

## Custom Client

Neste vídeo analisaremos o Custom Client fornecido por GeneXus

Custom Client é fornecido por GeneXus e pode ser importado para a Knowledge Base para utilizá-lo ou modificá-lo como parte de nosso projeto. Contém todos os Webpanels e procedimentos necessários para implementar muitas funcionalidades do GXflow Standard Client.

Utilizando o Custom Client, você pode executar a caixa de entrada do usuário, visualizar a caixa de saída, gerenciar as definições de processos, workitems, instâncias e até mesmo monitorar a execução.

Se olharmos o código fonte programado nestes Webpanels, podemos ver como são utilizadas as APIs do GXflow para conectar ao motor e obter as informações necessárias em cada caso, e executar diferentes funções para cada objeto.

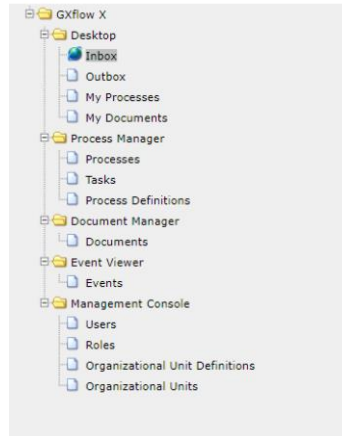
Contém toda a funcionalidade do Custom Client. Neste vídeo veremos alguns Web Panels para entender seu código e como podem ser modificados.

Podemos personalizar o Custom Client por vários motivos, por exemplo, podemos adicionar ou ocultar colunas na caixa de entrada ou em qualquer outra tela. Podemos adicionar ações ou personalizar algum código, alterar a UX e integrar uma parte ou todo o Custom Client em nossa aplicação.

Para ver mais informações sobre Custom Client, podemos seguir o link mostrado a seguir:

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

Custom Client contém algumas pastas, por exemplo: Desktop, Gerenciamento de Processos, Gerenciamento de Documentos, Visualizador de eventos e um Console de gerenciamento.



Na pasta do desktop contém a caixa de entrada, a caixa de saída, meus processos e meus documentos, fornecem as mesmas informações e ações que o Standard Client.

A pasta Gerenciador de processos contém as páginas do processo, a tarefa e as definições de processo, que são as páginas mais utilizadas da seção do gerenciador de processos.

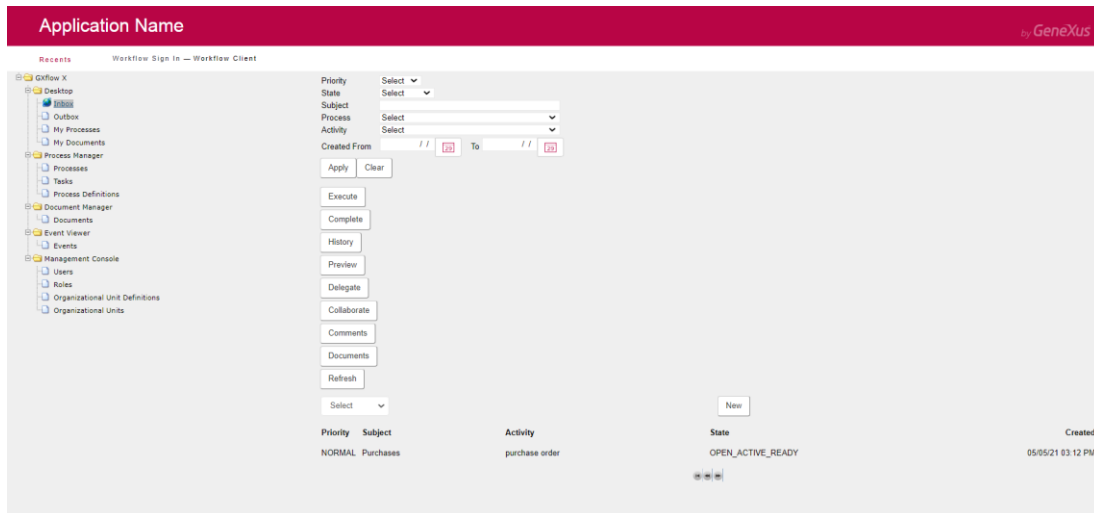
A pasta gerenciador de documentos contém gerenciamento de documentos da mesma maneira que o Standard Client.

A pasta visualizador de eventos contém a página do gerenciador de eventos, da mesma maneira que o Standard Client.

A pasta console de administração contém a administração de usuários, de roles, as definições das unidades organizacionais e as unidades organizacionais, da mesma maneira que o Standard Client.

As estatísticas, o back-end, a configuração do servidor e o gerenciador de licenças não são implementados no exemplo de Custom Client.

O uso principal do Custom Client é incorporar os painéis de GXflow aos nossos projetos utilizando a mesma UX. Por exemplo, este painel apresenta uma caixa de entrada personalizada com uma aparência e comportamento que corresponde ao restante de sua aplicação. O usuário de nossa aplicação não sentirá a diferença com outras partes dela.



Custom Client Inbox

## INBOX

Search Filters

Id  User

Process

Subject

Stage beginning from

Stage beginning until  Date order  Tags Filter

[Clear Filters](#)

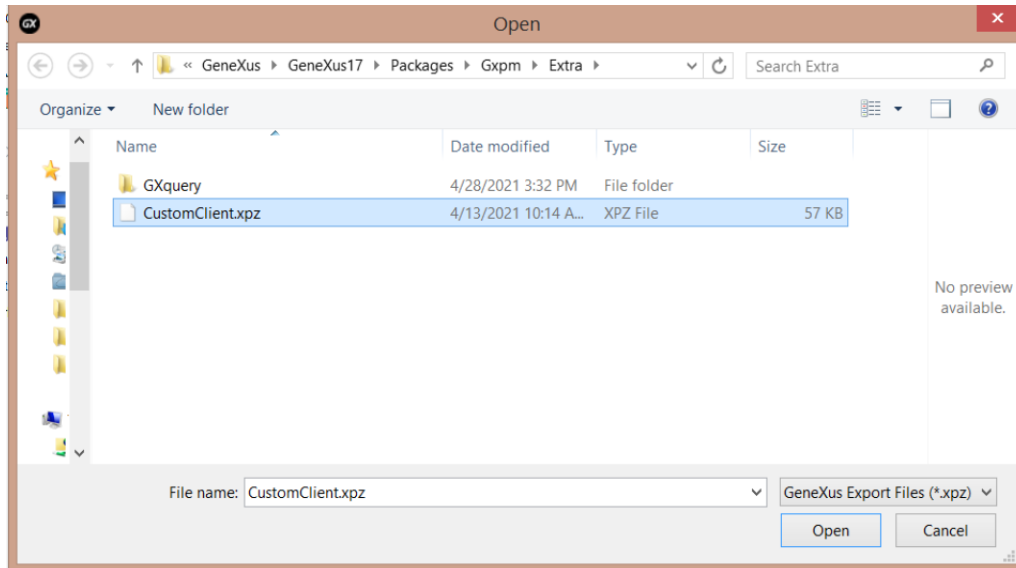
RJDEmo X

	Id	Process	User	Subject	Current Stage	Stage Beginning			
<input type="checkbox"/>	17762	RJETDemo	admin	RJETDemo	Entry	08/10/20 15:28	<a href="#">VIEW</a>	<a href="#">ACT</a>	

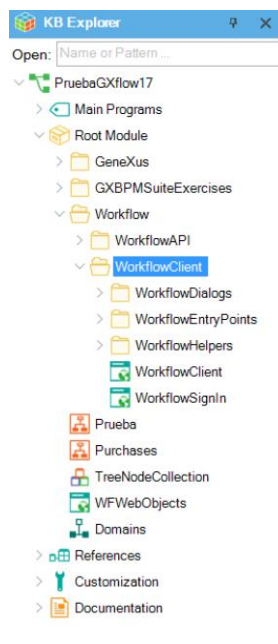
Customized Client Inbox

Existe um arquivo chamado CustomClient.xpz e é distribuído com a instalação de GeneXus dentro de: <diretório de instalação de GeneXus> \ Packages \ GXPM \ Extra.

Antes de importá-lo, verifique se foi criado pelo menos um Business Process Diagram na Knowledge Base. Isto é necessário para garantir que o modelo tenha os tipos de dados Workflow, caso contrário, ocorrerá um erro durante a compilação do projeto.



Ao importar o xpz para nossa Knowledge Base, podemos ver todos os Web panel, Procedimentos e SDTS que serão incorporados à base de conhecimento. Serão encontrados na pasta Workflow Client dentro da pasta Workflow do Knowledge Base Explorer. Se for executado o objeto principal, será possível ver o Custom Client em execução.



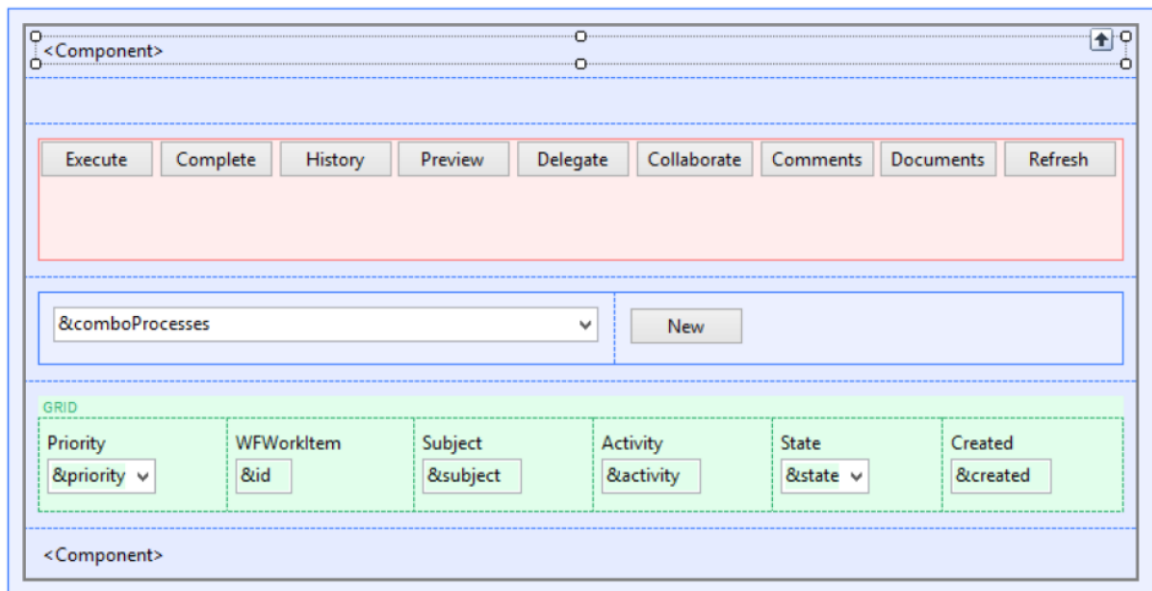
Primeiro precisamos fazer login, então vamos digitar o usuário de Workflow e depois sua senha. Após o login, no objeto principal do Workflow Client, clicará no link da caixa de entrada para ir até ela, então ali estão todos os filtros, as ações e o grid com todos os workitems que temos para executar e concluir. Iniciarei uma nova instância selecionando um processo e então clico no botão novo. Aqui é criado o workitem e uma instância do processo. Agora vou executar este workitem que me solicita carregar um documento, então vou criar um novo documento do tipo Doc e selecioná-lo no sistema de arquivos. Depois de selecionar meu arquivo, confirmei e ele será carregado aqui. Após concluir esta tarefa, pressiono o botão completar para criar os seguintes workitems na definição do processo. Posso ver, por exemplo, o histórico do processo que me mostra todos os itens de trabalho que foram criados.

O objeto principal do Custom Client é o Webpanel chamado Workflow Client, como vimos durante a execução. Se clicamos no link, a caixa de entrada aparecerá na parte direita do Webpanel.

Nesta página, o Workflow Client tem três partes: Primeiro, o Header que é obtido da Master Page do Web panel. Em seguida, um menu em árvore com todas as funções fornecidas pelo Cliente Personalizado, podemos clicar nesses links de Webpanels e nos mostrará a página selecionada na parte direita do panel. Neste caso, selecionei a caixa de entrada e exibirá o Webpanel da caixa de entrada do fluxo de trabalho.



A caixa de entrada tem os filtros do fluxo de trabalho, depois tem uma tabela com todas as ações que podem ser executadas a partir do grid, executar, completar, histórico, pré-visualizar, delegar. Depois tem um combo box que tem todos os processos que podem ser iniciados pelo usuário logado. Posteriormente são formados os workitems que pertencem à caixa de entrada do usuário. Depois disso, tem um Webcomponent com o controle de páginas do grid.



### VALIDATE SESION

Se vemos o código deste panel, o primeiro evento é Start, aqui o Webpanel validará a sessão do usuário.

```

Web Layout | Rules | Events | Conditions | Variables | Help | Documentation |
Start
1 | Event Start
2 |
3 |     &server = WorkflowCheckServerSession()
4 |     &user = &server.ConnectedUser
5 |
6 |     Do 'Initialize'
7 |         |
8 | EndEvent
9 |

```

Cada objeto do Custom Client verifica se existe uma sessão válida e, para isso, utiliza um procedimento chamado WorkflowCheckServerSession. Aqui temos duas variáveis, a variável &server, que é do tipo de dado WorkflowServer, e a variável &user, que é do tipo de dados Workflow User.

Primeiro carregará a variável WorkflowServer a partir da sessão web, usando o procedimento WorkflowCheckServerSession, que veremos na próxima parte do curso.

```

Sub 'Initialize'
    Do 'Load Creatable Processes'

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

```

Depois disso, se tiver o WorkflowServer carregado, obterá o usuário conectado do WorkflowServer e então inicializará a página web. Neste caso estamos inicializando a caixa de entrada do fluxo de trabalho, então serão carregados todos os processos criados do combo box, que vimos durante a execução.

```

314 Sub 'Load Creatable Processes'
315
316     &comboProcesses.Clear()
317     &comboProcesses.AddItem(0, GetMessageText('Select'))
318
319     &filter = new()
320     &processDefinitions = &user.ListCreatableProcessDefinitions(&filter)
321
322     If &processDefinitions.Count = 0
323         tableNewProcess.Visible = 0
324     Else
325         For &processDefinition in &processDefinitions
326             &comboProcesses.AddItem(&processDefinition.Id, &processDefinition.Name)
327         Endfor
328     Endif
329
330 EndSub

```

Primeiro, limpa os elementos do combo e é adicionado o valor padrão, em seguida, utilizando os dados do usuário que foram obtidos no Evento Start, é utilizado o método ListCreateTableProcessDefinition. Este método obtém todas as definições de processo que um usuário conectado pode iniciar.

O método ListCreateTableProcessDefinition possui um parâmetro de filtro do tipo de dado WorkflowFilter, mas neste caso o deixamos vazio.

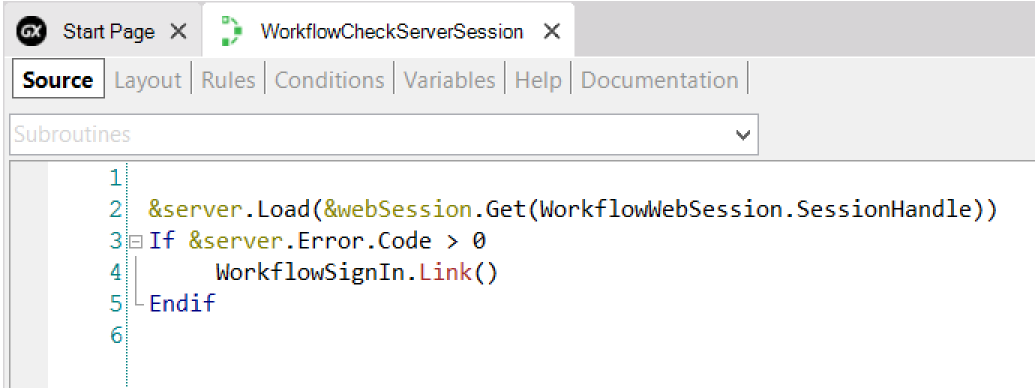
Se a variável &ProcessDefinition, que é uma lista de definição de processos, não estiver vazia, vamos percorrê-la para carregar todos os ID e o nome das definições do processo no combo box

Depois de carregar o combo box, criaremos dois Web Components, primeiro com um WorkflowFilter e o segundo com um WorkflowPaging. O primeiro Web Component serve para filtros e o segundo para paginação do grid.

Agora veremos o procedimento WorkflowCheckServerSession.

Este procedimento obtém o Id da sessão a partir da WebSession e carrega esta conexão na variável Server do tipo de dado WorkflowServer. Esta sessão foi salva na WebSession quando foi feito o login. Se ao carregar a sessão for detectado um erro, será redirecionado para o WorkflowSignIn.

Quando a sessão está carregada, o usuário logado é aquele que se utiliza para conectar com o WorkflowServer.



The screenshot shows a development environment with two main components: a source code editor and a variables table.

**Source Code Editor:** The editor displays the following code in a subroutine:

```
1
2 &server.Load(&webSession.Get(WorkflowWebSession.SessionHandle))
3 If &server.Error.Code > 0
4     WorkflowSignIn.Link()
5 Endif
6
```

**Variables Table:** Below the code editor is a table with the following columns: Name, Type, Is Collection, and Description.

Name	Type	Is Collection	Description
& Variables			
& Standard Variables			
• server	WorkflowServer	<input type="checkbox"/>	server
• session	WorkflowSession	<input type="checkbox"/>	session
• webSession	WebSession	<input type="checkbox"/>	web Session



## REFRESH & FILTERS

O seguinte método na caixa de entrada é o evento Refresh do Grid. Este executa uma sub-rotina chamada "Load Filters" que trará todos os filtros da WebSession. São carregados na variável &filter todos os filtros que foram recuperados da sessão. Depois disso, obtemos também da WebSession, a página atual da caixa de entrada.

Os filtros estão em um Web Component chamado WorkflowFilters. Este salva o SDT de filtros na WebSession.

```
10 Event Grid.Refresh
11
12     Do 'Load Filters'
13         &workitems = &user.GetWorklistOrderBy(&filter, WorkflowOrder.CREATED_DESC)
14
15 EndEvent
16
```

```
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
```

Se observamos o Web panel WorkflowFilters, tem uma prioridade, estados, assunto, processo, atividade, nome, usuário e tem duas sub-rotinas principais.

A primeira é executada quando é feita a carga, recupera a sessão que havia guardado anteriormente todos os filtros de tipo de dado e atribui os valores às variáveis do panel. Caso o painel seja chamado a partir da bandeja de saída, atribui a data do dia à variável Ended from.

```

104 Sub 'Load Filters'
105   &filter.Load(&session.Get(&entryPoint + '!Filter'))
106   &cmbPriority      = &filter.Priority
107   &cmbState        = &filter.State
108   &subject         = &filter.Subject
109   &name            = &filter.Name
110   &createdFrom     = &filter.CreatedFrom
111   &createdTo       = &filter.CreatedTo
112   &endedFrom       = &filter.EndedFrom
113   &endedTo         = &filter.EndedTo
114   &cmbEventType    = &filter.EventType
115   &cmbProcessDefinition = &filter.ProcessDefinition.Name
116   &cmbActivity     = &filter.Activity.Name
117   &cmbUser         = &filter.User.Id
118   Do Case
119     Case &entryPoint = WorkflowEntryPoint.OUTBOX
120       If &filter.EndedFrom.IsEmpty()
121         &endedFrom = &Today
122       Endif
123       If &filter.EndedTo.IsEmpty()
124         &endedTo = &Today
125       Endif
126     Case &entryPoint = WorkflowEntryPoint.MY_PROCESSES Or &entryPoint = WorkflowEntryPoint.PROCESSES
127       Or &entryPoint = WorkflowEntryPoint.TASKS Or &entryPoint = WorkflowEntryPoint.DOCUMENTS
128       Or &entryPoint = WorkflowEntryPoint.MY_DOCUMENTS Or &entryPoint = WorkflowEntryPoint.EVENTS
129       If &filter.CreatedFrom.IsEmpty()
130         &createdFrom = &Today
131       Endif
132       If &filter.CreatedTo.IsEmpty()
133         &createdTo = &Today
134       Endif
135     EndCase
136 EndSub

```

Se for chamado a partir de My processes, task, documents, my documents e events, atribui a data do dia à variável Created from.

Depois do usuário completar as informações e clicar no botão Aplicar, executará um evento de onde chamaremos uma sub-rotina para salvar filtros. Nesta sub-rotina, serão atribuídas todas as informações carregadas pelo usuário nas variáveis do filtro em um SDT.

Após carregar todas as informações com a variável de filtro, carregará essa informação na sessão web, para tê-la disponível

Uma vez carregados os filtros no evento Refresh, chamaremos o método “GetWorkListOrderBy” utilizando a variável de usuário que obtivemos no evento Start. Este método retorna uma coleção de tarefas, esta lista inclui todos os workitems que precisam ser concluídos pelo usuário. Este método pode ser filtrado com os valores que o usuário selecionou no Webcomponent workfilter. Também recebe um parâmetro que nos permite indicar a ordem em que queremos receber as informações, neste caso utilizaremos o domínio “WorkFlowOrder”, e indicaremos que ordene por data de criação em ordem decrescente. Isto nos retornará a lista de workitems. Vamos carregar esta lista nas variáveis correspondentes.

Posteriormente no evento Grid.Load, iteramos sobre a lista e para cada elemento da lista carregamos as informações nas variáveis do grid e executamos o método Gridl.load(). Isto fará com que todos os workitems sejam carregados no grid correspondente.