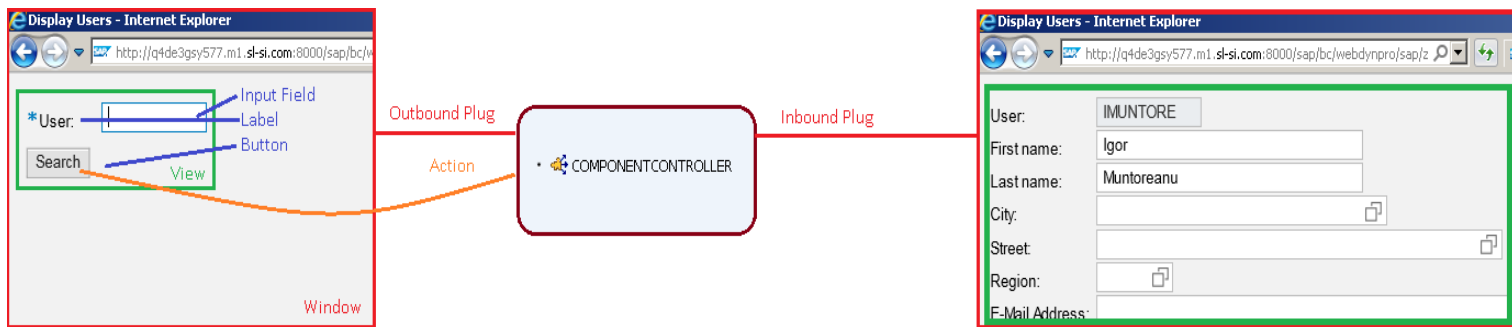Version   1.0
Date   27.06.2016
Status   Completed
Number 1
Initial version by Igor Muntoreanu

Exercise Review:

Practical Guide:

1) **Go to SE80 (that is the transaction that you are going to manipulate all the WebDynpro ABAP components) and create a WebDynpro Component:**

   When you create a WD Component, a Window and a View are required to be generated too.

2) **Manipulating the Component Controller**
   The Component Controller is always just one. It drives the functionality of the entire component. When starting a WebDynpro application, the component controller is instantiated by the WebDynpro runtime.
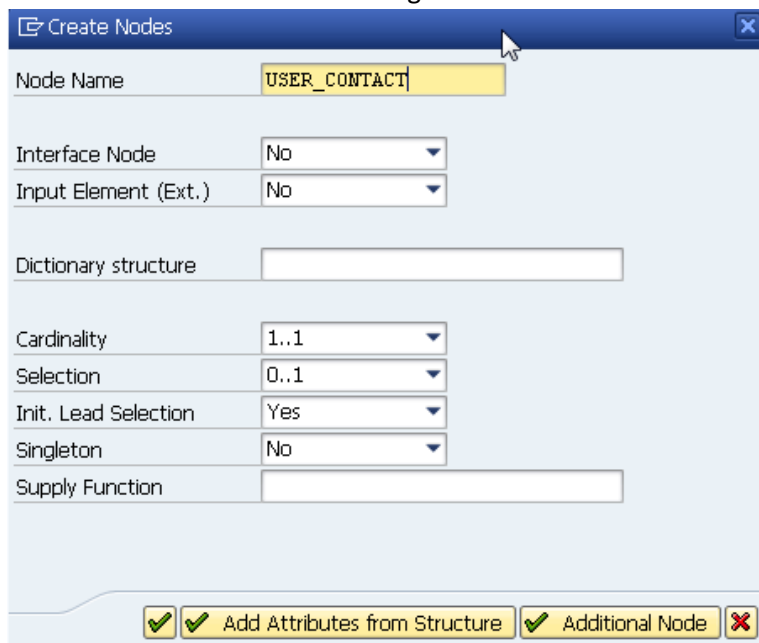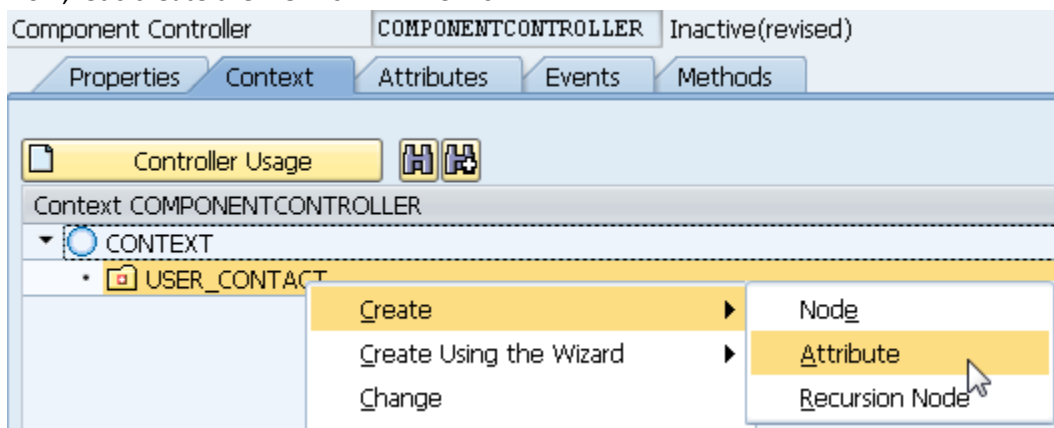
   a. Double click on the following target:
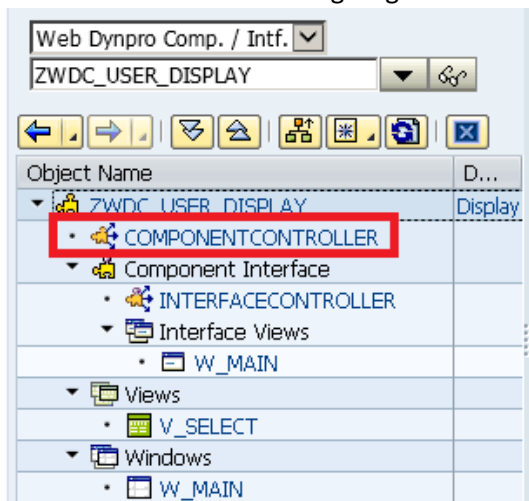
b. Inside the context of the COMPONENTCONTROLLER, tab Context, create a NODE:
    NODE is a group of Attributes all together.
    CONTEXT is the place where you defined what NODES/ATTRIBUTES the current Component is going to use.
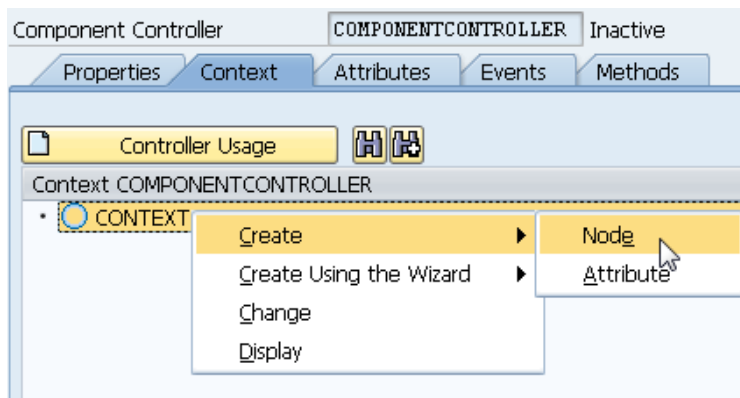
c. Create the NODE like the following Screen Shot:

d. Now, let's create the NODEs ATTRIBUTEs:
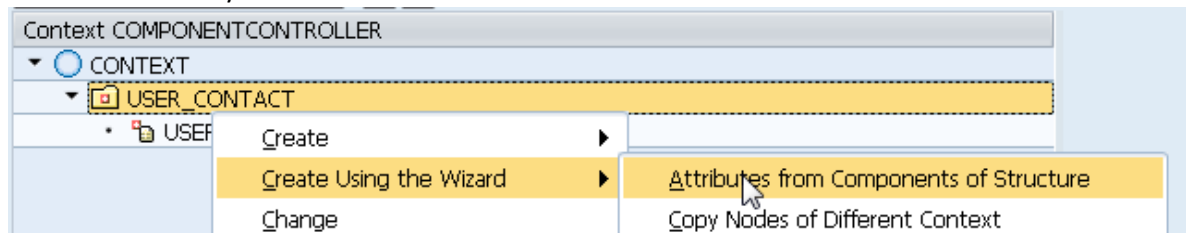
You can create one attribute at time. In this case, let´s use the WIZARD to bring all the attributes we want automatically:
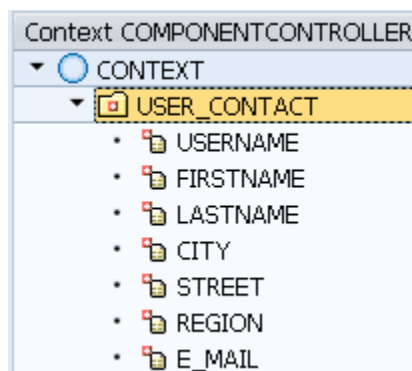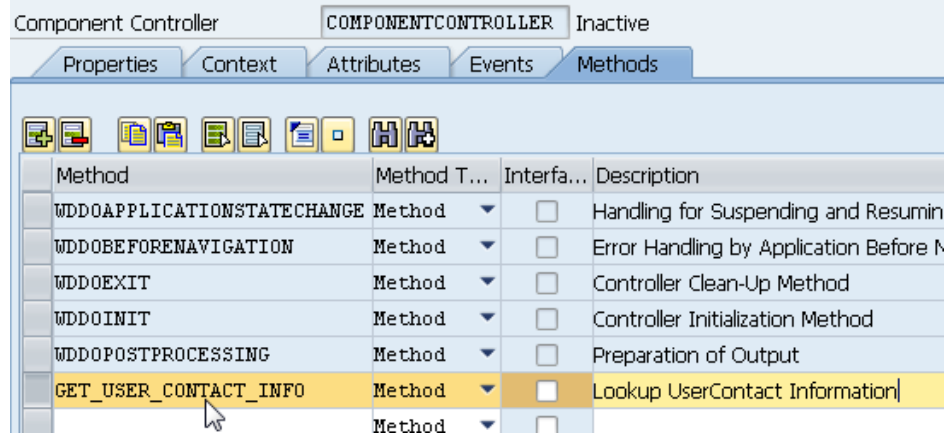

Select the fields and bring the following ATTRIBUTES:

e. Create a Lookup Method in the "Methods" tab and implement the following code (Do not pay attention to the code now, just copy and paste for while):
The methods that you can create here can be called anywhere in the next Views.

Component Controller	COMPONENTCONTROLLER	Inactive

Properties | Context | Attributes | Events | Methods

Method	Method T...	Interfa...	Description
WDDOAPPLICATIONSTATECHANGE	Method ▼	☐	Handling for Suspending and Resumin
WDDOBEFORENAVIGATION	Method ▼	☐	Error Handling by Application Before N
WDDOEXIT	Method ▼	☐	Controller Clean-Up Method
WDDOINIT	Method ▼	☐	Controller Initialization Method
WDDOPOSTPROCESSING	Method ▼	☐	Preparation of Output
GET_USER_CONTACT_INFO	Method ▼	☐	Lookup UserContact Information|
	Method ▼	☐	

```
DATA: lo_nd_user_contact TYPE REF TO if_wd_context_node,
      lo_el_user_contact TYPE REF TO if_wd_context_element.

DATA: lv_username TYPE REF TO wd_this->element_user_contact-username.

DATA: ls_address TYPE bapiaddr3,
      lt_return  TYPE TABLE OF bapiret2.

DATA: ls_user_contact TYPE wd_this->element_user_contact.

" Read the selected user name into context:
lo_nd_user_contact = wd_context->get_child_node( name = wd_this-
>wdctx_user_contact ).
lo_el_user_contact = lo_nd_user_contact->get_element( ).

lv_username ?= lo_el_user_contact->get_attribute_ref('USERNAME').

" Lookup the user's contact information:
CALL FUNCTION 'BAPI_USER_GET_DETAIL'
  EXPORTING
    username = lv_username->*
  IMPORTING
    address  = ls_address
  TABLES
    return   = lt_return.

" Copy the results to a structure and bind it to the context:

ls_user_contact-username       = lv_username->*.
ls_user_contact-firstname      = ls_address-firstname.
ls_user_contact-lastname       = ls_address-lastname.
ls_user_contact-street         = ls_address-street.
ls_user_contact-city           = ls_address-city.
ls_user_contact-region         = ls_address-region.
ls_user_contact-e_mail          = ls_address-e_mail.

" Bind it to the context
lo_el_user_contact->set_static_attributes( ls_user_contact ).
```
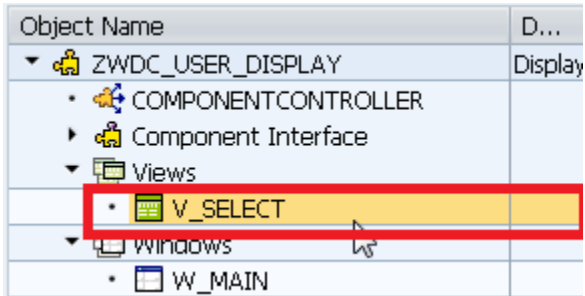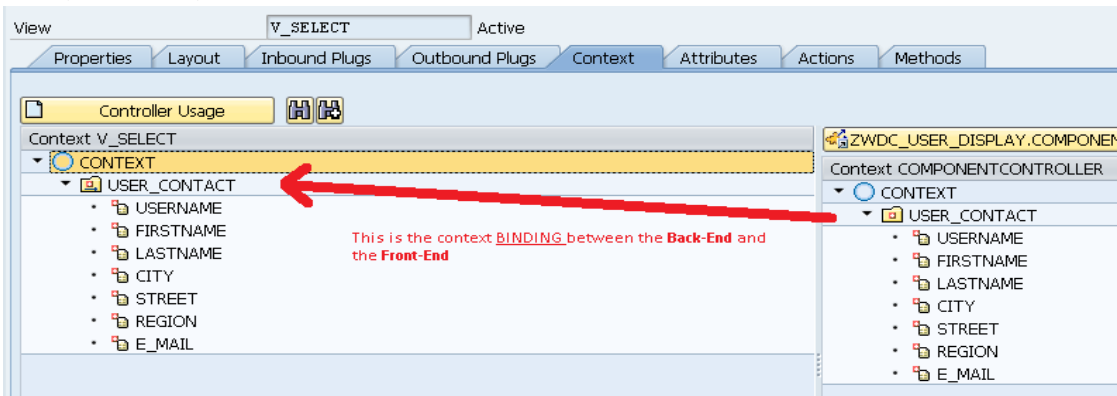
3) Laying out the Selection View – Front End:

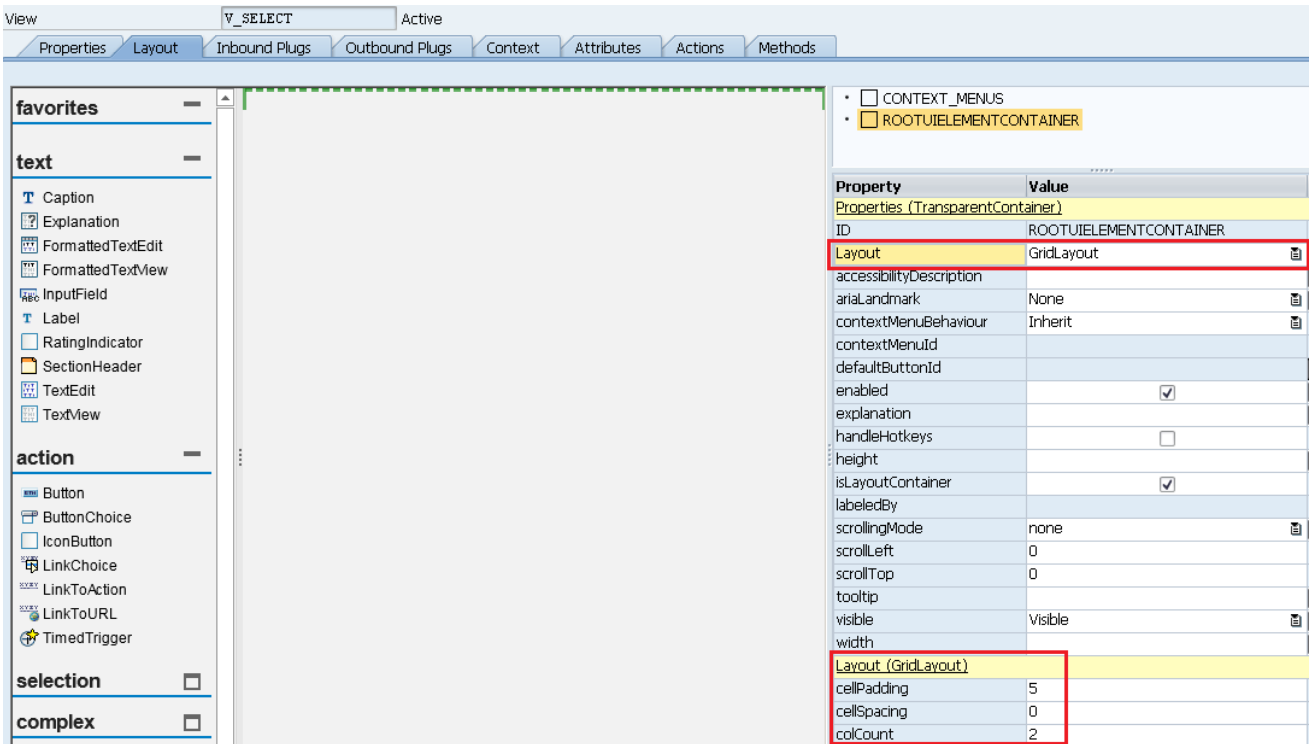The View is the place where you create the Front End Webpage.

a. Double Click on the V_SELECT view and go to the "Context" tab:

b. Drag and drop the NODE from the COMPONENTCONTROLLER(on the right) to the Context of the View(on the left).
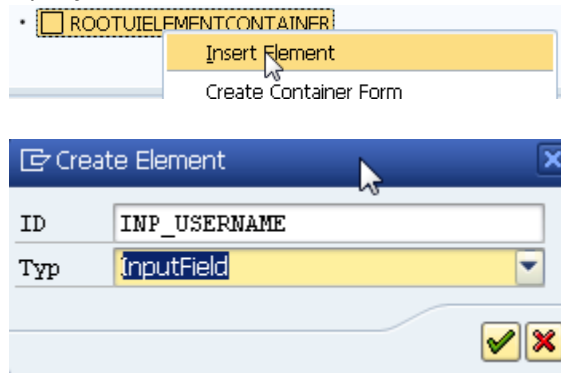

c. Go to the "Layout" tab, set the ROOTUIELEMENTCONTAINER as displayed below:

In the "Layout" tab you will be able to define all the Webpage Screen Elements and their layouts.
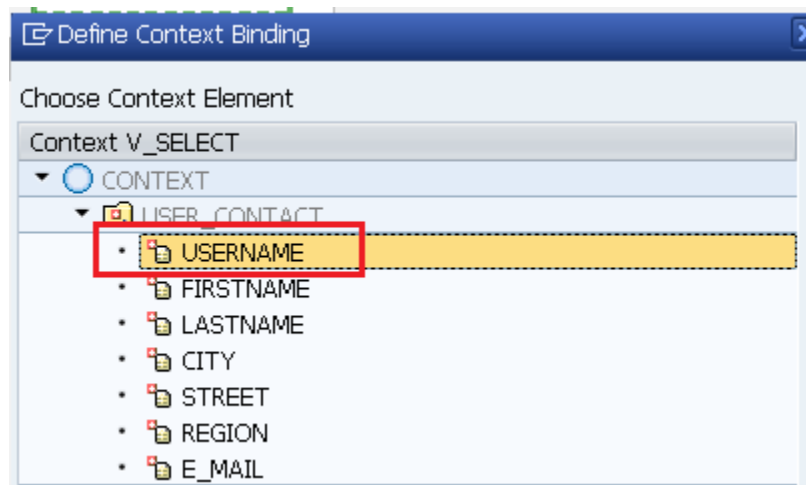
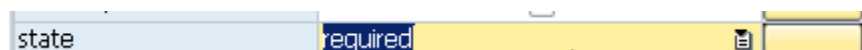So, let's create an Input field, a Label and a Button:

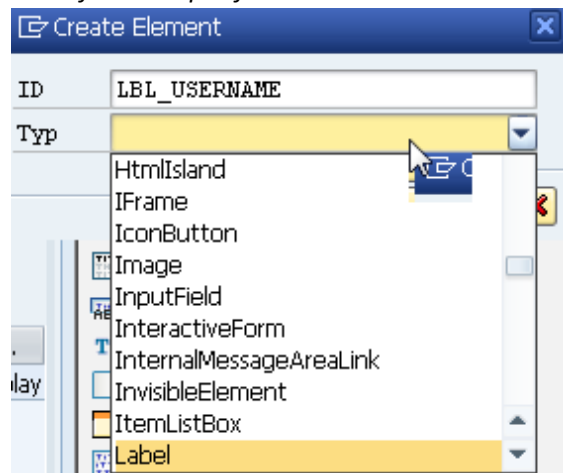- *Input field:*

Bind the "value" Property:

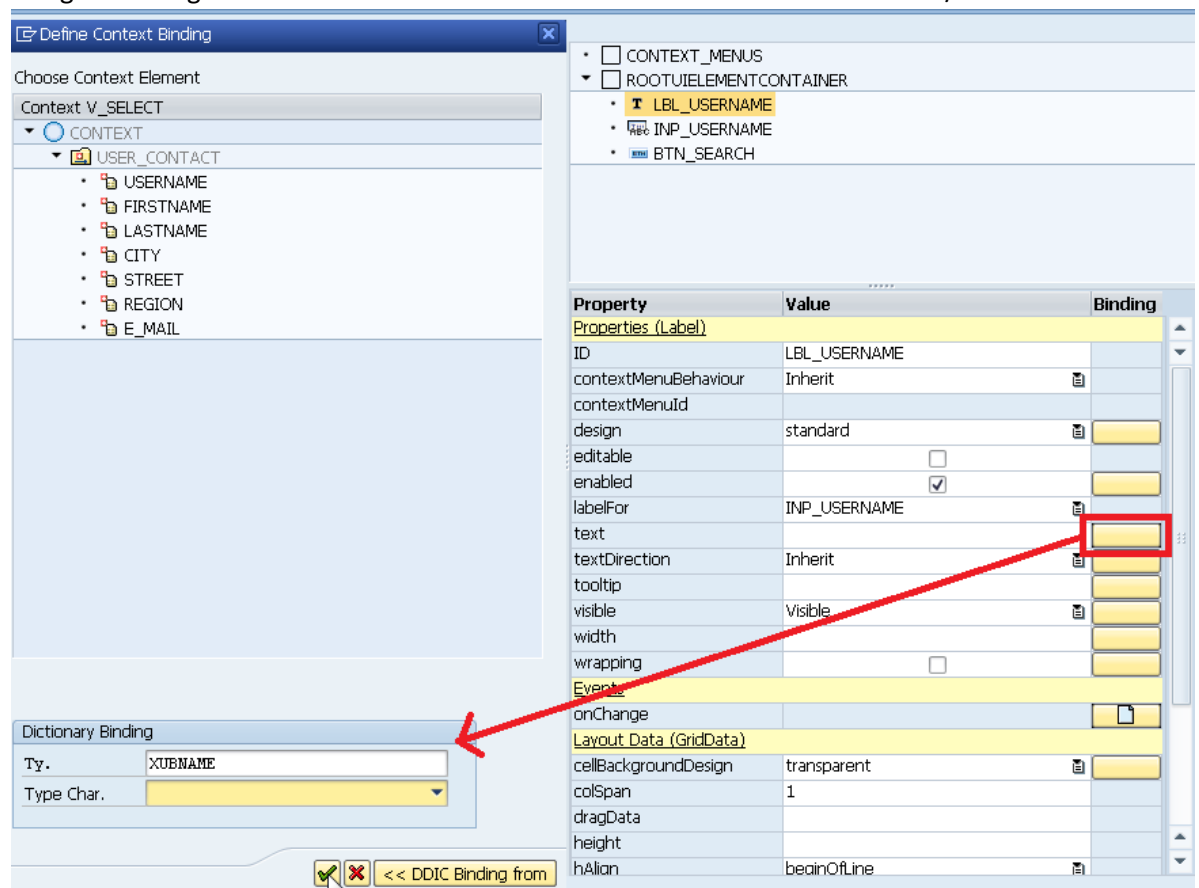Choose the USERNAME attribute from the Context Element

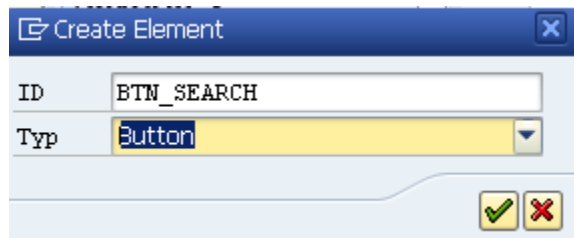Make the field required:

- *Label for the Input field:*

Links the Label with the Input field in the property "labelFor". Give a text to the label linking it by using the Biding with a Data Element. It will take the text direct from the Table/Structure:
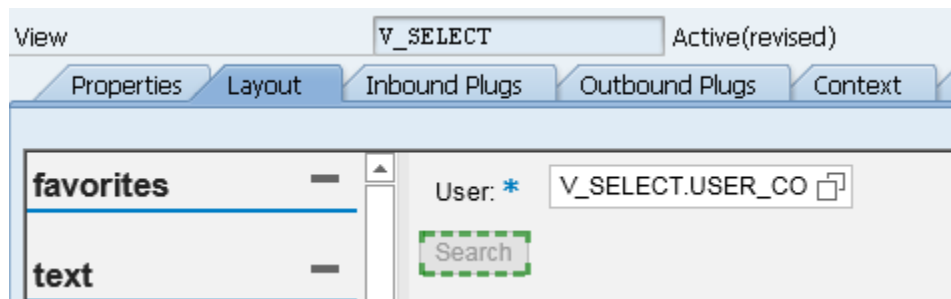
- *Button:*

Give a text to the Button

The final result of the view layout should be like this:
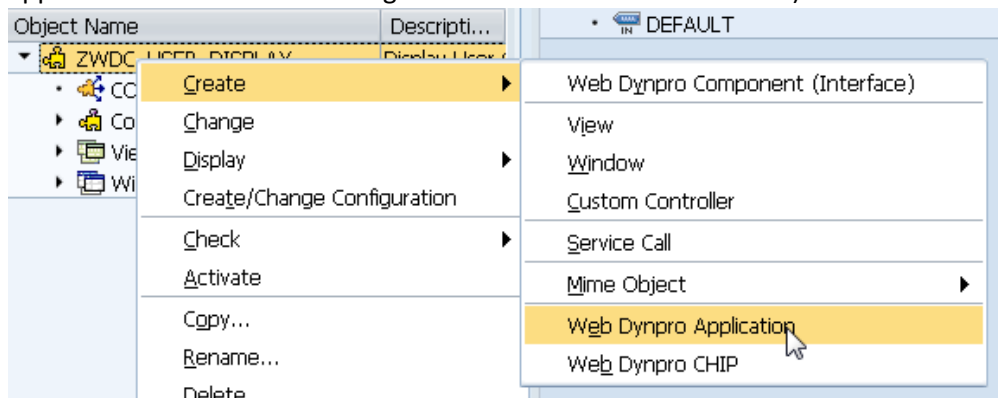
4) **Create the WebDynpro Application**

Analogously, it is like generating a transaction for a Report.

a. Create the WebDynpro Application like showed below (it generates automatically an URL for the application if all the BASIS configuration in SCIF transaction are ok).

Make sure to Save and Activate your development times in times:

: Create Web Dynpro Application

Application   ZWDC_USER_DISPLAY   New

Properties | Parameters

Description   Display Users
Component   ZWDC_USER_DISPLAY
Interface View   W_MAIN
Plug Name   DEFAULT
Help Links

Authorization Check
◉ Check for Application
◯ Check for Application and Application Configuration

Handling of Messages
◉ Show Message Component on Demand
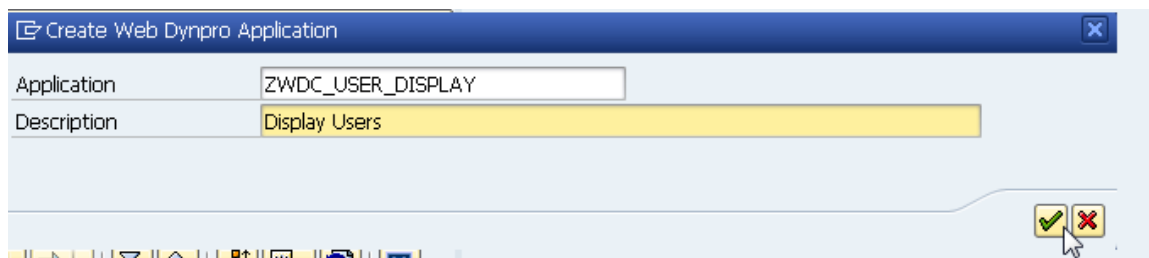◯ Always Display Message Component

Administrative Data

Created By	IMUNTORE	Created on	27.06.2016
Last Changed By		Changed On	
Package			
Language			
URL	http://q4de3gsy577.ml.sl-si.com:8000/sap/bc/webdynpro/sap/zwdc_user_d…		

Descripti...
Display User
ER

Create Administration Service

☑ Administration Service   🛈

✔ ✖

Run the WDA (WebDynpro Application):

Web Dynpro Explorer: Change Web Dynpro Application

← → | ✎ 🗗 | 🔀 ⇄ 🖳 | 🔀 🖥 📧 🛈

The webpage will open:

Display Users - Internet Explorer

← → ▾ 🔍 http://q4de3gsy577.m1.sl-si.com:8000/sap/bc/webdynpro/sap/z 🔍 ▾ ↻ | Display Users   ✕

*User:

Search

As you can see that is no functionality yet. The next steps will create them.

5) **Laying out the Details View – Front End:**
   a. Right click on "Views" and create a new one like showed below:
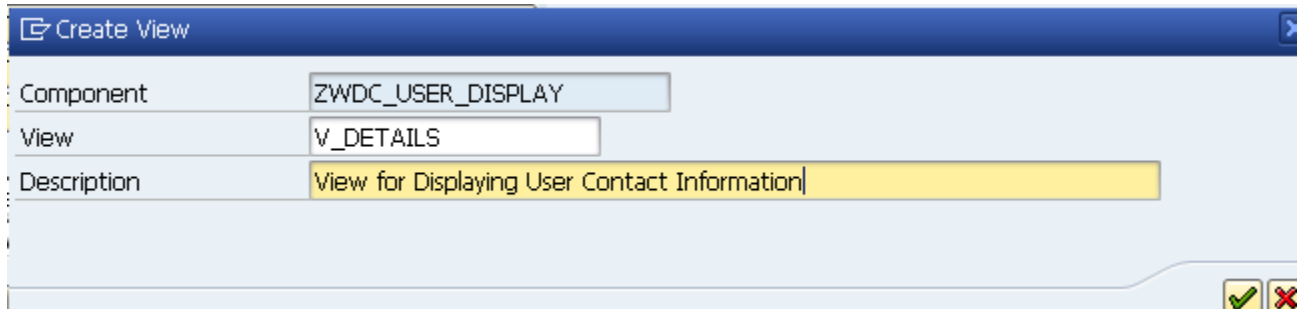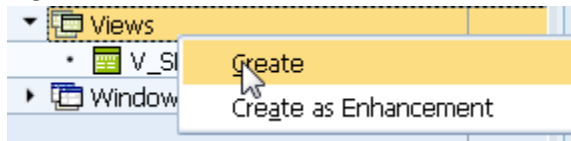
   b. Drag and Drop the context for this view also, just like we did in the V_SELECT view:
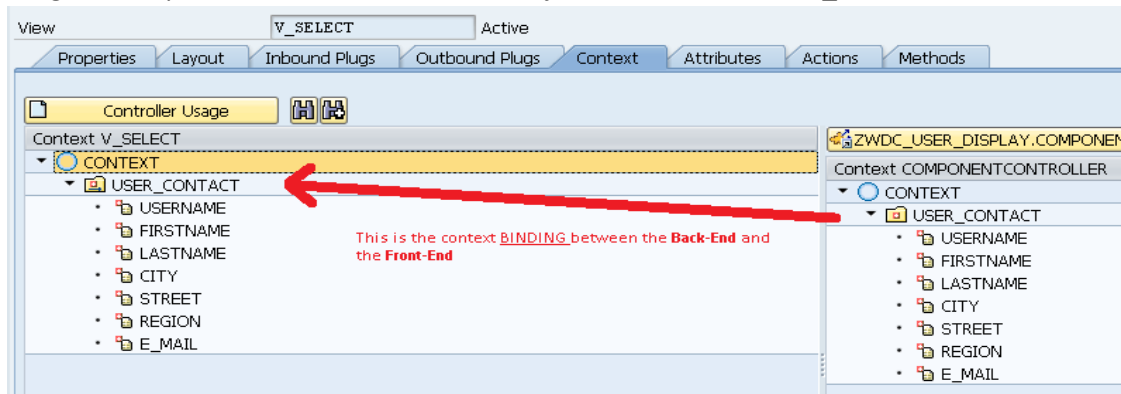
   c. Inside the view "V_DETAILS", create a "Container Form" in the layout tab:

Set the Input Field "USERNAME" as read-only:

Set property "isLayoutContainer" from FORM_CONTAINER marked:

6) Outbound Plug in the V_SELECT View (Plugs are nothing more than Navigation specification between the Views).

    a. Open the V_SELECT View and in the tab Outbound Plugs and create a Plug like below:

View	V_SELECT	Active

Properties | Layout | Inbound Plugs | Outbound Plugs | Con

Outbound Plugs

Plug Name	Description
TO_DETAILS	

    b. Now, open the Inbound Plugs and create like below:

View	V_SELECT	Active(revised)

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

Inbound Plugs

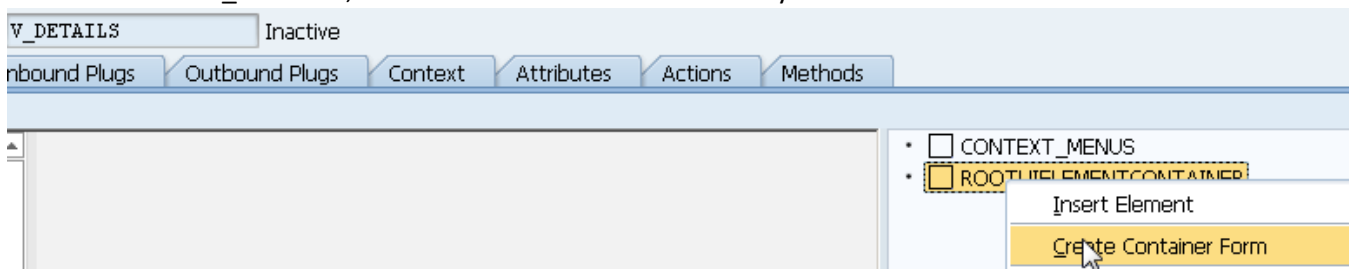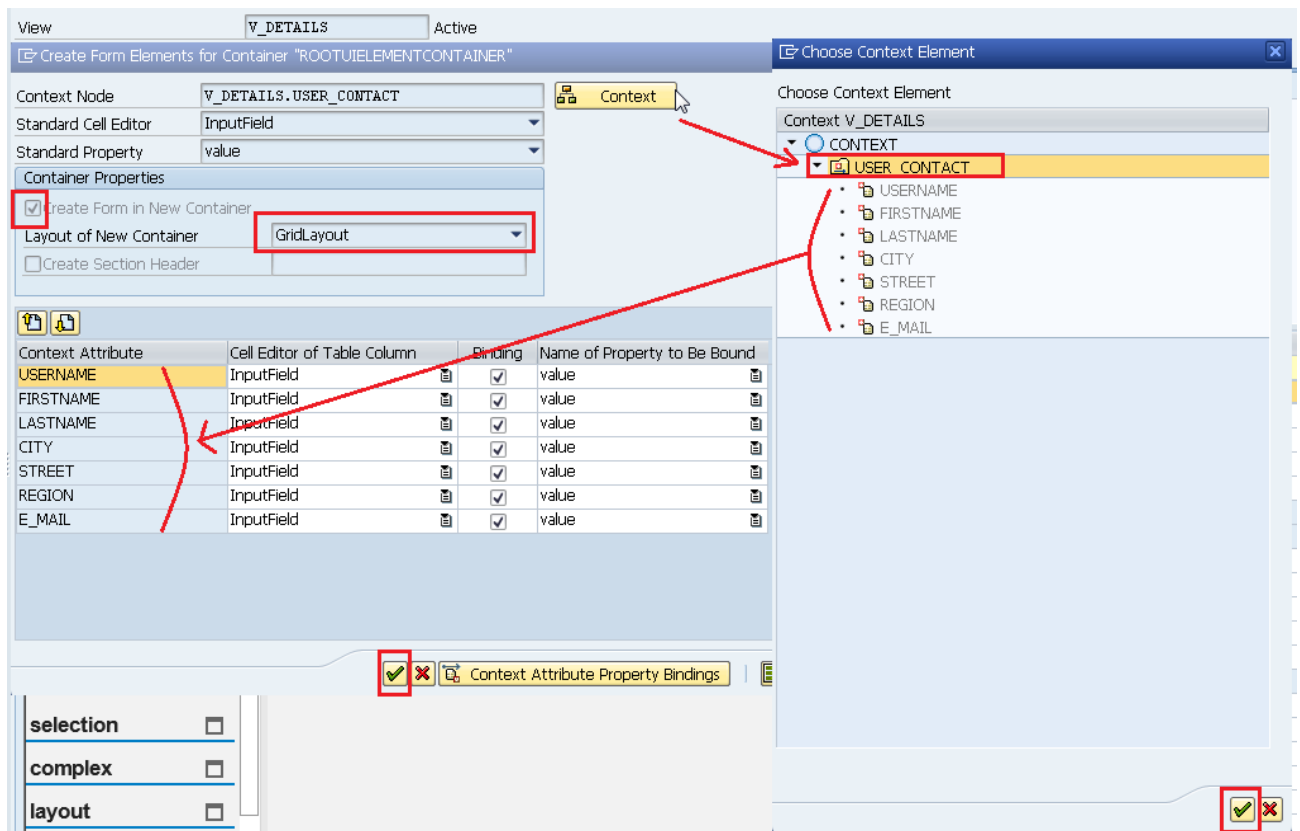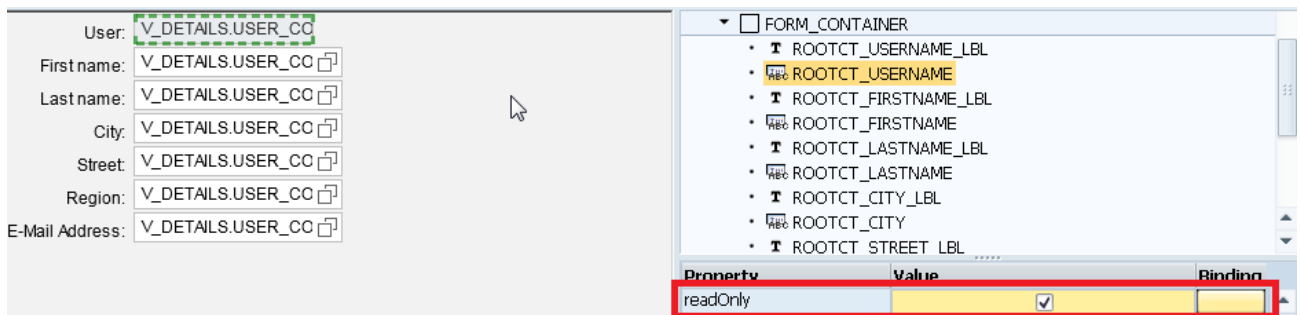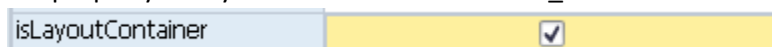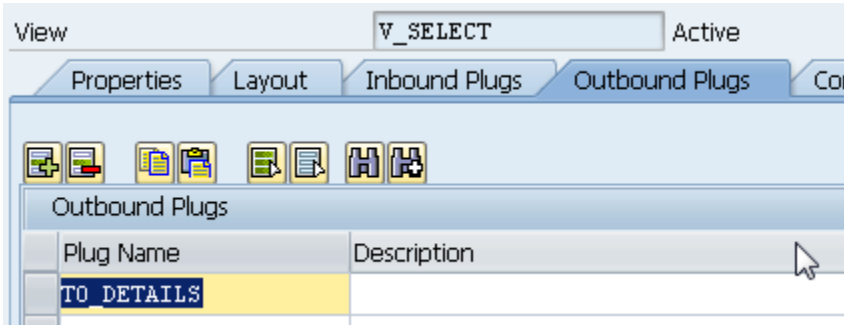| Plug Name	Description	Event Handler
FROM_DETAILS		HANDLEFROM_DETAILS
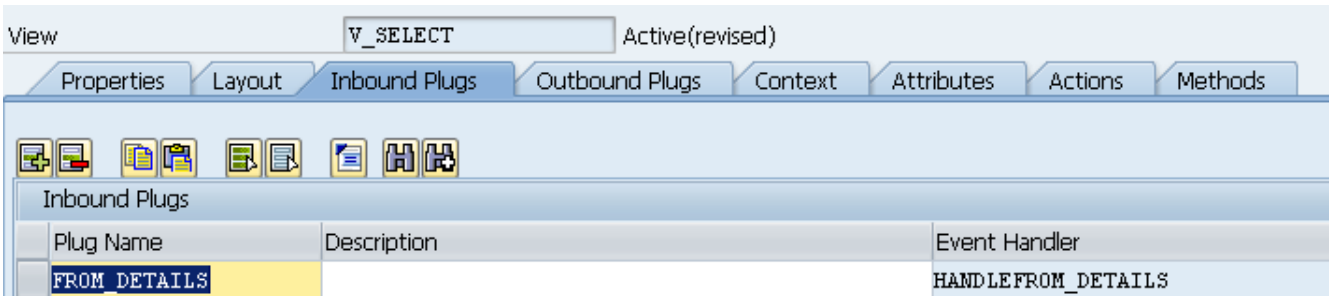
7) Inbound Plug in the V_DETAILS View

    a. Open the V_DETAILS View and in the tab Outbound Plugs create a plug like below:

View	V_DETAILS	Active

Properties | Layout | Inbound Plugs | Outbound Plugs | Context | Attributes | Actions

Outbound Plugs

Plug Name	Description
TO_SELECT	

    b. Now, open the Inbound Plugs and create like below:

View	V_DETAILS	Active(revised)

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

Inbound Plugs

Plug Name	Description	Event Handler
FROM_SELECT		HANDLEFROM_SELECT
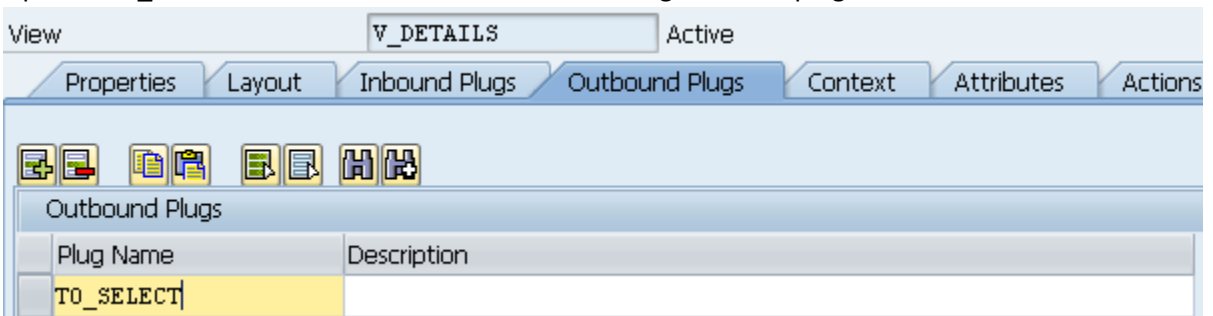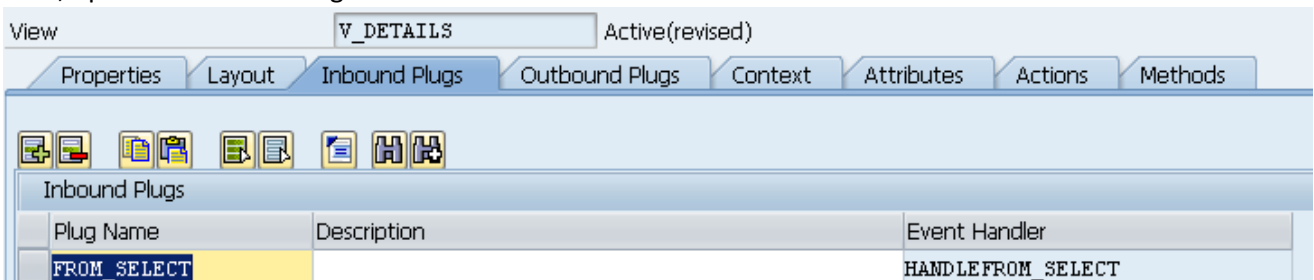
8) Window Manipulation

A Windows is different from the view, take a look on the following SS:
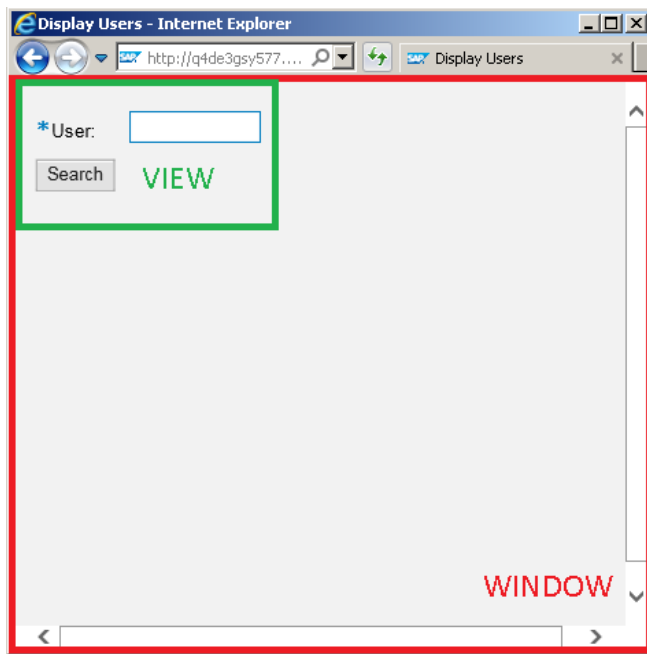
The Window can have a lot of Views together.

   a. Go to the Main Window:
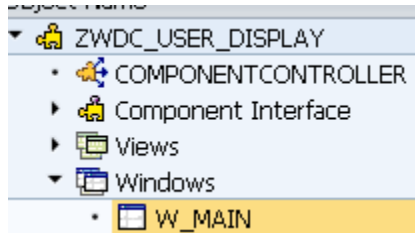
   b. Here we need to create the relations between the Views via the Inboud/Outbound Plugs:
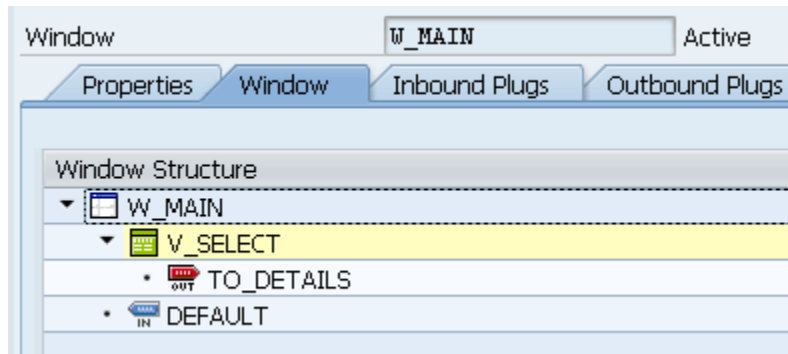
Right click on W_MAIN and choose "Embed View":
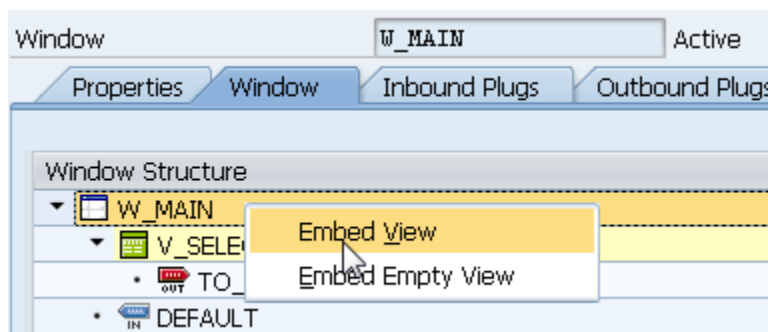
Select the V_DISPLAY view to be Embedded like the SS below:
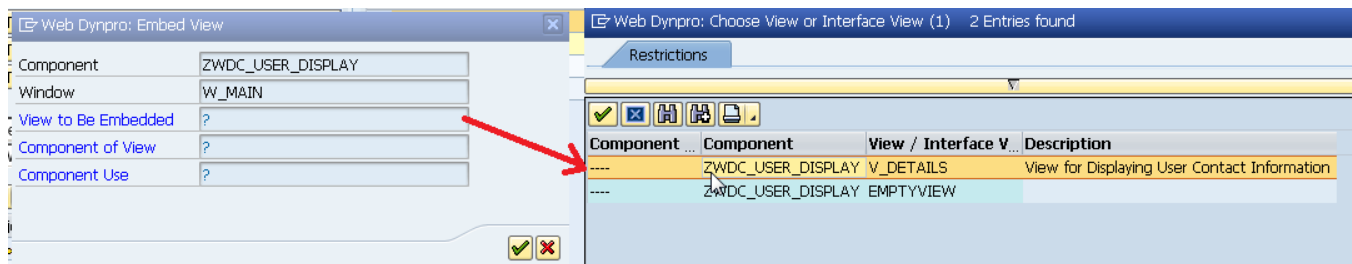
Your Window configuration will be like this:

Now its necessary to create the link between the Plugs, by creating a Navigation Link:

9) Create an ACTION to the SEARCH Button.

    a. Enter in the V_SELECT View and Click 2x In the SEARCH Button:

Create a new Action for the button:

Events	
onAction	
Layout Data (GridData)	

Create Action

Component	ZWDC_USER_DISPLAY
View	V_SELECT
Action	SEARCH
Description	Search User

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

Outbound Plug	

Plug Name (1)   1 Entry found

Restrictions

Plug Name	Descripti...
TO_DETAILS	

Now enter in the Actions tab and click twice in the new action created:

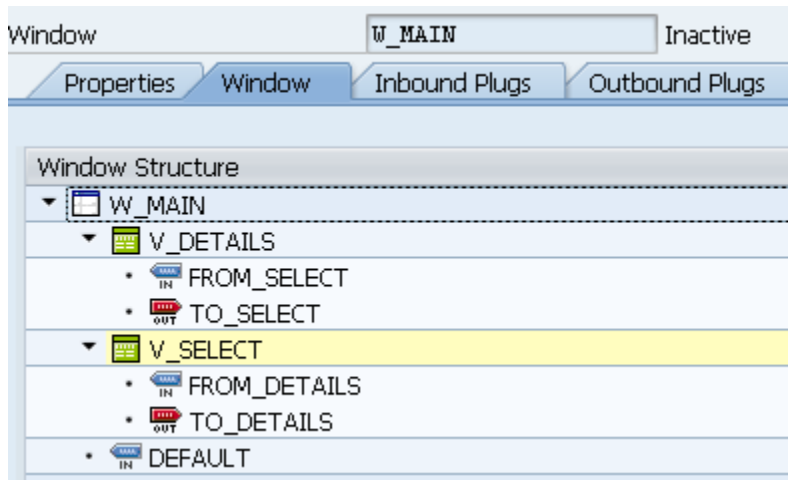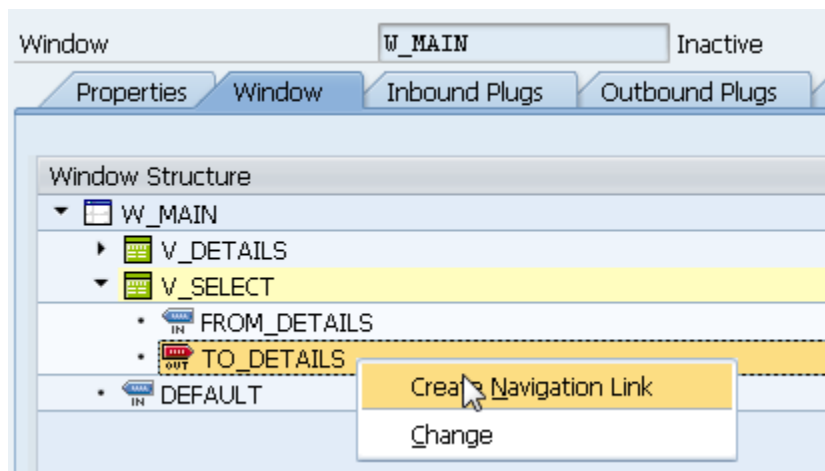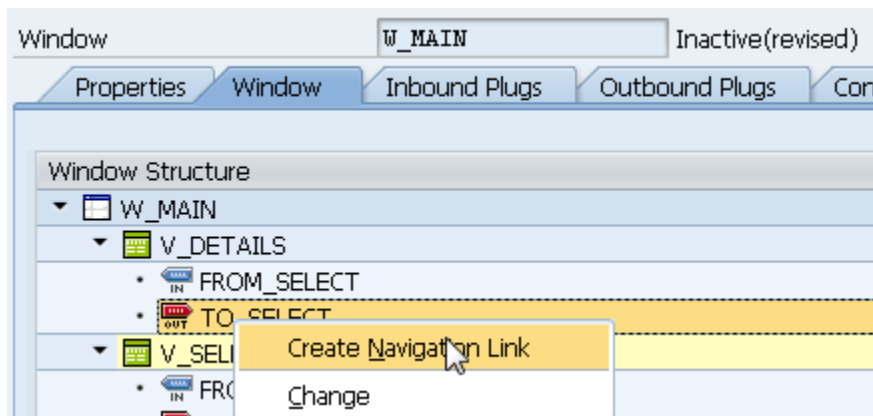| View	V_SELECT	Inactive

Properties | Layout | Inbound Plugs | Outbound Plugs | Context | Attributes | **Actions**

Action	Action Type	Description
SEARCH	Standard ▼	Search User
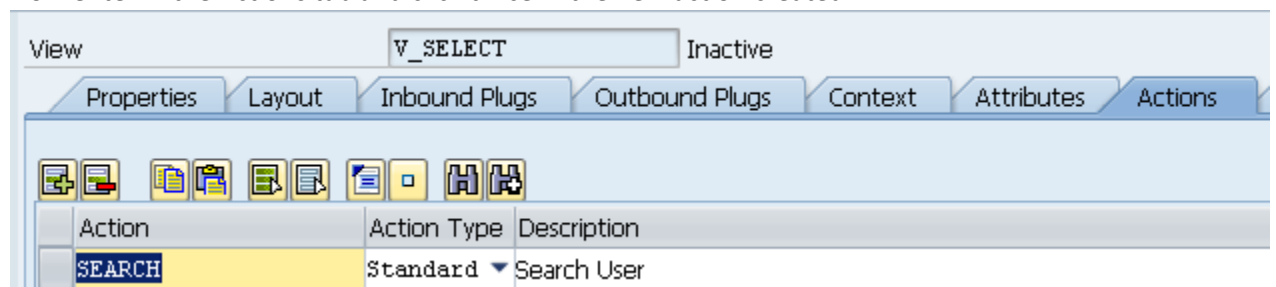
The coding brings automatically:

```
method ONACTIONSEARCH .
    wd_this->fire_to_details_plg(
    ).
endmethod.
```

10) Coding in the View V_DETAILS under Inbound Plug
   a. We will call the created method GET_USER_CONTACT_INFO

View	V_DETAILS	Active

Properties | Layout | **Inbound Plugs** | Outbound Plugs | Context | Attributes | Actions | Methods

Inbound Plugs

Plug Name	Description	Event Handler
FROM_SELECT		HANDLEFROM_SELECT
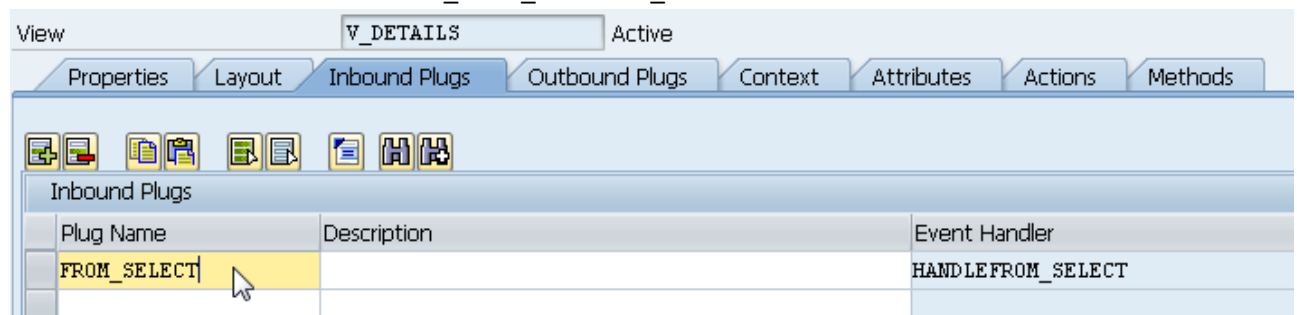
```
1   ⊟ METHOD handlefrom_select .
2
3        " Invoke the component controller method to load the user contact info:
4
5        wd_comp_controller->get_user_contact_info( ).
6
7   └ ENDMETHOD .
```

11) Using a Pre-Defined Method to promotes validations.

    a. In our case despite of the field USERNAME of the view V_SELECT is mandatory, it is necessary to create a validation for that differently from common Reports for example.

    b. Open the V_SELECT view and select WDDOBEFOREACTION:

View	V_SELECT	Active					
Properties	Layout	Inbound Plugs	Outbound Plugs	Context	Attributes	Actions	Methods

Method	Method T...	Description	Event	Controller
HANDLEFROM_DETAILS	Event H... ▼			
ONACTIONSEARCH	Event H... ▼	Search User		
WDDOAFTERACTION	Method ▼	Method for non-action specific operations b…		
WDDOBEFOREACTION	Method ▼	Method for Validation of User Input		
WDDOEXIT	Method ▼	Controller Clean-up Method		

    c. As you can see a pre-code is already defined. We will take it:

```
1    ⊟ method WDDOBEFOREACTION .
2    ⊟ *   data lo_api_controller type ref to if_wd_view_controller.
3      *   data lo_action          type ref to if_wd_action.
4
5    ⊟ *   lo_api_controller = wd_this->wd_get_api( ).
6      *   lo_action = lo_api_controller->get_current_action( ).
7
8    ⊟ *   if lo_action is bound.
9      *     case lo_action->name.
10     *       when '...'.
11
12   ⊟ *     endcase.
13     *   endif.
14   └ endmethod.
```

d. The new code will be like that:

```
View                          V_SELECT            Active
    Properties    Layout    Inbound Plugs    Outbound Plugs    Context    Attributes    Actions    Methods

  Method List        [icon]        Method        ▲ ▼
Method            WDDOBEFOREACTION


  1  ⊟ METHOD wddobeforeaction .
  2       DATA lo_api_controller TYPE REF TO if_wd_view_controller.
  3       DATA lo_action          TYPE REF TO if_wd_action.
  4
  5       lo_api_controller = wd_this->wd_get_api( ).
  6       lo_action = lo_api_controller->get_current_action( ).
  7
  8  ⊟    IF lo_action IS BOUND.
  9  ⊟      CASE lo_action->name.
 10  ○        WHEN 'SEARCH'.
 11             cl_wd_dynamic_tool=>check_mandatory_attr_on_view(
 12               EXPORTING view_controller = lo_api_controller ).
 13           ENDCASE.
 14         ENDIF.
 15    └ ENDMETHOD .
```
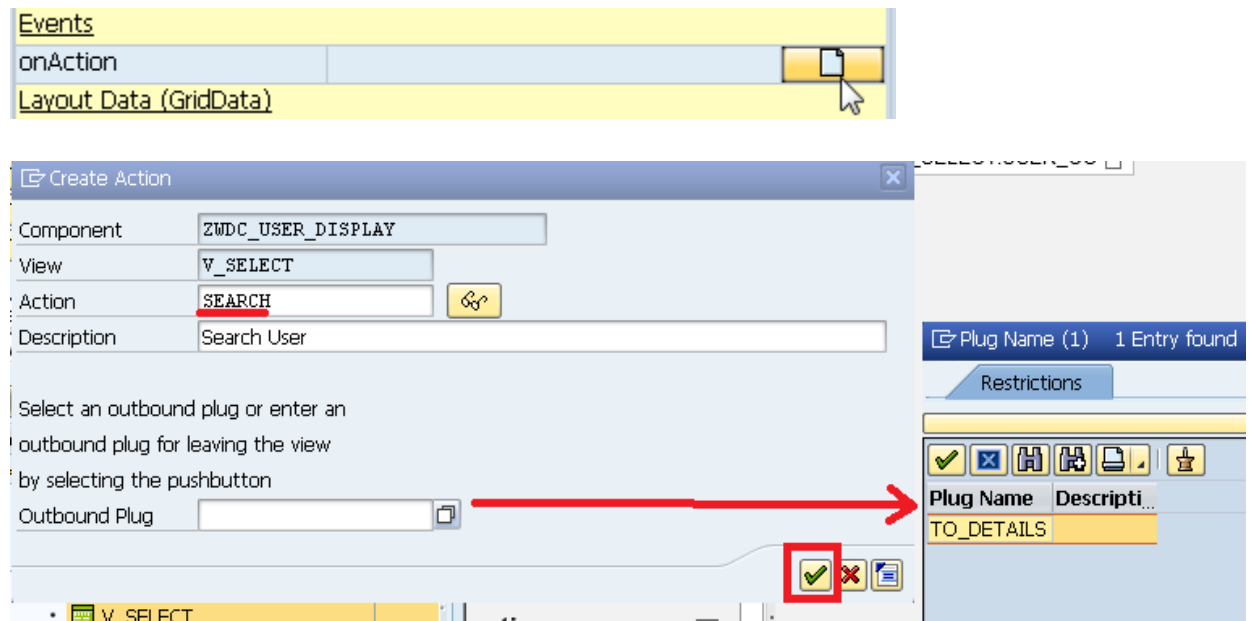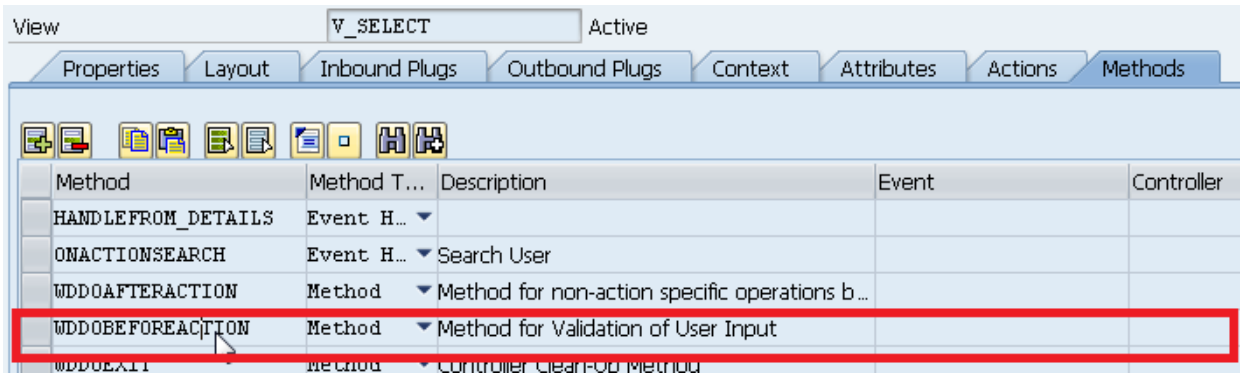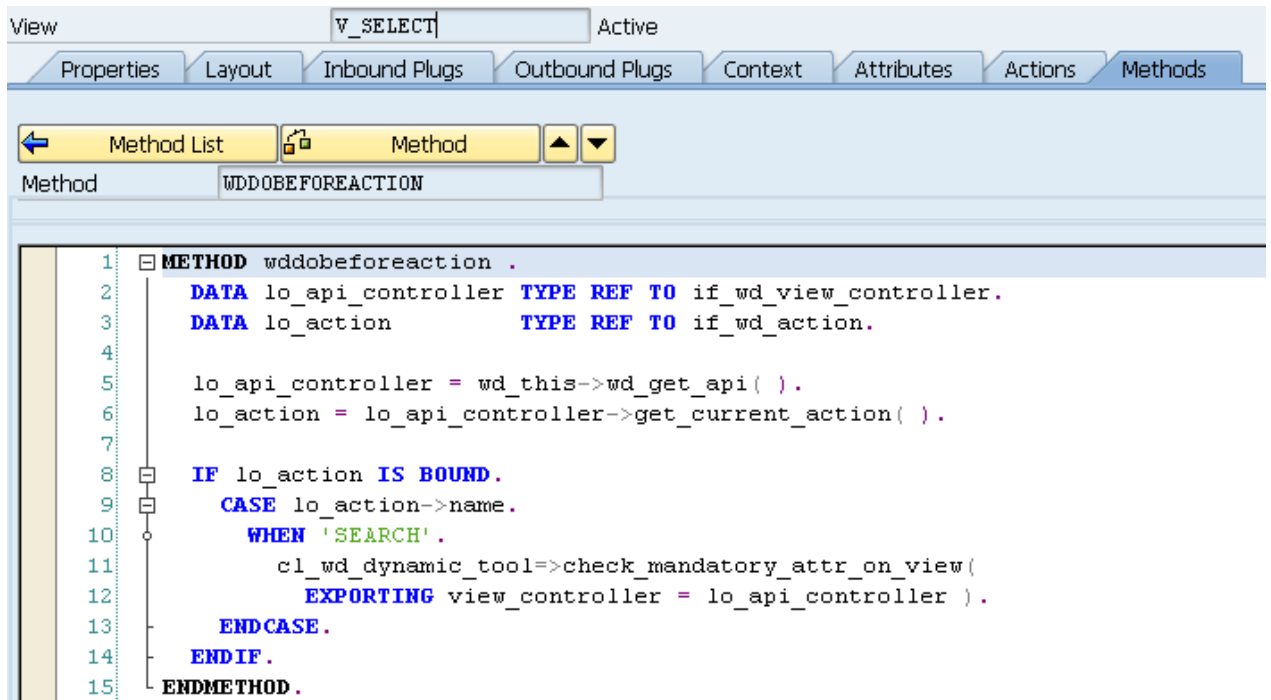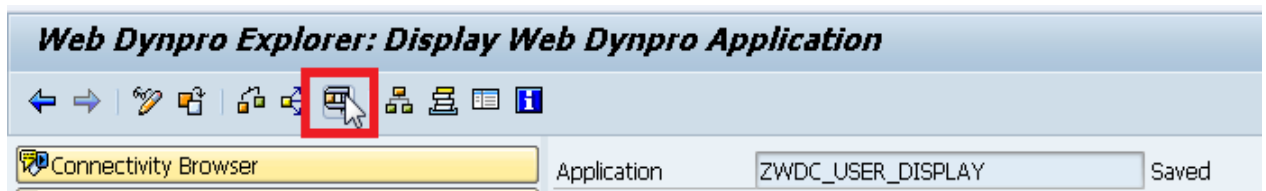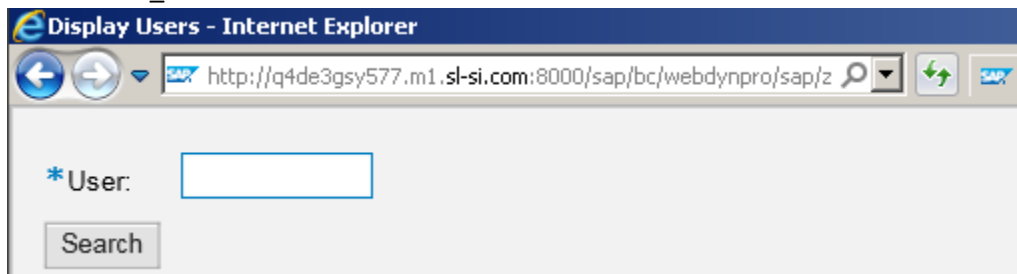
12) The WebDynpro is finally done, let's check it out:
   a. Go to the WDA and run it.

Web Dynpro Explorer: Display Web Dynpro Application

| Connectivity Browser | Application | ZWDC_USER_DISPLAY | Saved |

   b. This is the V_SELECT view

Display Users - Internet Explorer

http://q4de3gsy577.m1.sl-si.com:8000/sap/bc/webdynpro/sap/z

*User:

Search

Click on Search without typing the user:

⚠ Fill all required entry fields

*User:

Search

Now put some valid value and execute:

This is the V_DETAILS view.

13) Debugging Tool Tips:
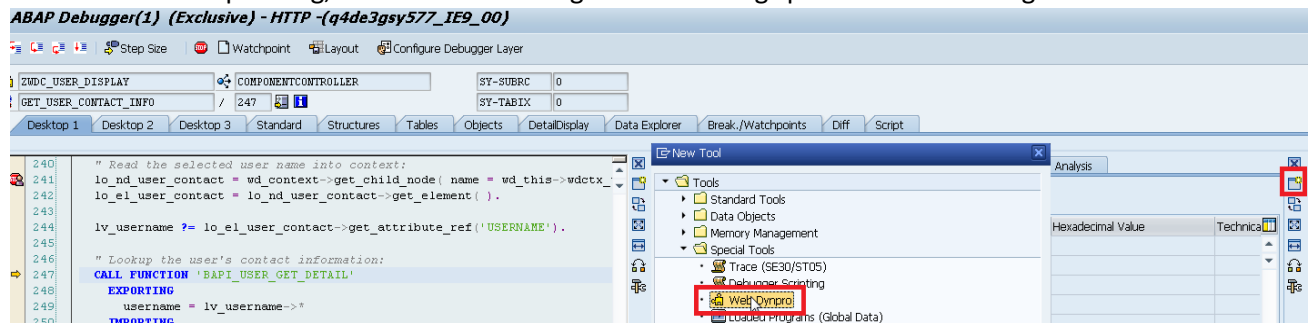   a. Put an external break-point in the method WDDOBEFOREACTION
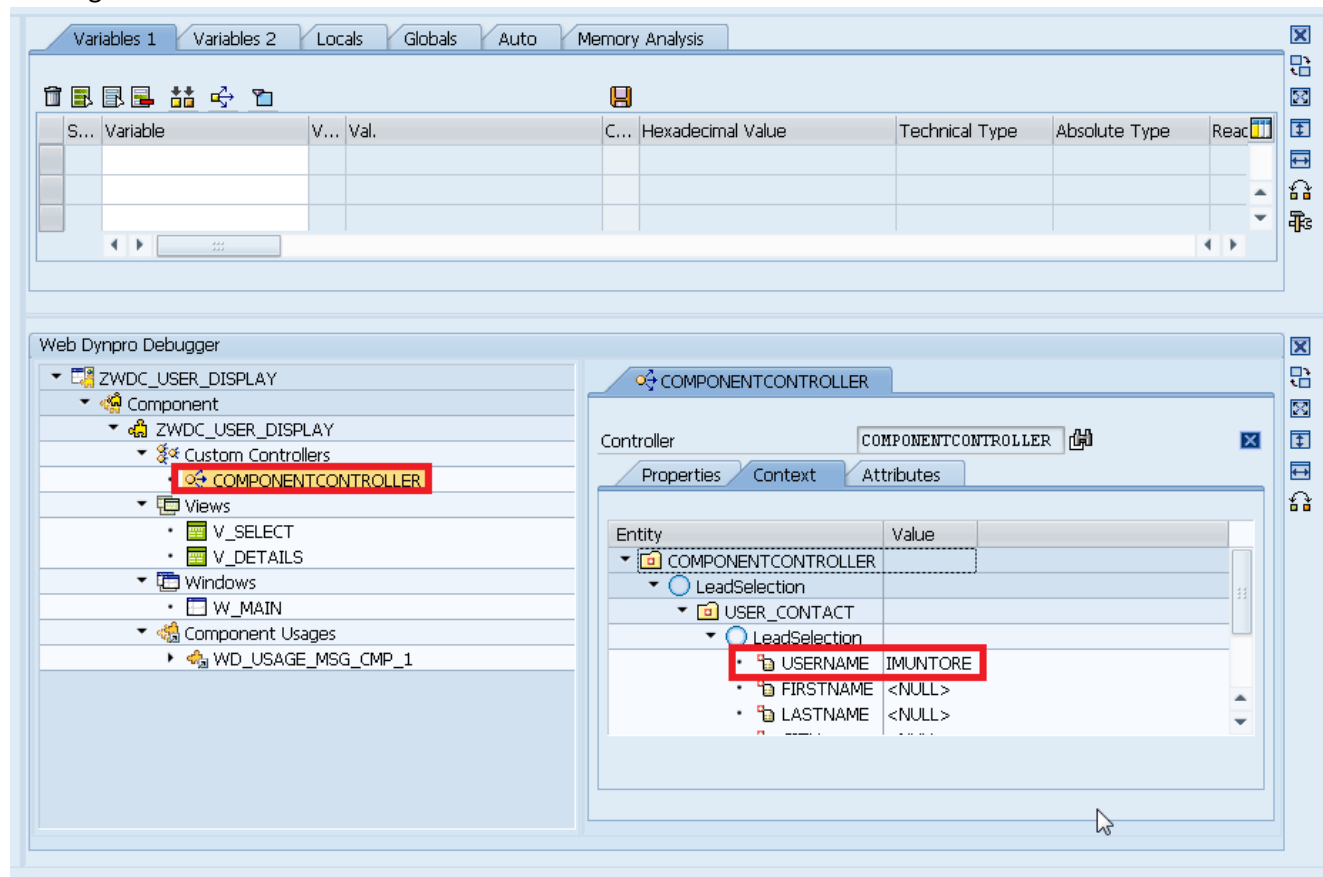

   b. Now, execute the WDA:

c. The debug opens automatically:

Go ahead with F6 pressing, until the BAPI calling. Call the debug special tool for Debug:

Your screen will be like the SS showed below. It is a very powerful that helps a lot during problems investigation:

14) Homework:
- a. Create a button in the V_DETAILS to return to the V_SELECT view.
- b. Create a validation if the entry data is valid. If not display a POP UP and freeze in the V_SELECT view.