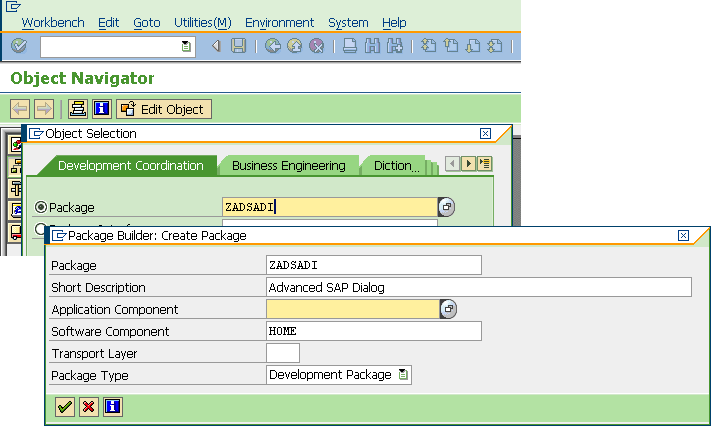
Author: Frithjof Eckhardt, updated: 25. Okt. 2013

**Note**: The package comes with the namespace /UKW/, which is the namespace for Wuerzburg University Hospital, Germany. You can’t create objects in this namespace in your SAP system. Therefore map the namespace prefix of each object from /UKW/ to any name starting with Z or Y, or your own customer namespace.

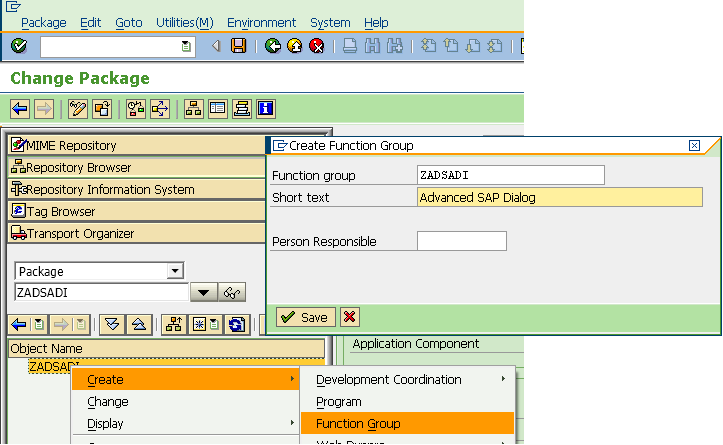
Installation

1. Package ZADSADI
2. Function Group ZADSADI
3. Function Module Z\_ADSADI with one screen 100 carrying a custom control
4. Create dictionary objects for storing html content
5. Interface ZIF\_ADSADI\_CALLBACK
6. Class ZCL\_ADSADI\_DIALOG

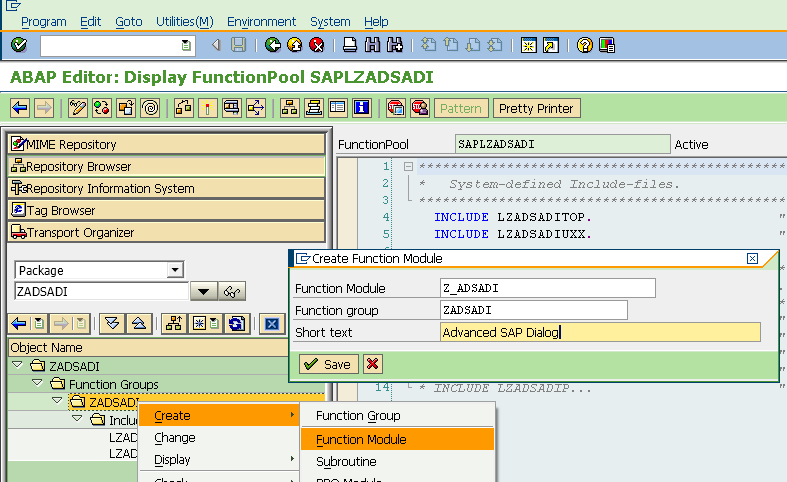
# Create new Package, e.g. ZADSADI



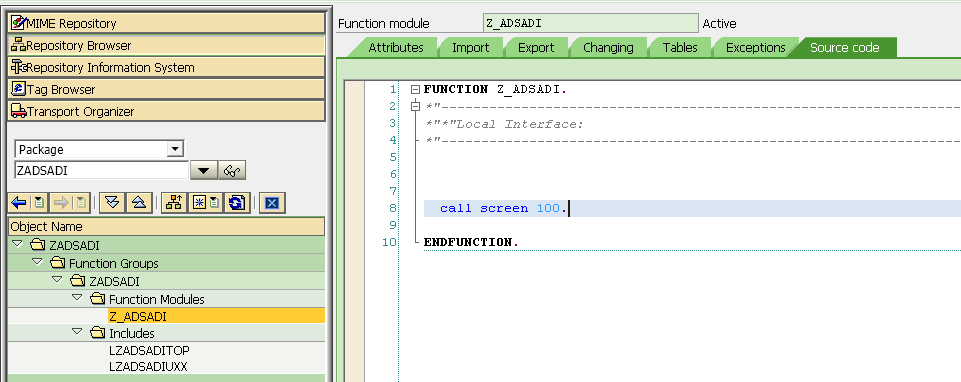
# Within the package create and activate Function Group, e.g. ZADSADI



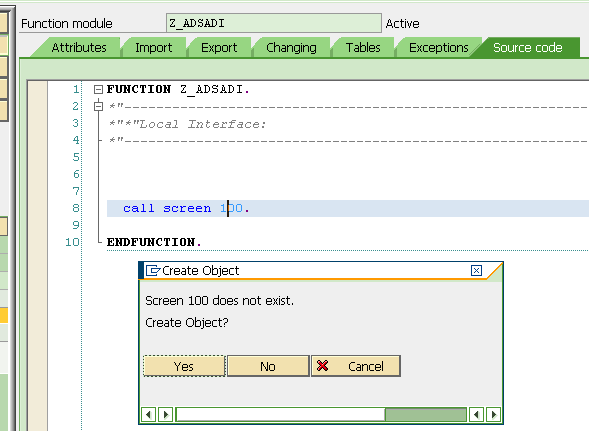
# Within the function group create Function Module, e.g. Z\_ADSADI



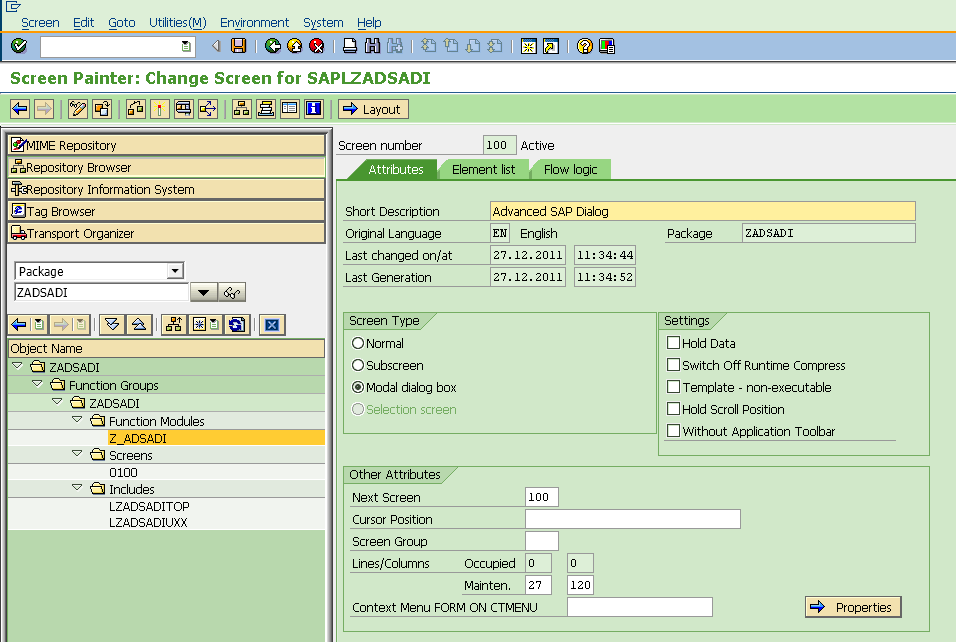
### Add the following line to the source code of the function module and activate



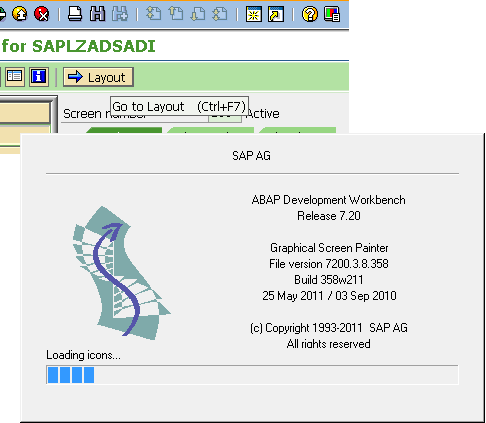
### Create screen 100 by double clicking the „100“ in the source code:



### Set the screen type to „Modal dialog box“ and activate



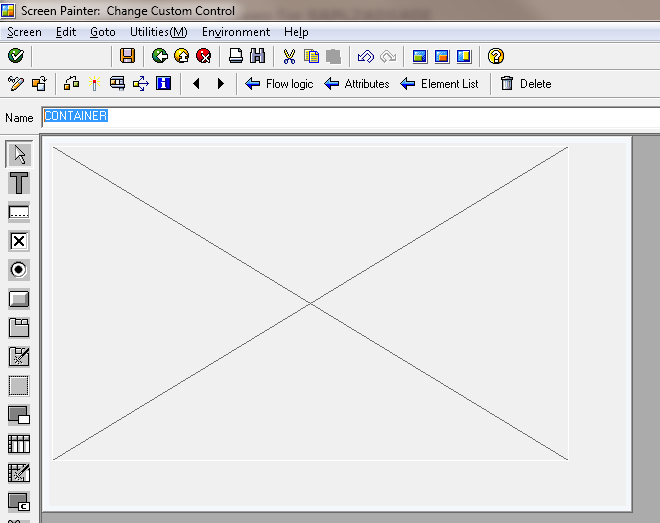
### Create the layout for the screen



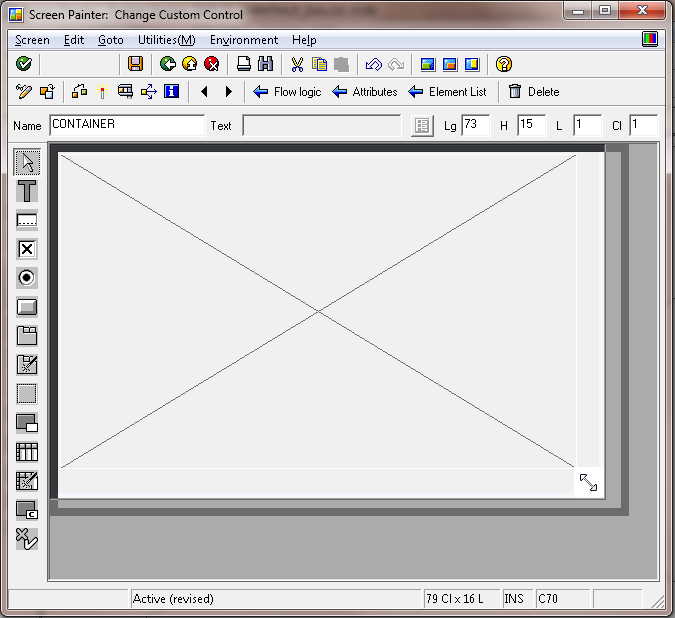
### Create a custom control screen element



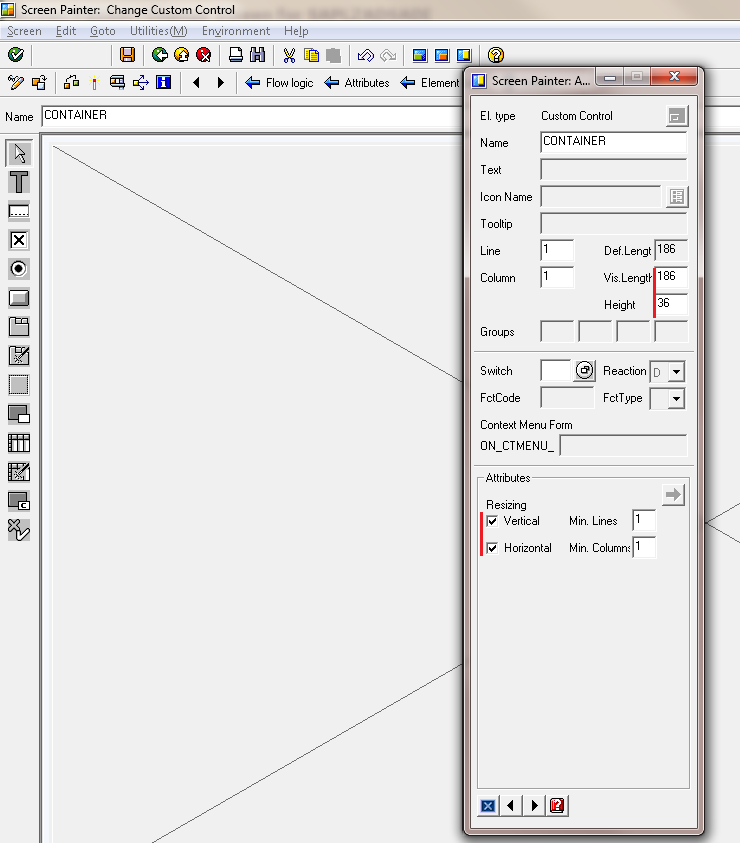
### Set the name of the control to „CONTAINER“

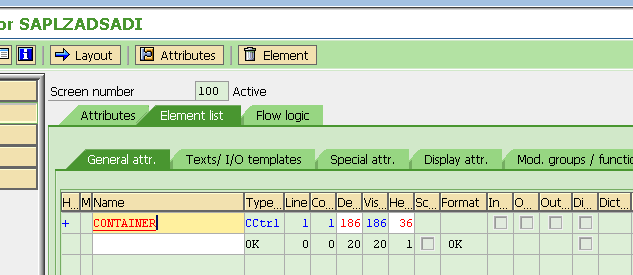


### Resize the window below the container so that it is a little greater than the screen size of the target monitor , e.g. 186 columns, 36 rows for 19“ monitor (trial&error determined)



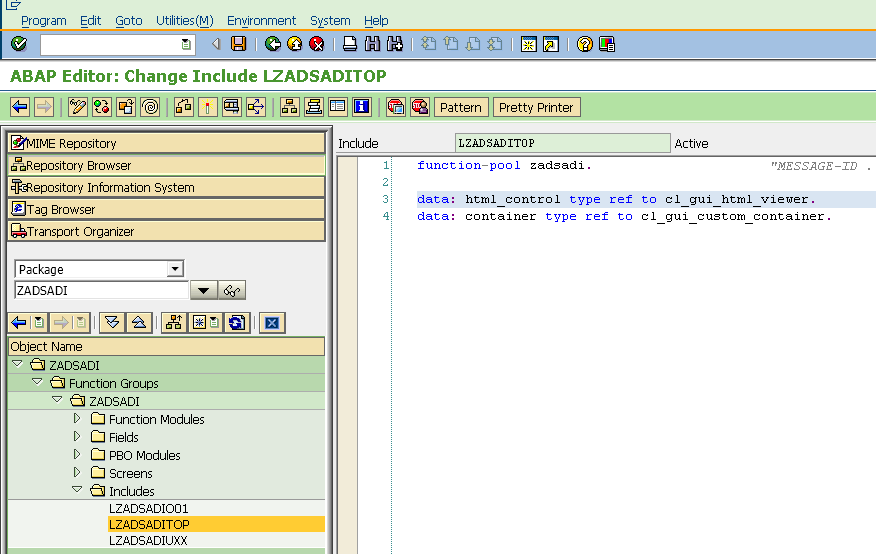
### Set the dimension of the container to the same size as the underlying window and link both together by setting the resizing attributes of the container and activate



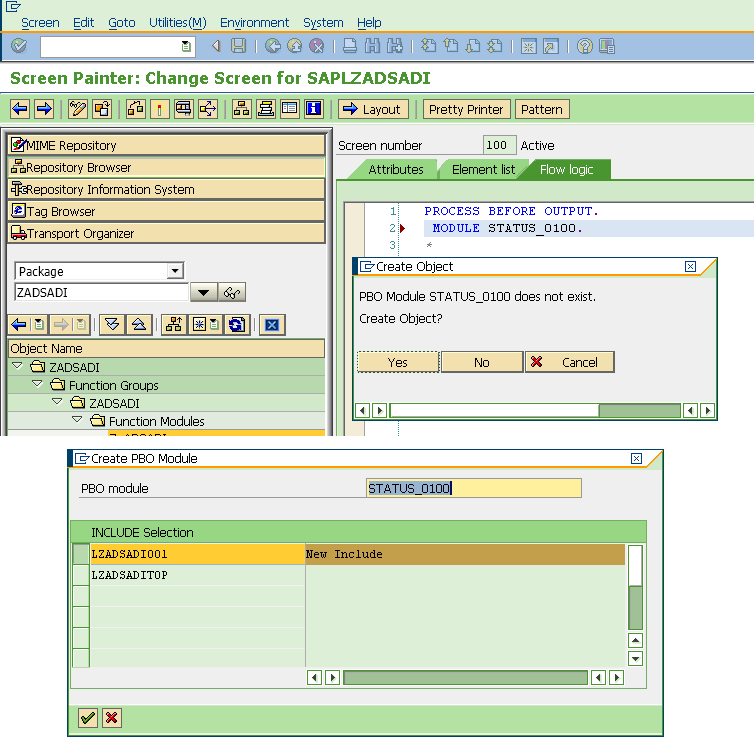


### Edit the top include of the function module and enter the following data definitions

data: html\_control type ref to cl\_gui\_html\_viewer.  
data: container type ref to cl\_gui\_custom\_container.

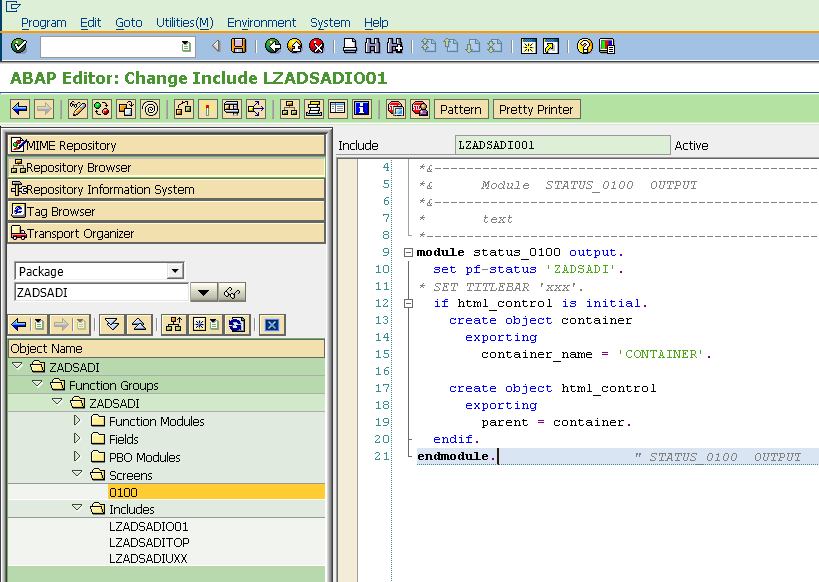


### Create Flow logic of the screen for „PROCESS BEFORE OUTPUT“ by uncommenting the respective module call and creating the source code by clicking twice on the module name „STATUS\_0100“

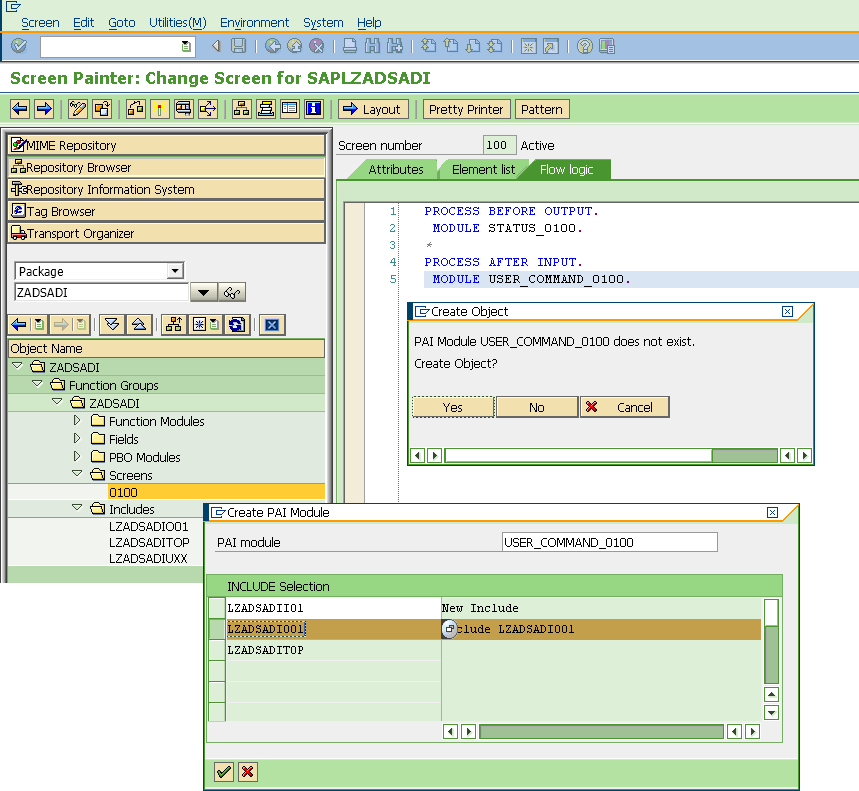


### Enter the following code

module status\_0100 output.  
  set pf-status 'ZADSADI'.  
\* SET TITLEBAR 'xxx'.  
  if html\_control is initial.  
    create object container  
      exporting  
        container\_name = 'CONTAINER'.  
  
    create object html\_control  
      exporting  
        parent = container.  
  endif.  
endmodule.

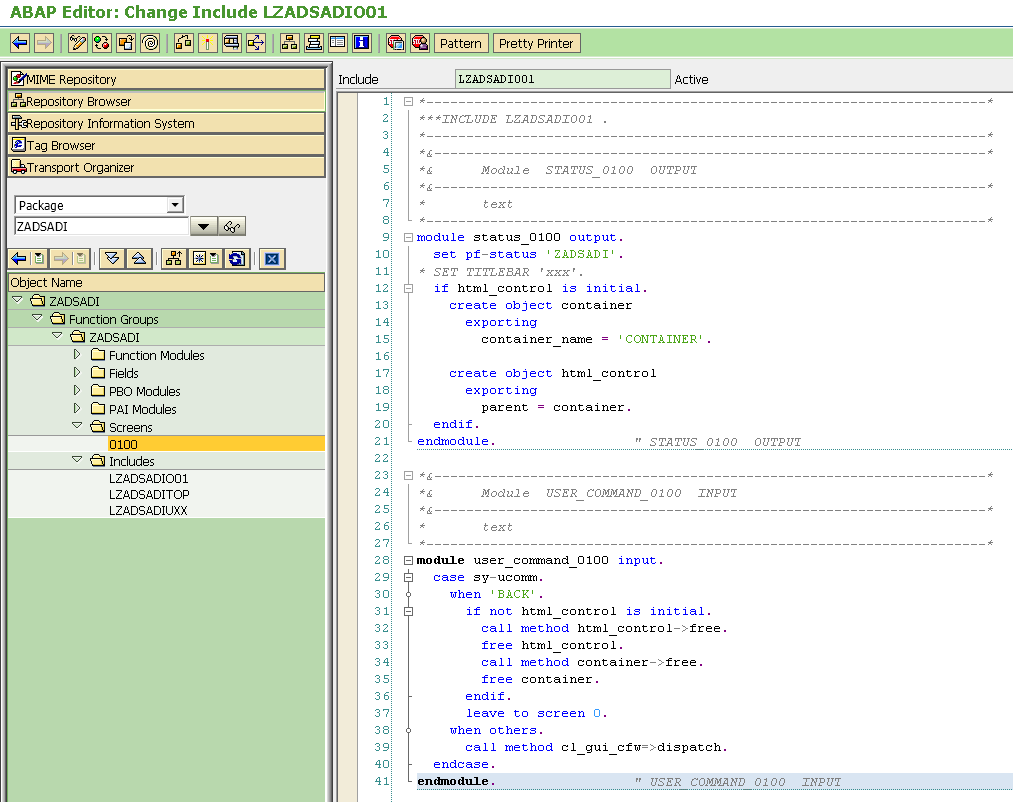


### In the same include, create Flow logic of the screen for „PROCESS AFTER INPUT“ by uncommenting the respective module call and creating the source code by clicking twice on the module name „USER\_COMMAND\_0100“

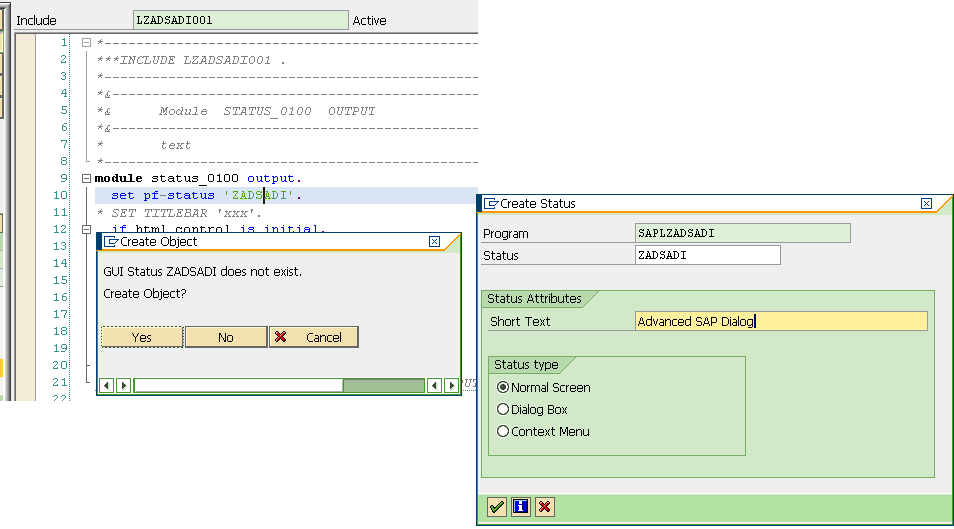


### Enter the following code

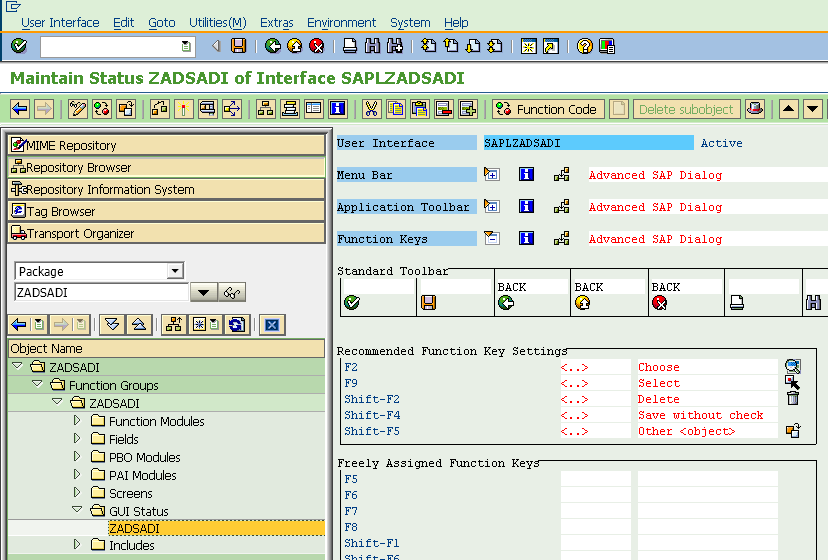
module user\_command\_0100 input.  
  case sy-ucomm.  
    when 'BACK'.  
      if not html\_control is initial.  
        call method html\_control->free.  
        free html\_control.  
        call method container->free.  
        free container.  
      endif.  
      leave to screen 0.  
    when others.  
      call method cl\_gui\_cfw=>dispatch.  
  endcase.  
endmodule.



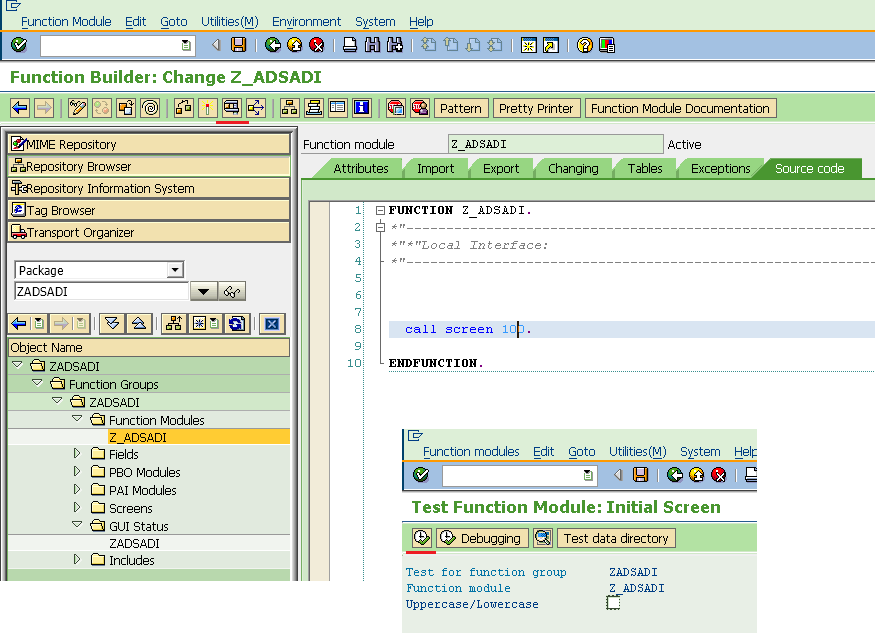
### Create the GUI status by double-clicking on the „ZADSADI“ in the status\_0100 module in the line „set pf-status 'ZADSADI'

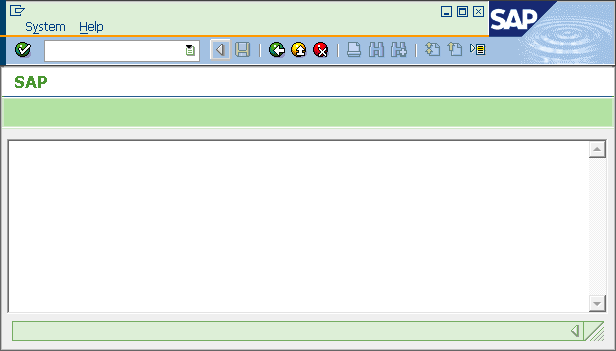


### Enter the function code „BACK“ for all the navigation buttons (back, exit, cancel) in the toolbar section and activate the status

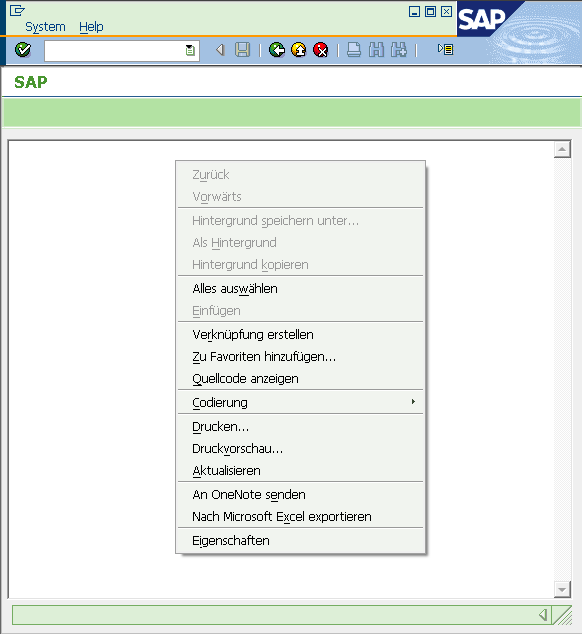


### Test the screen by testing the function module



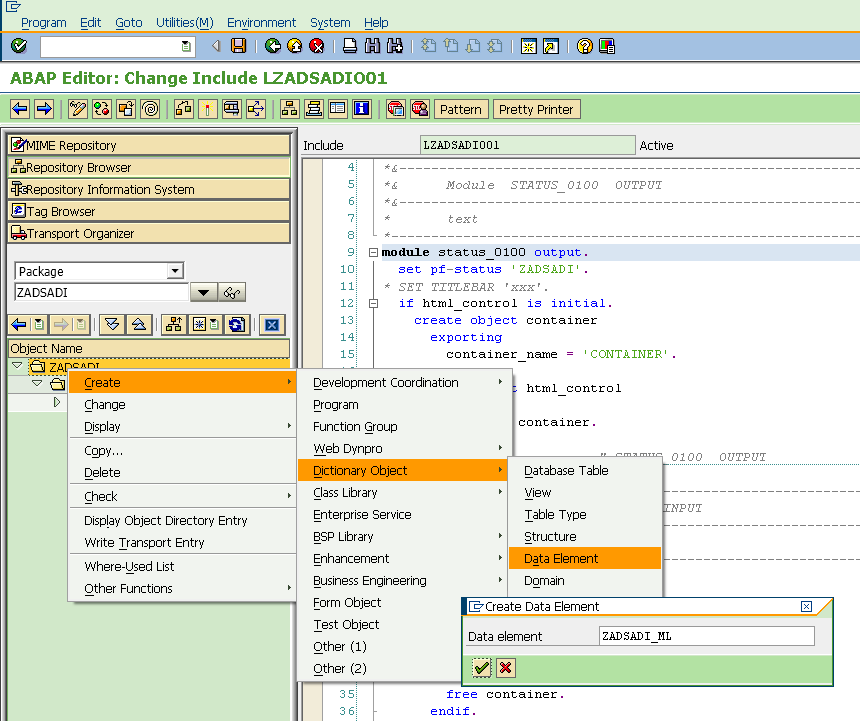
A white window appears, that is resizeable and any navigation button brings the user back

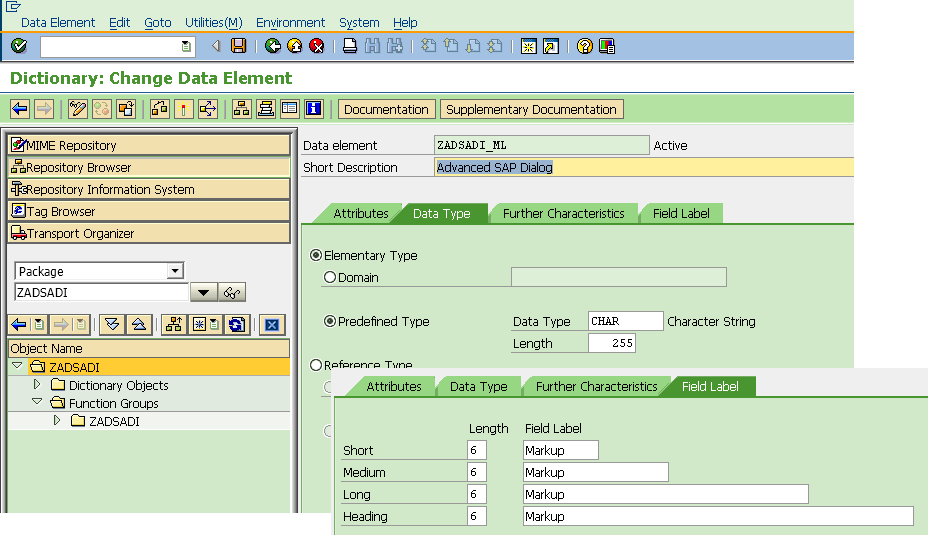
The white window is an html page, verify it by hiting the right mouse button to show the Internet Explorer context menu



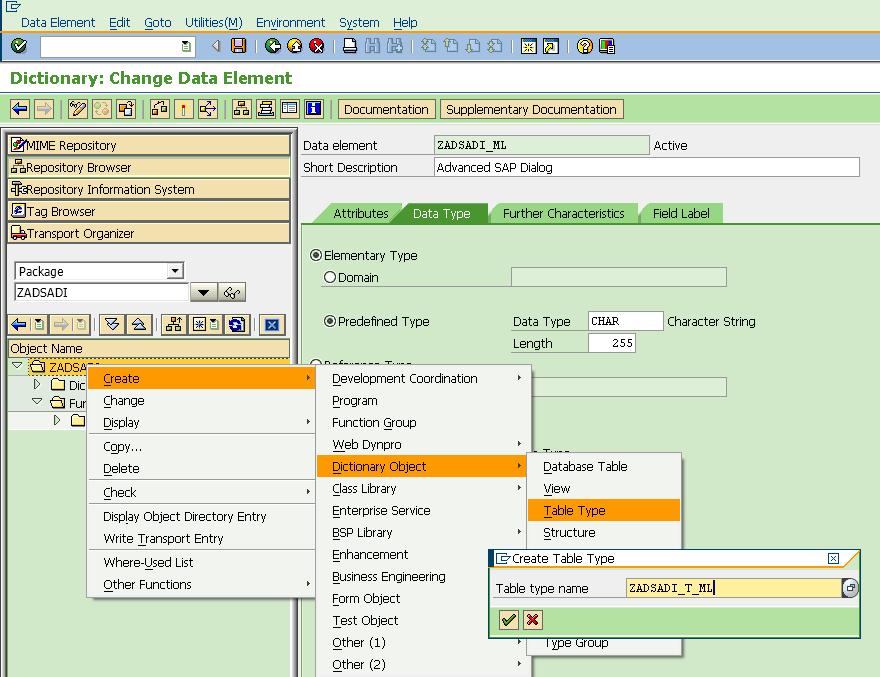
Create the backing logic

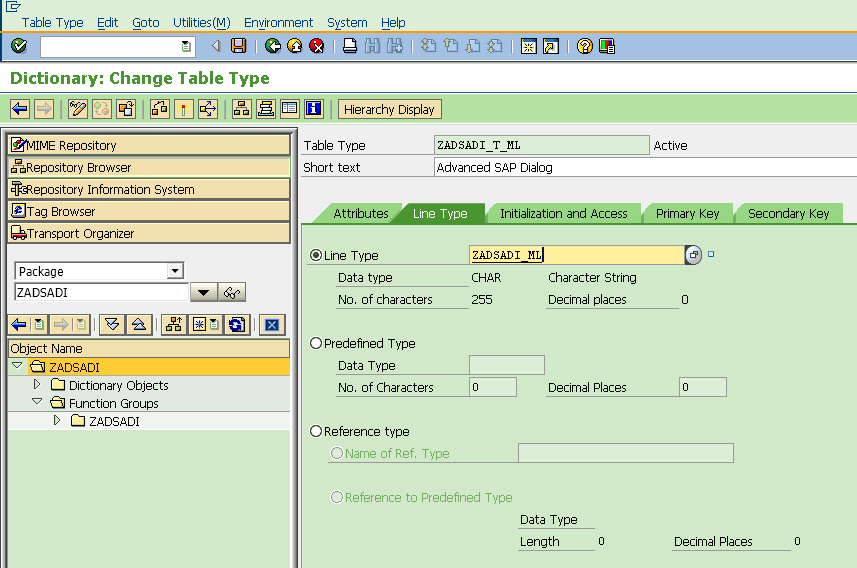
# Create a data element ZADSADI\_ML for storing a line of html markup





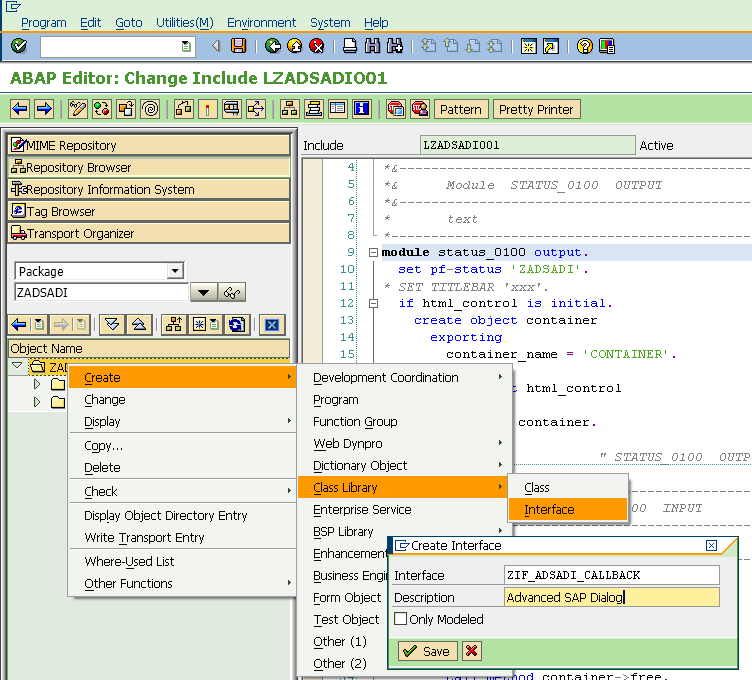
# Create a table type ZADSADI\_T\_ML for html content

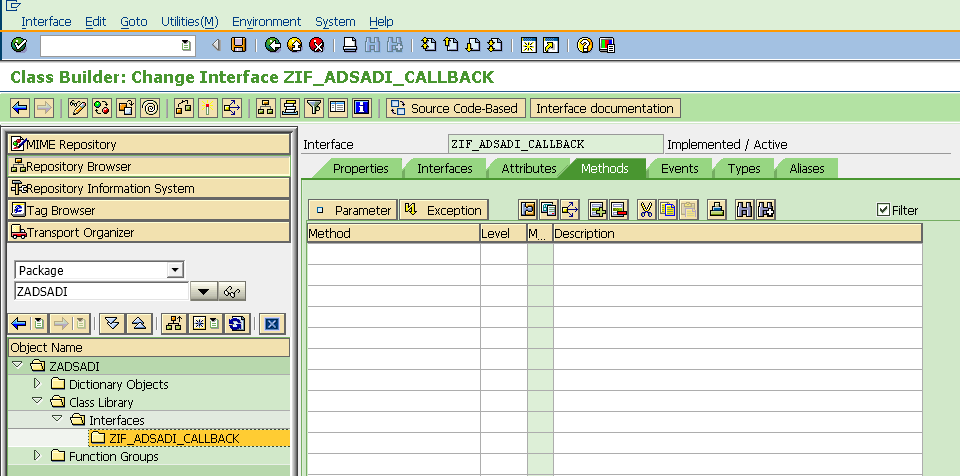




# Create an interface ZIF\_ADSADI\_CALLBACK

This interface is used from within the screen 100 implementation to get the html content to be displayed and to provide the parameters returned from the html viewer.



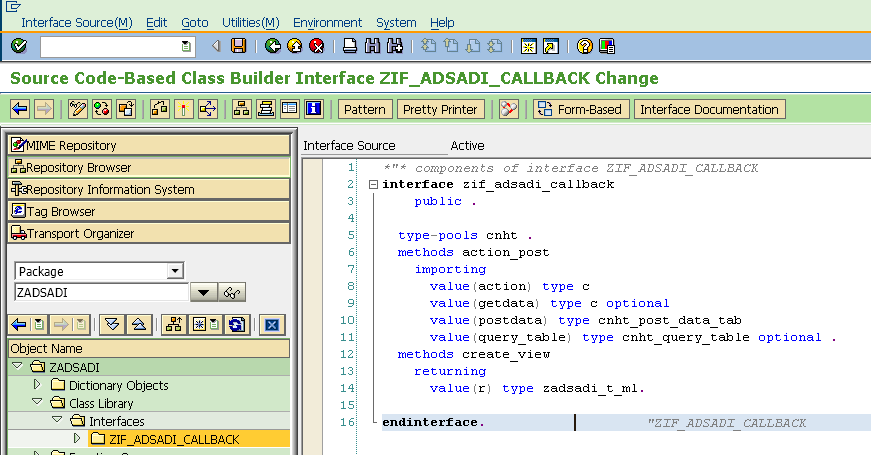


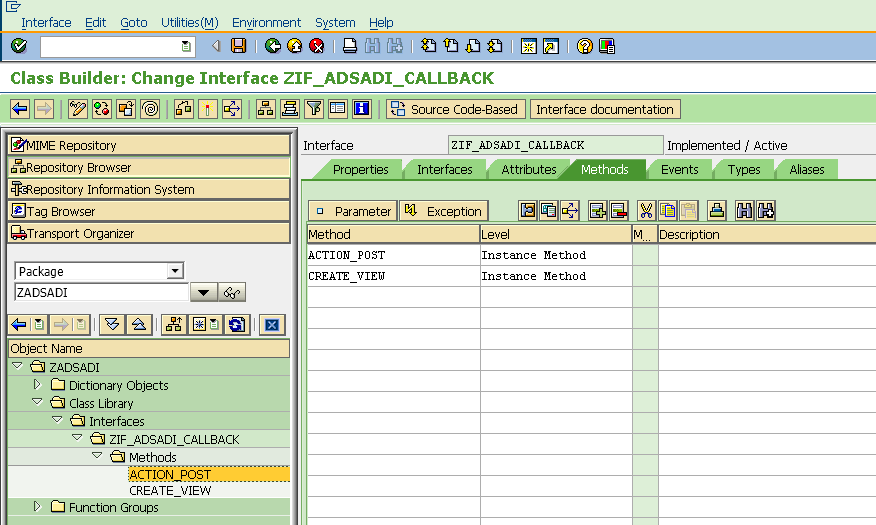
Switch to „Source Code-Based“ view of the class builder. If this button is not available (due to an older release), create the methods the traditional way one at a time with the class builders help.

In the source code-based view, enter the following interface code and switch back to „Form-based“ view.

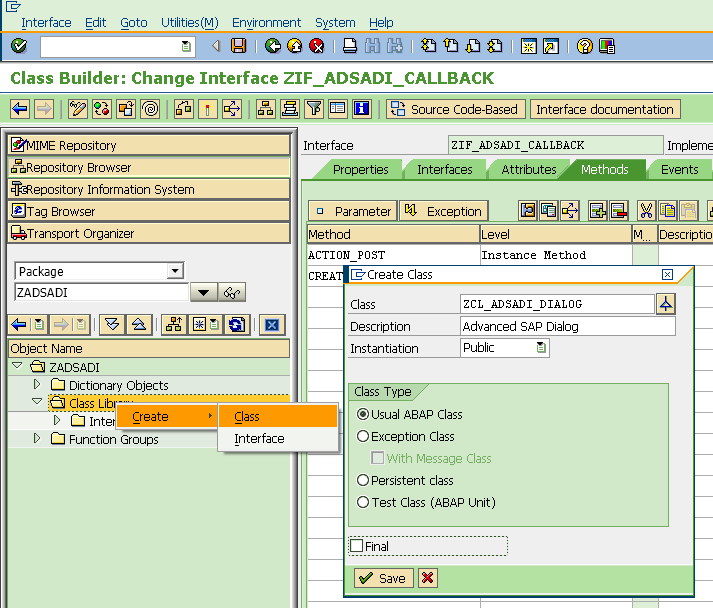
interface zif\_adsadi\_callback public.  
  
  type-pools cnht.

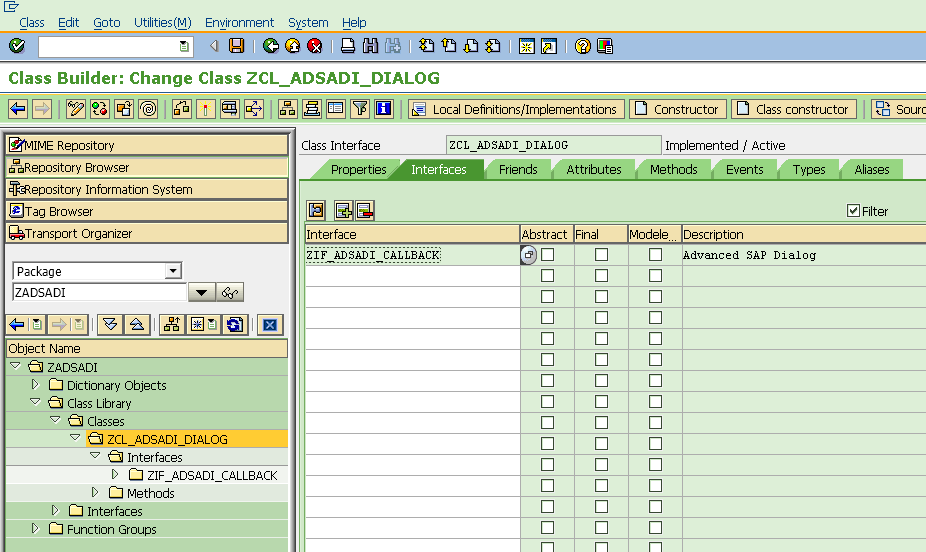
  constants dlg\_cancel type i value 1.                       
  constants dlg\_no\_cancel type i value 0.                    
  
  methods action\_post  
    importing  
      value(action) type c  
      value(getdata) type c optional  
      value(postdata) type cnht\_post\_data\_tab  
      value(query\_table) type cnht\_query\_table optional .  
  methods create\_view  
    returning  
      value(r) type zadsadi\_t\_ml.  
  
endinterface.



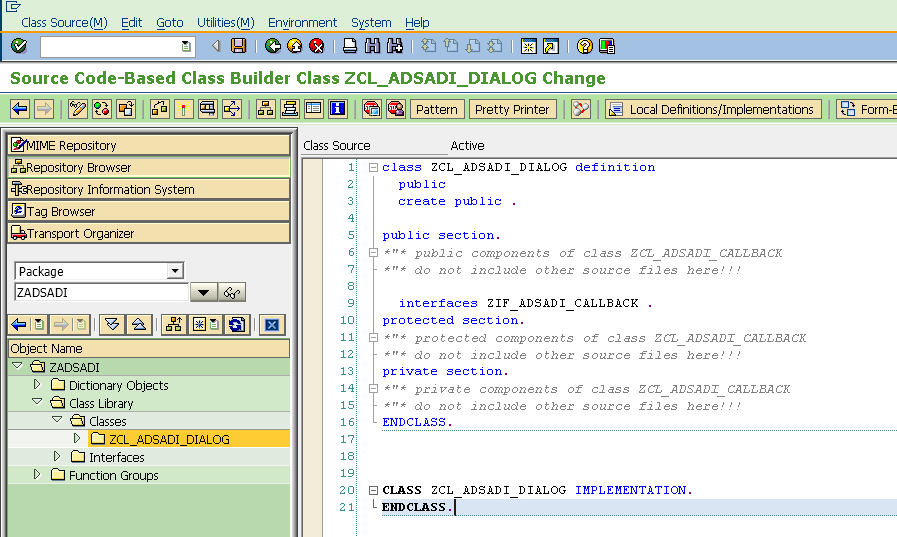


# Create an implementation ZCL\_ADSADI\_DIALOG against this interface





Switch to „Source Code-based“ view and edit the source code the following way:



## Public section

    interfaces zif\_adsadi\_callback .  
  
    aliases create\_view  
      for zif\_adsadi\_callback~create\_view.  
    aliases do\_post  
      for zif\_adsadi\_callback~action\_post.  
  
    type-pools abap.   
    type-pools cnht.  
  
    methods constructor  
      importing  
        value(i\_xslt) type csequence  
        value(i\_xml) type csequence optional  
        value(i\_xml\_doc) type ref to if\_ixml\_node optional  
        value(i\_params) type abap\_trans\_parmbind\_tab optional  
        value(i\_size) type csequence default ''  
        value(i\_top\_left\_y) type i default 3  
        value(i\_top\_left\_x) type i default 40  
        value(i\_width) type i default 100  
        value(i\_height) type i default 25.  
    methods get\_value  
      importing  
        value(n) type csequence  
      returning  
        value(r) type string .  
    methods render  
      returning  
        value(canceled) type i .

## Protected section

    data xml type string .  
    data xml\_doc type ref to if\_ixml\_node .

## Private section

    data xslt type string.  
    data xslt\_params type abap\_trans\_parmbind\_tab.  
    data query\_table type cnht\_query\_table.  
    data size type string.  
    data top\_left\_y type i.  
    data top\_left\_x type i.  
    data width type i.  
    data height type i.

## Constructor implementation

  method constructor.  
    me->xslt\_params  = i\_params.  
    me->xslt         = i\_xslt.  
    me->size         = i\_size.  
    me->top\_left\_y   = i\_top\_left\_y.  
    me->top\_left\_x   = i\_top\_left\_x.  
    me->width        = i\_width.  
    me->height       = i\_height.  
    if i\_xml is not initial.  
      me->xml = i\_xml.  
    elseif i\_xml\_doc is bound.  
      me->xml\_doc = i\_xml\_doc.  
    else.  
      concatenate  
        `<?xml version="1.0" encoding="utf-8" ?>`  
        `<dialog />`  
        into me->xml.  
    endif.  
  endmethod.

## Method get\_value implementation

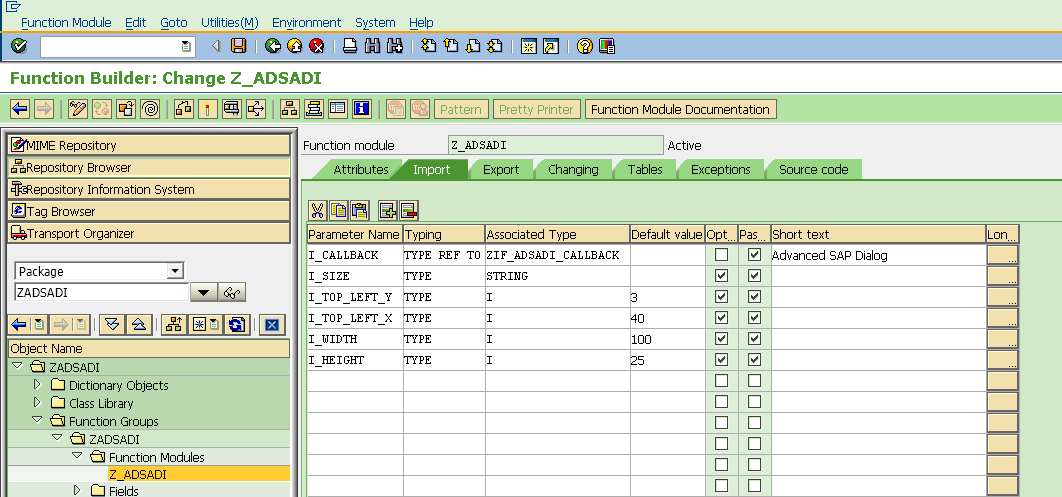
  method get\_value.  
    data: the\_param type w3query.  
  
    loop at me->query\_table into the\_param where name eq n.  
      r = the\_param-value.  
      exit.  
    endloop.  
  endmethod.

## **Method** render **implementation**

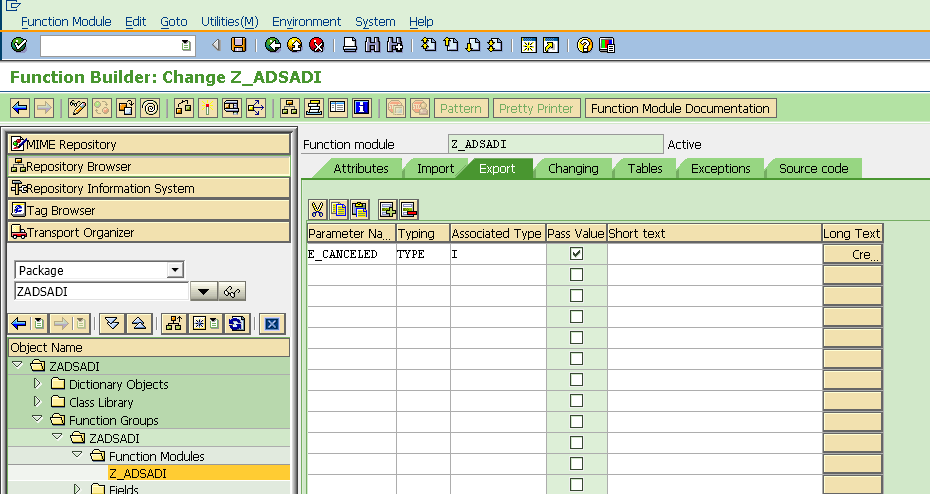
  method render.  
    call function 'Z\_ADSADI'  
      exporting  
        i\_callback   = me  
        i\_size       = me->size  
        i\_top\_left\_y = me->top\_left\_y  
        i\_top\_left\_x = me->top\_left\_x  
        i\_width      = me->width  
        i\_height     = me->height  
      importing  
        e\_canceled   = canceled.  
  endmethod.

## Modify the interface of the function module to match the call from the render Method

## Import



## Export



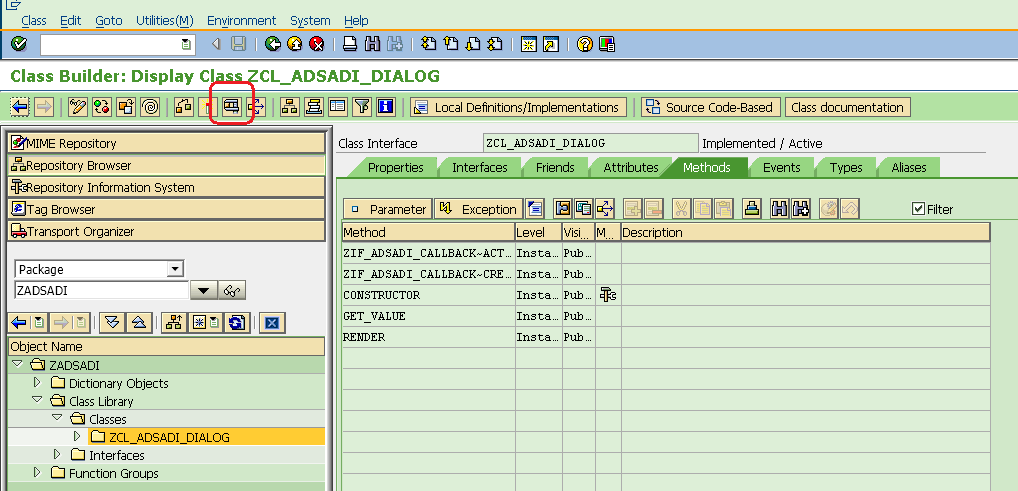
## Source code function module Z\_ADSADI

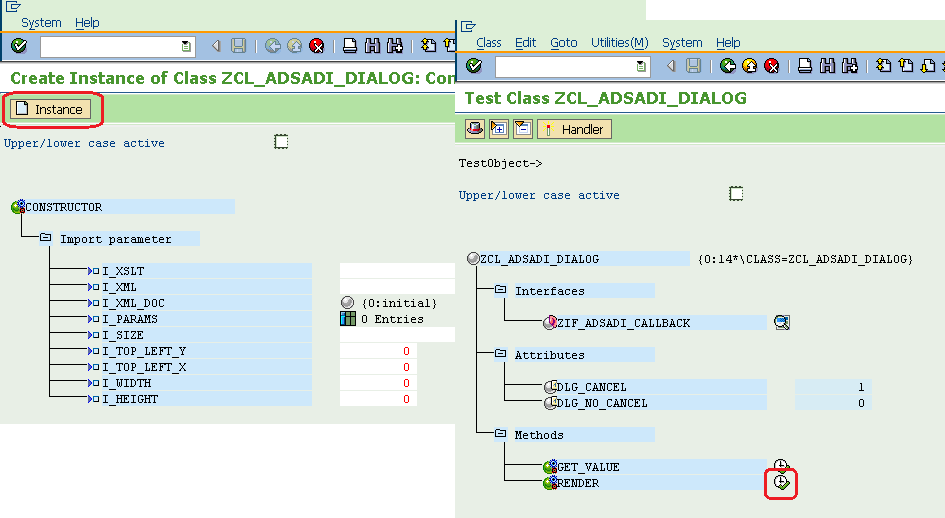
  g\_callback = i\_callback.  
  g\_canceled = zif\_adsadi\_callback=>dlg\_cancel.  
  
  if i\_size eq 'NORMAL'.  
    call screen 100  
      starting at 40  3  
      ending   at 140 30.  
  elseif i\_size eq 'FULL'.  
    call screen 100.  
  else.  
    data: bottom\_right\_x type i.  
    data: bottom\_right\_y type i.  
  
    bottom\_right\_x = i\_top\_left\_x + i\_width.  
    bottom\_right\_y = i\_top\_left\_y + i\_height.  
    call screen 100  
      starting at i\_top\_left\_x   i\_top\_left\_y  
      ending   at bottom\_right\_x bottom\_right\_y.  
  endif.  
  
  e\_canceled = g\_canceled.

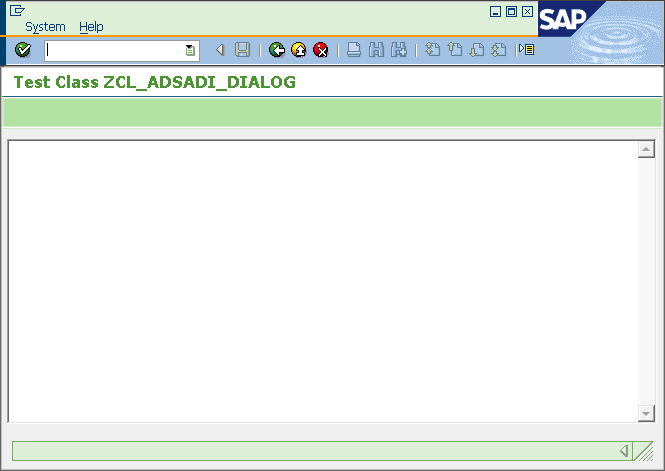
## Source code top include of function module Z\_ADSADI

function-pool zadsadi.                      "MESSAGE-ID ..  
  
data: html\_control type ref to cl\_gui\_html\_viewer.  
data: container type ref to cl\_gui\_custom\_container.  
  
data: g\_callback type ref to zif\_adsadi\_callback.  
data: g\_canceled type i.

# Test the dialog implementation







# Back to screen 100, breathe life into it

## Top include

data: myevent\_tab TYPE cntl\_simple\_events.  
data: myevent TYPE cntl\_simple\_event.

In the module include above status\_0100, add an event handler for the HTML Viewer control:

class cl\_myevent\_handler definition.  
  public section.  
    methods: on\_sapevent  
      for event sapevent  
      of cl\_gui\_html\_viewer  
      importing action frame getdata postdata query\_table.  
endclass.

class cl\_myevent\_handler implementation.  
  
  method on\_sapevent.  
    if g\_callback is bound.  
      call method g\_callback->action\_post  
        exporting  
          action      = action  
          getdata     = getdata  
          postdata    = postdata  
          query\_table = query\_table.  
      if action is not initial.  
        g\_canceled = zif\_adsadi\_callback=>dlg\_no\_cancel.  
      endif.  
      perform close.  
    endif.  
  endmethod.                    "on\_sapevent  
endclass.

data: evt\_receiver type ref to cl\_myevent\_handler.

## Refactor out code to destroy the control

module user\_command\_0100 input.  
  case sy-ucomm.  
    when 'BACK'.  
      perform close.  
    when others.  
      call method cl\_gui\_cfw=>dispatch.  
  endcase.  
endmodule.

form close.  
  if not html\_control is initial.  
    call method html\_control->free.  
    free html\_control.  
    call method container->free.  
    free container.  
  endif.  
  leave to screen 0.  
endform.

## Add code for event registration and page content loading

module status\_0100 output.  
  set pf-status 'ZADSADI'.  
\* SET TITLEBAR 'xxx'.  
  if html\_control is initial.  
    create object container  
      exporting  
        container\_name = 'CONTAINER'.  
  
    create object html\_control  
      exporting  
        parent = container.  
  
\*   register event  
    myevent-eventid = html\_control->m\_id\_sapevent.  
    myevent-appl\_event = 'x'.  
    append myevent to myevent\_tab.  
  
    html\_control->set\_registered\_events( myevent\_tab ).  
    create object evt\_receiver.  
    set handler evt\_receiver->on\_sapevent for html\_control.  
  
    perform load\_page\_content.  
  endif.  
endmodule.

form load\_page\_content.  
  data: doc\_url(80).  
  data: html\_tab type zadsadi\_t\_ml.  
  data: html\_l type line of zadsadi\_t\_ml.  
  
  refresh html\_tab.  
  
  html\_tab = g\_callback->create\_view( ).  
  
  call method html\_control->load\_data  
    exporting  
      url          = doc\_url  
      size         = 0  
      type         = 'text'  
      subtype      = 'html'  
    importing  
      assigned\_url = doc\_url  
    changing  
      data\_table   = html\_tab  
    exceptions  
      others       = 1.  
  
  if sy-subrc eq 0.  
    call method html\_control->show\_url  
      exporting  
        url = doc\_url.  
  endif.  
endform.

## Implement ZCL\_ADSADI\_DIALOG methods derived from interface

### Method action\_post

  method zif\_adsadi\_callback~action\_post.  
    data: the\_param type w3query.  
    clear: me->query\_table.  
    " Gather post and get data  
    if getdata is not initial.  
      the\_param-name = action.  
      the\_param-value = getdata.  
      append the\_param to me->query\_table.  
    endif.  
    loop at query\_table into the\_param.  
      append the\_param to me->query\_table.  
    endloop.  
  endmethod.

### Method create\_view

  method zif\_adsadi\_callback~create\_view.  
    data: ex type ref to  cx\_xslt\_runtime\_error.  
    data: ex2 type ref to cx\_transformation\_error.  
  
    data: param type abap\_trans\_parmbind.  
  
    data: begin of bgr,  
          h1 type c length 2,  
          h2 type c length 2,  
          h3 type c length 2,  
        end of bgr.  
  
    data: hex type x length 3.  
    data: buf type c length 10.  
    data: color type i.  
  
\*   Background-Color1  
    param-name = 'SAP\_BACKGROUNDCOLOR1'.  
    param-value = '#FFFFFF'.  
  
    " Overwrite default with system value  
    call method cl\_gui\_resources=>get\_background\_color  
      exporting  
        id     = cl\_gui\_resources=>col\_background\_level1  
        state  = 0  
      importing  
        color  = color  
      exceptions  
        others = 1.  
    if sy-subrc = 0.  
      bgr = buf = hex = color.  
      concatenate `#` bgr-h3 bgr-h2 bgr-h1 into param-value.  
    endif.  
    append param to me->xslt\_params.  
  
\*   Background-Color2  
    param-name = 'SAP\_BACKGROUNDCOLOR2'.  
    param-value = '#CCCCCC'.  
  
    " Overwrite default with system value  
    call method cl\_gui\_resources=>get\_background\_color  
      exporting  
        id     = cl\_gui\_resources=>col\_background\_level2  
        state  = 0  
      importing  
        color  = color  
      exceptions  
        others = 1.  
    if sy-subrc = 0.  
      bgr = buf = hex = color.  
      concatenate `#` bgr-h3 bgr-h2 bgr-h1 into param-value.  
    endif.  
    append param to me->xslt\_params.  
  
  
    try.  
        if me->xml\_doc is bound.  
          call transformation (me->xslt)  
                   parameters (me->xslt\_params)  
                   source xml me->xml\_doc  
                   result xml r.  
        else.  
          call transformation (me->xslt)  
                   parameters (me->xslt\_params)  
                   source xml me->xml  
                   result xml r.  
        endif.  
      catch: cx\_xslt\_runtime\_error into ex.  
      catch: cx\_invalid\_transformation into ex2.  
    endtry.  
  endmethod.