, so return to the ABAP code:

Click 2x in one of the marked attributes:

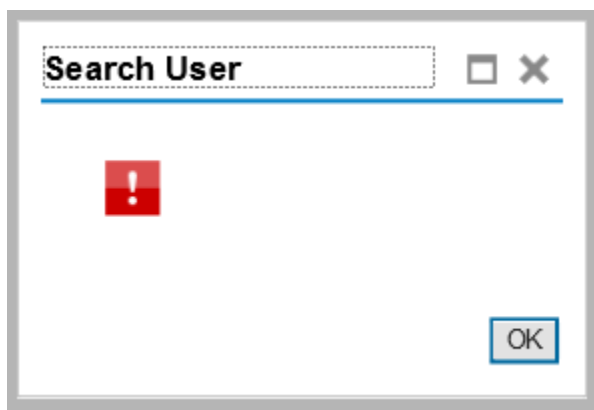
```
message_type           = if_wd_window=>co_msg_type_none
message_display_mode   = if_wd_window=>co_msg_display_mode_selected
```

Instead of using the type “none”, lets use the “error” type:

CO_MSG_TYPE_NONE	Constant	<input type="checkbox"/>	Type	WDR_POPUP_MSG_T...		Web Dynpro: Message T...	0
CO_MSG_TYPE_WARNING	Constant	<input type="checkbox"/>	Type	WDR_POPUP_MSG_T...		Warning	5
CO_MSG_TYPE_INFORMATION	Constant	<input type="checkbox"/>	Type	WDR_POPUP_MSG_T...		Information	1
CO_MSG_TYPE_QUESTION	Constant	<input type="checkbox"/>	Type	WDR_POPUP_MSG_T...		Question	2
CO_MSG_TYPE_ERROR	Constant	<input type="checkbox"/>	Type	WDR_POPUP_MSG_T...		Error	3
CO_MSG_TYPE_STOPP	Constant	<input type="checkbox"/>	Type	WDR_POPUP_MSG_T...		Cancel	4

```
message_type           = if_wd_window=>co_msg_type_error
message_display_mode   = if_wd_window=>co_msg_type_error
```

Now, your popup will look like this:



e. Changing the Title and the content:

Backing to the code, do the following change:

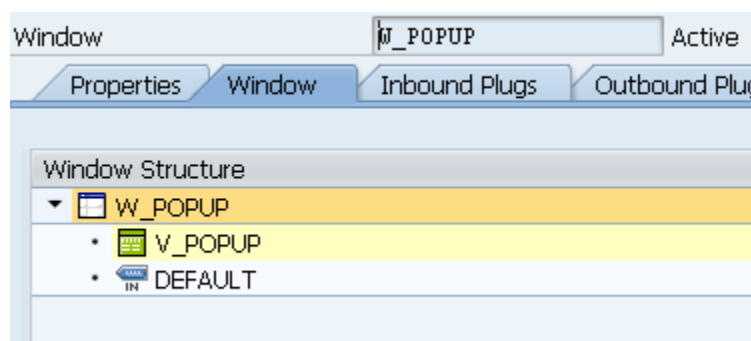
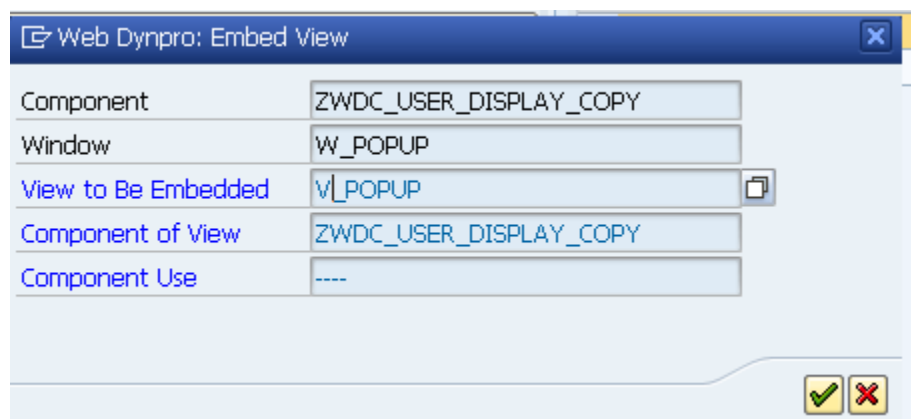
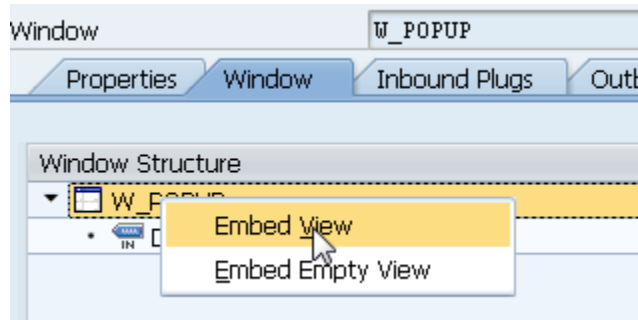
```
59 | title = 'This is the new Title'
```

Result:



Adding some content (Help procedure for the user ):

Just Embed the View V\_POPUP into the W\_POPUP window:



Final Result:

