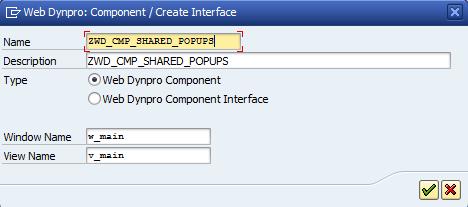
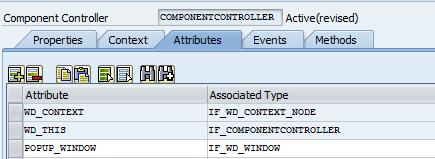
**Part 1:**

**Step 1:  Create an ABAP WebDynpro component with a method to share with other WD apps.**

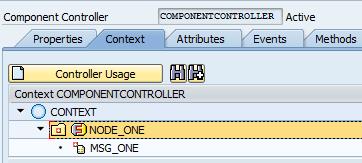
ABAP DynPro Name: ZWD\_CMP\_SHARED\_POPUPS

[](https://blogs.sap.com/wp-content/uploads/2013/04/p_205924.jpg)

**Step 2:  Create an attribute of the COMPONENTCONTROLLER called POPUP\_WINDOW.**

***[](https://blogs.sap.com/wp-content/uploads/2013/04/p_206024.jpg)***

Then create a context node and attribute on the component controller context as follows:

***[](https://blogs.sap.com/wp-content/uploads/2013/04/p_206025.jpg)***

\*Make sure to select the check box “Interface Node” on the node attributes:

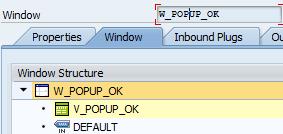
***[](https://blogs.sap.com/wp-content/uploads/2013/04/p_206026.jpg)***

\*Select STRING for the type of attribute for MSG\_ONE

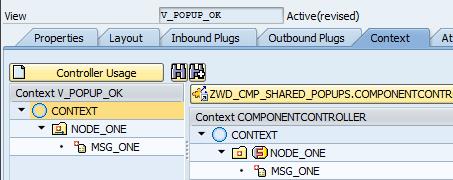
**Step 3:  Create a new view called V\_POPUP\_OK and a new window called W\_POPUP\_OK.**

Set Width (400px) and Height (300px) properties on the ROOTUIELEMENTCONTAINER of the View

Then embed the View onto the Window:

[](https://blogs.sap.com/wp-content/uploads/2013/04/p_206027.jpg)

On the views context, drag (map) the node “NODE\_ONE” from the component controller context to the view context:

[](https://blogs.sap.com/wp-content/uploads/2013/04/p_206028.jpg)

On the layout of this view, create a TextView element and bind its text to the MSG\_ONE attribute:

[](https://blogs.sap.com/wp-content/uploads/2013/04/p_206029.jpg)

**Step 4:  Create a method called POPUP\_OK (check as Interface) on the Component Controller.**

***[](https://blogs.sap.com/wp-content/uploads/2013/04/p_206030.jpg)***

method popup\_ok .

  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 current\_action    type wdapi\_action.

  lo\_api\_component  = wd\_this->wd\_get\_api( ).  
  lo\_window\_manager = lo\_api\_component->get\_window\_manager( ).

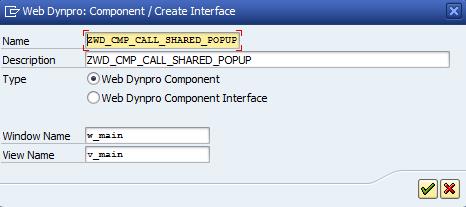
  call method lo\_window\_manager->create\_window  
    exporting  
      window\_name          = ‘W\_POPUP\_OK’  
      title                = ‘Confirmation’  
      message\_display\_mode = if\_wd\_window=>co\_msg\_display\_mode\_selected  
      close\_in\_any\_case    = abap\_true  
      button\_kind          = if\_wd\_window=>co\_buttons\_yesno  
      message\_type         = if\_wd\_window=>co\_msg\_type\_none  
      default\_button       = if\_wd\_window=>co\_button\_no  
      close\_button         = abap\_false  
      modal                = abap\_true  
    receiving  
      window               = wd\_this->popup\_window.

  wd\_this->popup\_window->set\_window\_position( position = if\_wd\_window=>co\_center ).  
  wd\_this->popup\_window->open( ).  
  wd\_this->popup\_window->subscribe\_to\_button\_event(  
                                     button = if\_wd\_window=>co\_button\_yes  
                                     action\_name = ‘YES’  
                                     action\_view = lo\_api ).  
  wd\_this->popup\_window->subscribe\_to\_button\_event(  
                                     button = if\_wd\_window=>co\_button\_no  
                                     action\_name = ‘NO’  
                                     action\_view = lo\_api ).

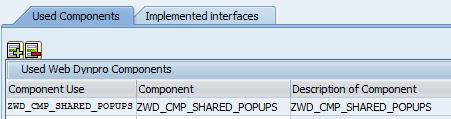
  free lo\_window\_manager.  
  if sy–subrc ne 0.  
  endif.  
  free lo\_api\_component.  
  if sy–subrc ne 0.  
  endif.

endmethod.

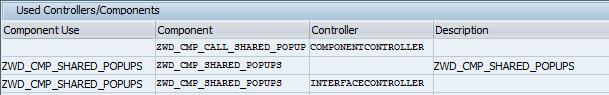
**Step 5:  Create another ABAP WebDynpro application to call this shared popup.**

***[](https://blogs.sap.com/wp-content/uploads/2013/04/p_206032.jpg)***

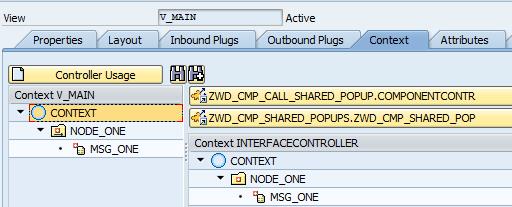
On the Used Components tab, add an entry for your ZWD\_CMP\_SHARED\_POPUPS component

[](https://blogs.sap.com/wp-content/uploads/2013/04/p_206033.jpg)

**Step 6:  In the main view, create a Component Usage for both the Shared Component and the Shared Component’s Interface Controller.**

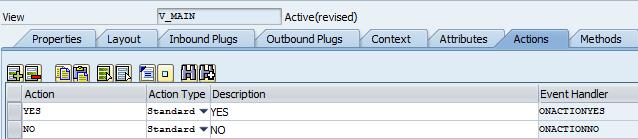
***[](https://blogs.sap.com/wp-content/uploads/2013/04/p_206034.jpg)***

Select the context tab and you will see the context of the shared component controller’s context.  Drag (bind) the node “NODE\_ONE” to your view as follows:

[](https://blogs.sap.com/wp-content/uploads/2013/04/p_206035.jpg)

\*This binding is possible because we made the node an “Interface Node”.

**Step 7:  In the same main view, create two event handlers for the popup responses.**

***[](https://blogs.sap.com/wp-content/uploads/2013/04/p_206036.jpg)***

\*you can choose your own code to add in these events (if any at all).

**Step 8:  Add the following code to the hook method: WDDOINIT.**

method wddoinit .

\* Use the Wizard to Set a Value of MSG\_ONE which the Popup will Display  
  data lo\_nd\_node\_one type ref to if\_wd\_context\_node.  
  data lo\_el\_node\_one type ref to if\_wd\_context\_element.  
  data ls\_node\_one type wd\_this->element\_node\_one.  
  data lv\_msg\_one type wd\_this->element\_node\_one–msg\_one.

  lo\_nd\_node\_one = wd\_context->get\_child\_node( name = wd\_this->wdctx\_node\_one ).  
  lo\_el\_node\_one = lo\_nd\_node\_one->get\_element( ).  
  lv\_msg\_one = ‘Are You Sure You Wish To See The Houston Dynamo Play Soccer?’.  
  lo\_el\_node\_one->set\_attribute( name =  `MSG\_ONE` value = lv\_msg\_one ).

\* Use the Wizard to Instantiate the Used Component  
  data lo\_cmp\_usage type ref to if\_wd\_component\_usage.  
  lo\_cmp\_usage = wd\_this->wd\_cpuse\_zwd\_cmp\_shared\_popups( ).  
  if lo\_cmp\_usage->has\_active\_component( ) is initial.  
    lo\_cmp\_usage->create\_component( ).  
  endif.

\* Use the Wizard to call the method of the Used Component  
  data lo\_api type ref to if\_wd\_view\_controller.  
  data lo\_interfacecontroller type ref to ziwci\_wd\_cmp\_shared\_popups .  
  lo\_api = wd\_this->wd\_get\_api( ).  
  lo\_interfacecontroller = wd\_this->wd\_cpifc\_zwd\_cmp\_shared\_popups( ).  
  lo\_interfacecontroller->popup\_ok( lo\_api = lo\_api ).

endmethod.

That’s it…  Test your application to see if it correctly calls the shared component and displays the popup and also displays the message you saved in the mapped context attribute.

[](https://blogs.sap.com/wp-content/uploads/2013/04/p_206037.jpg)