Custom Client

GXflow Custom Client is provided by GeneXus and can be imported into the Knowledge Base to use or modify it as a part of our project. It contains all the Webpanels and procedures needed to implement many functionality of GXflow Standard Client, we have seen in previous parts of this course.

Using the GXflow Custom Client, you can login to GXflow, execute the user inbox, see the outbox tray, administrate de processes definitions, workitems, instances and even monitor the execution.

If you look at the source code programmed in these Webpanels, we can see how they use the GXflow APIS, to connect to the GXflow engine and get the information needed in each case, and execute different functions to each object, etc.

In this video we are going to see some Webpanels from the Custom Client to understand its code and how they can be modified.

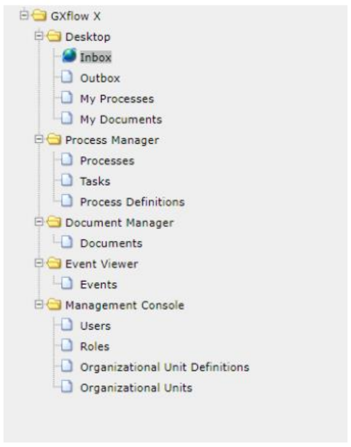
We can modify the Custom Clients for several reasons, for example we can add o hide information, integrate de Custom Client to our diferentes panels, Add actions at the different panels to include some code . All the UX of this Webpanels can be modified and adjusted to your project, .

We use the Custom Client to integrate the Client of GXflow into our projects as a part of our project without the user interface of GXflow, and using our own UX interface.

To see more information about Custom Client we can follow this link

https://wiki.GeneXus.com/commwiki/servlet/wiki?11364,HowTo%3A+Customize+The+GXflow+Client

The Custom Clients contains a fews folders: Desktop folders, the ProccesManager Folders, a Document Managment folder, Event viewer and Managment Console folder



The Custom Clients contains in the desktop folder the inbox, outbox, my processes and my documents, they provide the same informaction and actions as the Standard Client.

The Process Manager folder contains the process pages, the task and the process definitions, that are de more used pages from the process manager section.

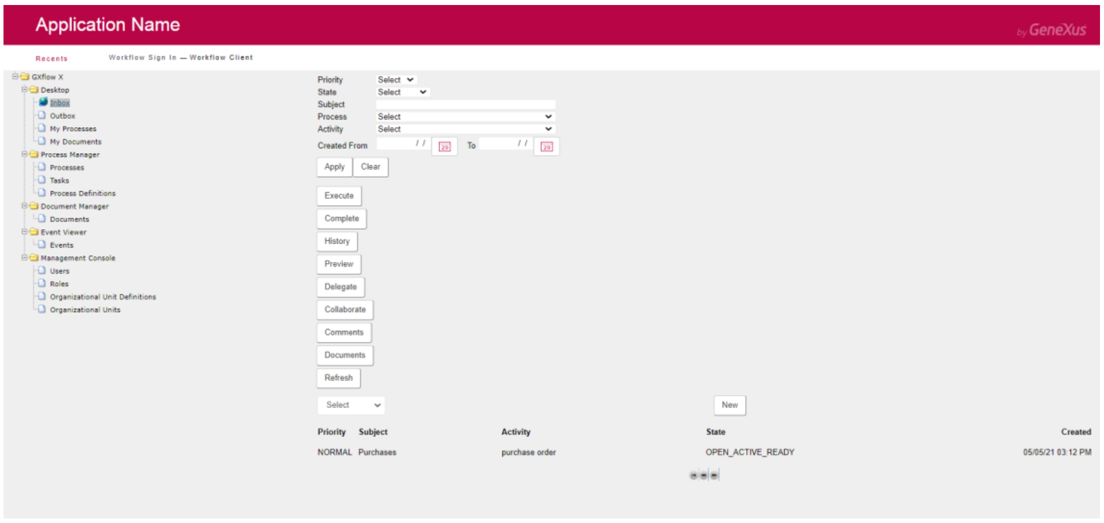
The Documents manager folder contains the document administration page the same as the Standard Client.

The event viewer folder contains the events manager page the same as the Standard Client.

The management console folder contains the user administration, the roles administration, the organizational units definitions and the organizational units pages, the same as the standard Client.

Statistics, backend, server settings and license manager are not implemented in the Custom Clients example.

The main use of the Custom Clients is incorporate to GXflow pages to our projects using the same UX. For example this page presents an inbox Customized with a look & feel and behavior that matches the rest of your application. The user of our application will not feel the difference from other parts of the application.

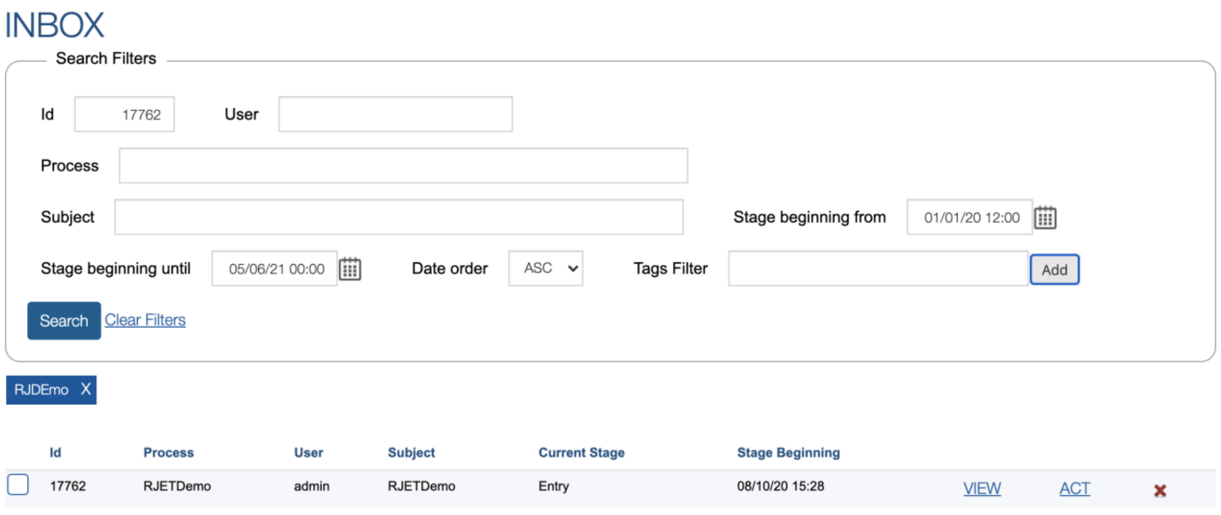


Priority	Select
State	Select
Subject	
Process	Select
Activity	Select
Created From	/ / To / /

Select

Priority	Subject	Activity	State	Created
NORMAL	Purchases	purchase order	OPEN_ACTIVE_READY	05/05/21 03:12 PM

Custom Client Inbox



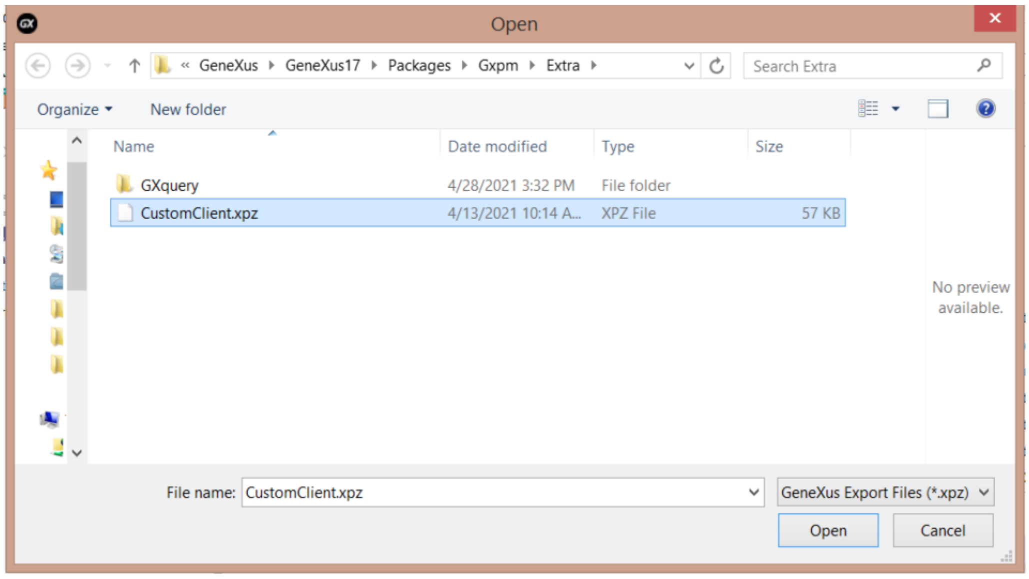
Id	17762	User			
Process					
Subject				Stage beginning from	01/01/20 12:00
Stage beginning until	05/06/21 00:00	Date order	ASC	Tags Filter	

	Id	Process	User	Subject	Current Stage	Stage Beginning			
☐	17762	RJETDemo	admin	RJETDemo	Entry	08/10/20 15:28	VIEW	ACT	✖

Customized Client Inbox

There is a file called CustomClient.xpz and is distributed with the GeneXus installation within: <GeneXus installation directory>\Packages\GXPM\Extra.

Before importing it, check that a least one Business Process Diagram has been created in this Knowledge Base. This is necessary to make sure that the model has the workflow data types. Otherwise, there will be an error during compilation of the project.

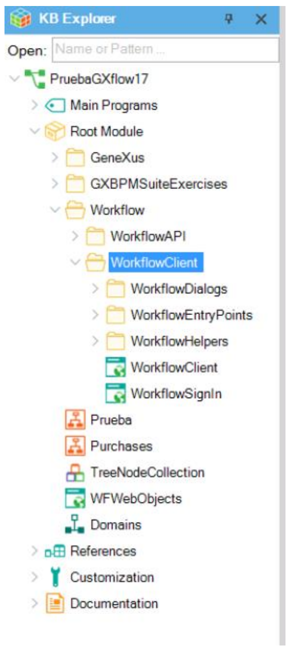


Name	Date modified	Type	Size
GXquery	4/28/2021 3:32 PM	File folder	
CustomClient.xpz	4/13/2021 10:14 A...	XPZ File	57 KB

File name: CustomClient.xpz — GeneXus Export Files (*.xpz)

When we are importing this file to our Knowledge base, we can see all the Webpanels, Procedures and Structure Data Type that is going to be incorporated to the Knowledge base. And it is going to be a folder, Client folder inside the workflow folder of our Knowledge base explorer.

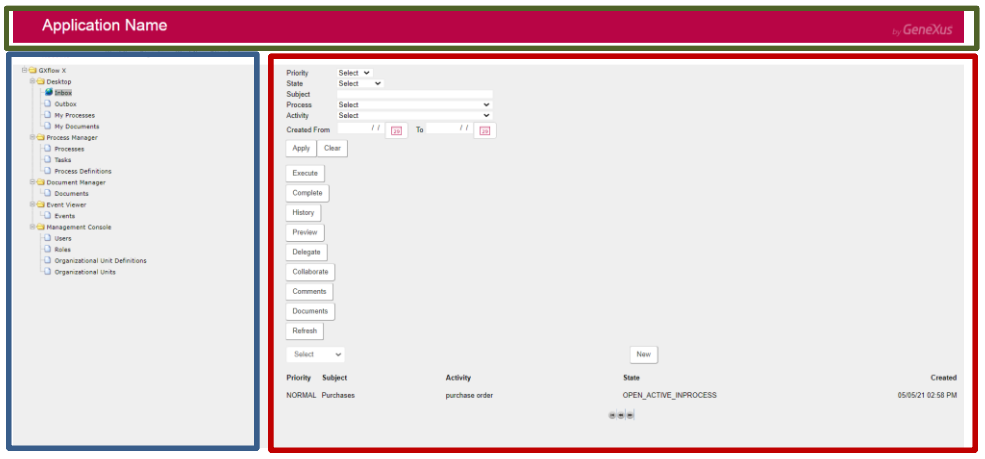
After we import the workflow Custom Client into our Knowledge base, we are going to find a workflow Client folder. Inside this folder are all the Webpanels import. If we run the main object the workflow Client, we can see this workflow Custom Client and execution.



First we have to sign in, so we are going to write the workflow user and then its password. After I sign in, Im here at the workflow Client main object. I will click the inbox link to go to the workflow inbox page. Then here are all the filters, the actions and the grid with all the workitems we have to ejecute and complete. I Will start a new Project selecting processes and then the new button. Here it creates the workitem and the first …. Process. Now i Will execute this workitem. This ask me to upload a document, so i Will create a new document called doc, of the type and will select it from the file sistem. After I select my file, I have confirm and that would be uploaded here. After i complete this task i press the button complete to create the next workitems in the Process Definition. I can see for example the history of the process that shows me all the workitems that have been created

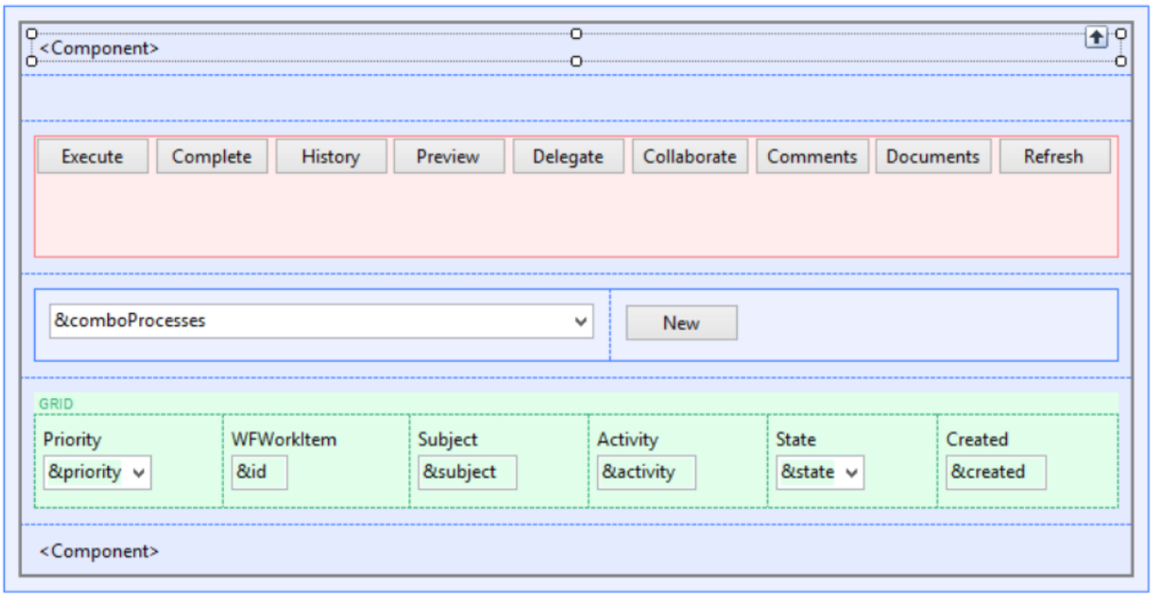
The main object of the workflow Custom Client is our Webpanel called Workflow Client as we have seen during the execution. If click the link inbox is going to appear in the right part of the Webpanel the words inbox code

This page, the workflow Client has three parts. First the header which is taken from the master page of the Webpanel. Then a tree menu with all the functions that are provided by the Custom Client, we can click on those Webpanels links and it will show the selected page at the right part of the panel. In this case I have selected the inbox and it is going to show the workflow inbox workpanel.



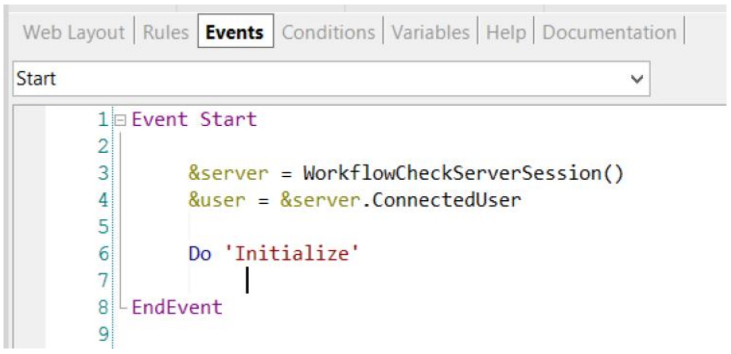
Priority	Subject	Activity	State	Created
NORMAL	Purchases	purchase order	OPEN_ACTIVE_INPROCESS	05/05/21 02:58 PM

The workflow inbox has the workflow filters, then it has a table with all the actions that can be executed from the grid, execute, complete, history, preview, delegate. Then it has a combo box which has all the processes that the user logged in can start. After that was agreed with the workitems that belongs to the workflow inbox of this user. After that it has a Webcomponent with the control of page in this grid



VALIDATE SESION

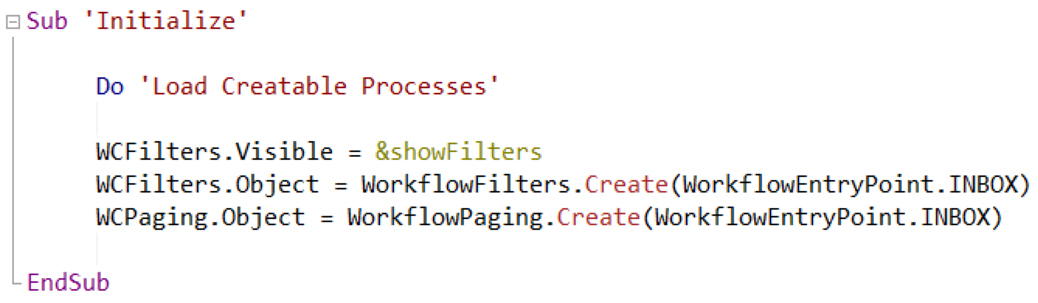
If we see the code of this panel, the first event is the event Start. here the Webpanel will validate the user sesión.



```
Event Start

	&server = WorkflowCheckServerSession()
	&user = &server.ConnectedUser

	Do 'Initialize'

EndEvent
```

Every object of the Custom Client, checks if there's a valid session, and to do so, it use a procedure called WorkflowCheckServerSession. Here we have two variables, the server variable, which is the WorkflowServer datatype; and the user variable, which is of the WorkflowUser datatype.

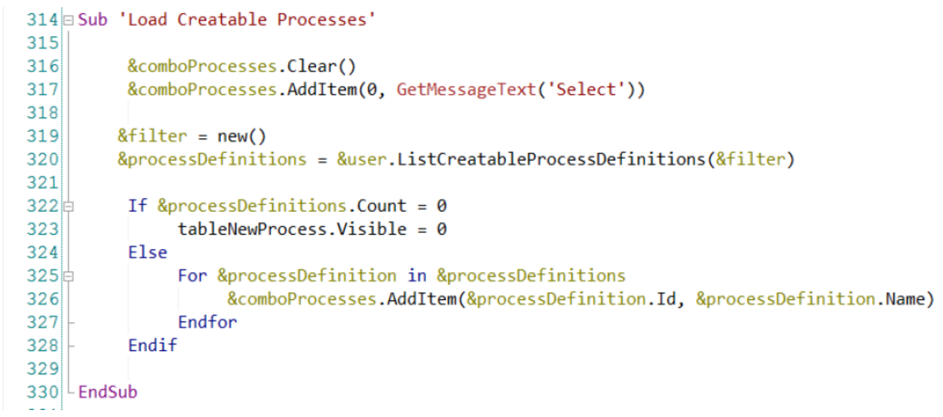
First it's going to load the WorkflowServer variable from the web session, using the WorkflowCheckServerSession procedure, that we are going to see in the next part of the course.



```
Sub 'Initialize'

    Do 'Load Creatable Processes'

    WCFilters.Visible = &showFilters
    WCFilters.Object = WorkflowFilters.Create(WorkflowEntryPoint.INBOX)
    WCPaging.Object = WorkflowPaging.Create(WorkflowEntryPoint.INBOX)

EndSub
```

After that if it has the WorkflowServer loaded it will obtain the connected user from the WorkflowServer. And then it will initialize the webpage. In this case we are initializing the workflow inbox, so it is going to load all the created processes of the combo box that we have seen during the execution.

.



```
Sub 'Load Creatable Processes'

    &comboProcesses.Clear()
    &comboProcesses.AddItem(0, GetMessageText('Select'))

   &filter = new()
   &processDefinitions = &user.ListCreatableProcessDefinitions(&filter)

    If &processDefinitions.Count = 0
        tableNewProcess.Visible = 0
    Else
        For &processDefinition in &processDefinitions
            &comboProcesses.AddItem(&processDefinition.Id, &processDefinition.Name)
        Endfor
    Endif

EndSub
```

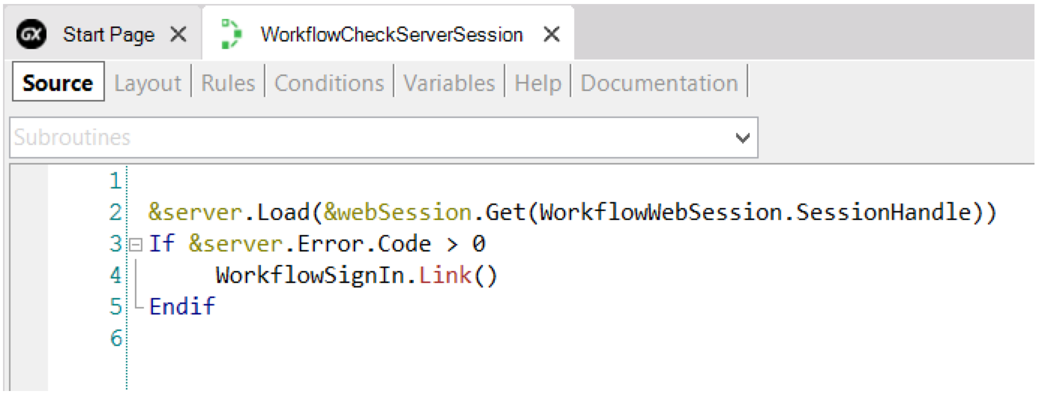
First it clears the components at the first ítem, the selected element, and then the user logged in obtained previousy in the event start and it use the method ListCreateTableProcessDefinition. This method obtains all the process definitions that a user logged in can create in the start. We are going to filter this method, it has a filter parameter, but in this case we are leaving this filter empty, this filter variable is from the WorkflowFilter datatype.

After that if the process definition, which is a list of process definition, is not empty, we are going to iterate this list and load all the process definitions ID and name to the combo box.

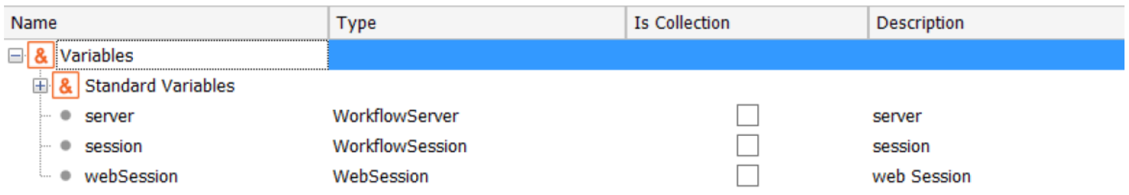
After we load the combo box, we are going to create two WebComponents, first with a WorkflowFilter, which use the workflow filter with component, and the WorkflowPaging.

After we finish this start event, we are going to see the WorkflowCheckServerSession procedure.

This procedures get the session handler id from the websession and load this connection into the WorkflowServer variable. This session was previously saved during the logg in into the websession. If there is an error during this loading session the programm will redirect to the WorkflowSignIn. And after loaded session, the connected user is connected to the WorkflowServer variable.

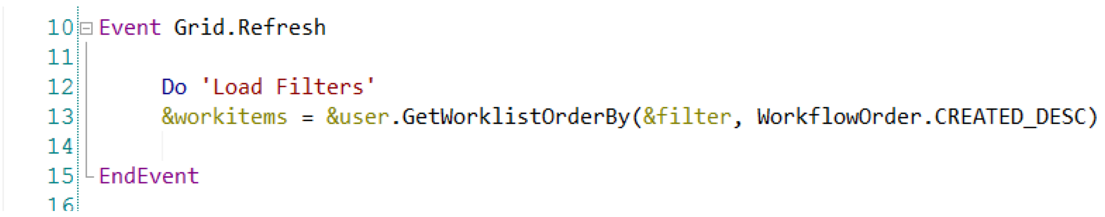


```

&server.Load(&webSession.Get(WorkflowWebSession.SessionHandle))
If &server.Error.Code > 0
      WorkflowSignIn.Link()
Endif

```

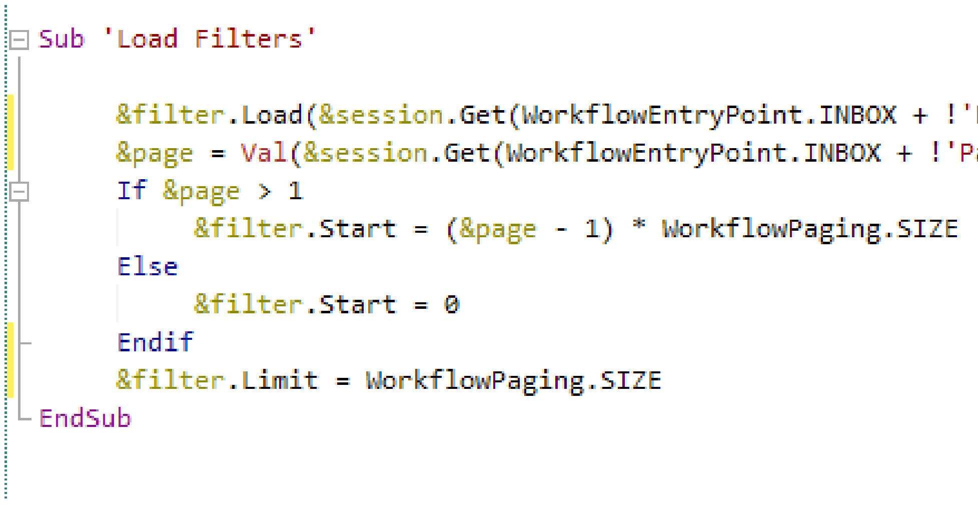


Name	Type	Is Collection	Description
Variables			
Standard Variables			
server	WorkflowServer	☐	server
session	WorkflowSession	☐	session
webSession	WebSession	☐	web Session

The next method at the workflow inbox is the refresh event. This excecute a sub routine called load filters that will bring all the filters from the Websession, that were previously saved in the workflow filter web component. So close at the workflow filter variable, all the workflow filter that were saved at the Websession. After that we obtain from the Websession also, the actual page of the workflow inbox. The filters are in a Webcomponent called workflow filters and it saves the filters sdt into the web session.



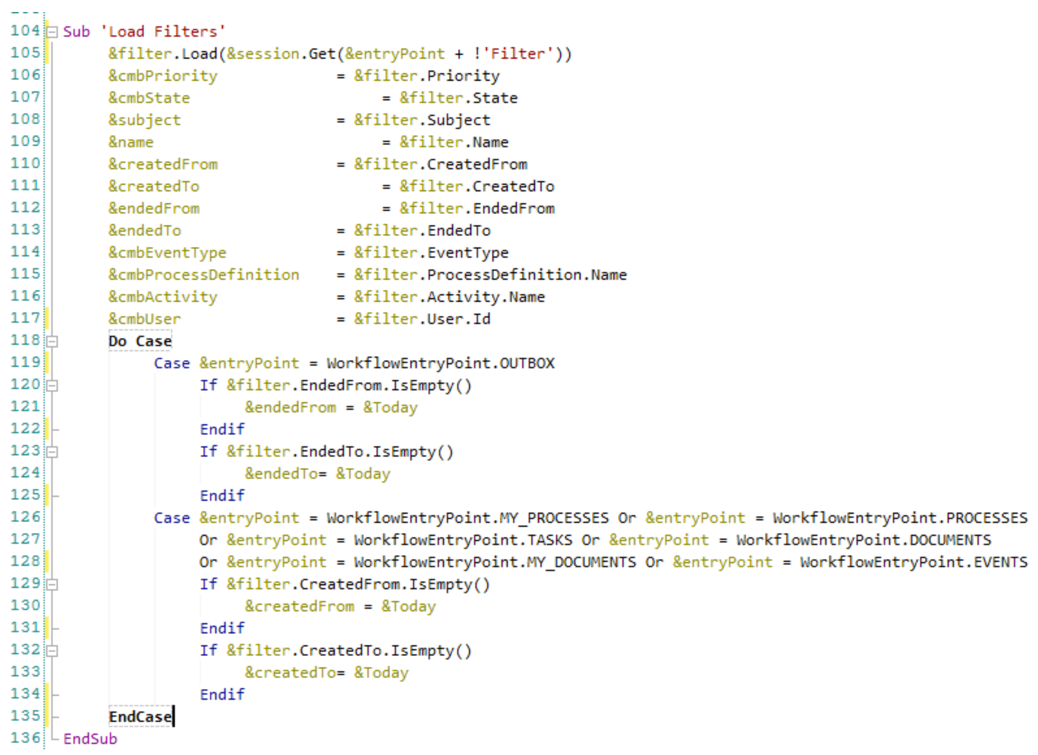
```
Event Grid.Refresh

    Do 'Load Filters'
    &workitems = &user.GetWorklistOrderBy(&filter, WorkflowOrder.CREATED_DESC)

EndEvent
```



```
Sub 'Load Filters'

    &filter.Load(&session.Get(WorkflowEntryPoint.INBOX + !'Filter'))
    &page = Val(&session.Get(WorkflowEntryPoint.INBOX + !'Page'))
    If &page > 1
        &filter.Start = (&page - 1) * WorkflowPaging.SIZE
    Else
        &filter.Start = 0
    Endif
    &filter.Limit = WorkflowPaging.SIZE
EndSub
```

If we see this Webpanel WorkflowFilter, it looks like it has a priority, states, subject, processess, activity, name, user and has two main subroutines.

The first is the one it executes when it’s loading, that loads all the filters workflow data type from the web session, that was previously saved, and it loads all the data into the variables of the panel. In case that this WorkflowFilter is executed from an outbox it loads the today variable into the ended from the variable.



```
Sub 'Load Filters'
    &filter.Load(&session.Get(&entryPoint + !'Filter'))
    &cmbPriority                = &filter.Priority
    &cmbState                       = &filter.State
    &subject                    = &filter.Subject
    &name                           = &filter.Name
    &createdFrom                = &filter.CreatedFrom
    &createdTo                      = &filter.CreatedTo
    &endedFrom                      = &filter.EndedFrom
    &endedTo                    = &filter.EndedTo
    &cmbEventType               = &filter.EventType
    &cmbProcessDefinition       = &filter.ProcessDefinition.Name
    &cmbActivity                = &filter.Activity.Name
    &cmbUser                    = &filter.User.Id
    Do Case
        Case &entryPoint = WorkflowEntryPoint.OUTBOX
            If &filter.EndedFrom.IsEmpty()
                &endedFrom = &Today
            Endif
            If &filter.EndedTo.IsEmpty()
                &endedTo= &Today
            Endif
        Case &entryPoint = WorkflowEntryPoint.MY_PROCESSES Or &entryPoint = WorkflowEntryPoint.PROCESSES
            Or &entryPoint = WorkflowEntryPoint.TASKS Or &entryPoint = WorkflowEntryPoint.DOCUMENTS
            Or &entryPoint = WorkflowEntryPoint.MY_DOCUMENTS Or &entryPoint = WorkflowEntryPoint.EVENTS
            If &filter.CreatedFrom.IsEmpty()
                &createdFrom = &Today
            Endif
            If &filter.CreatedTo.IsEmpty()
                &createdTo= &Today
            Endif
    EndCase
EndSub
```

If it’s called from the my process, or process, task, document, my document and event, it loads the today variable into the created for default values.

After the user completes the information and clicks the apply button, it will execute an event we will call a save filters sub routine. In this subroutine will say all the information loaded by the user in the variables to the filter data type SDT

After load all the information with the filter variable it will load that information into the web session, in order to have it available.

After we load the filters in the refresh event we will use the user variable, previously loaded at the start event, with the method get work list order by. This method Gets the work list of the user, which is the list of work items that are needed to be completed by this user. This method can be filter with information that was entered in the workfilter webcomponent. And also it has another parameter, which is the order that we are going to see the information, in this case we are going to order with the workflow domain and will be defined that, that will be the order with the created date descendant. This will return us a list of workflow work items. We are going to load this list into the workflow work items variable. In the gridload event we are going to iterate this list and for each element of this list we are going to load information of that work item into the variables of the grid and then we are going to execute the gridload method.