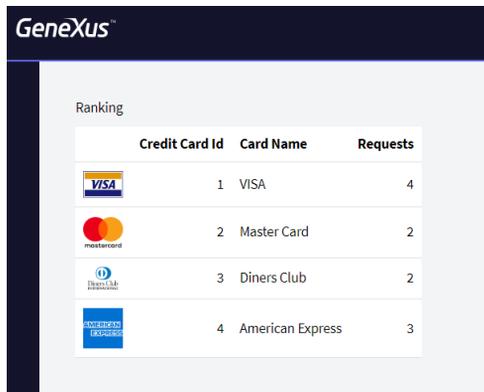# Loading a Collection SDT in a Web Panel Grid

GeneXus™

Many times we need to display the content of a data collection, and a web panel is an option to do so.

Let's see an example of loading a collection SDT in the grid of a web panel, in order to be able to perform some action on its lines.

Example: Credit cards Ranking

Let's suppose then that the travel agency offers the possibility to register customer requests for the issuance of different credit cards with which the agency works. Thus, we want to implement a ranking of credit cards, according to the number of applications registered in 2020. Since this is a ranking, the information should be presented in order from highest to lowest by that amount.

Example: Credit cards Ranking

Let's consider the transaction design we are looking at and the structured data type that will allow us to load the collection of all cards, each with its corresponding number of requests registered in the year 2020.

Example: Credit cards Ranking

```
SDTCardRequests from CreditCard
{
    SDTCardRequestsItem
    {
        CreditCardId
        CreditCardImage
        CreditCardName
        CreditCardRequests = count(CardRequestDate,CardRequestDate.Year()=2020)
    }
}
```

Name	Type	Description	Is Collection
⊟ 📋 SDTCardRequests		SDTCard Requests	☑
⊟ 🔲 SDTCardRequestsItem			
• CreditCardId	Attribute:CreditCardId	Credit Card Id	☐
• CreditCardImage	Attribute:CreditCardImage		☐
• CreditCardName	Attribute:CreditCardName	Card Name	☐
• CreditCardRequests	Numeric(4.0)	Requests	☐

We open GeneXus:

How do we load the SDT? With a Data Provider
This Data Provider will load each credit card with its corresponding
number of applications registered in 2020, which explains the calculation
condition stated in the formula.

Example: Credit cards Ranking


```
Event Start
    &CardRequests = DPRequests()
    &CardRequests.Sort("[CreditCardRequests]")
Endevent
```

Name	Type	Is Collection
Variables		
Standard Variables		
CardRequests	SDTCardRequests	☐

Then, from the Start event of a web panel we receive the output of this Data Provider, and we do it in the &CardRequests variable that has been defined based on the SDTCardRequests data type.

OK, but for it to really be a ranking, we must order it from highest to lowest by the number of requests registered, and for that we use the Sort method.

We still have to effectively display this content in a grid.

For this, from the web panel form, we are going to insert the variable &CardRequests. Since it is a structured collection variable, GeneXus will automatically associate a grid with it.
But it is a variable, and we already know that variables in a web panel are input data. Therefore, so that its content cannot be modified, we will indicate that it is read only.

Note that we can also choose whether we want its content to be displayed in a Freestyle grid. If this option is not checked, it will be displayed in a common grid.

At runtime:

Now let's think about this:

Is there any difference between this grid loaded from a structured collection variable and another grid loaded with variables in the usual way? No, in any case it is a grid without a base table.

Let's see it at runtime. We press F5:

All right, we have our ranking in place. Now we want to solve the following:
We need to be able to select one of these credit cards, and have a PDF list of registered applications displayed in order to obtain it.

Let's return to GeneXus.

We have already defined the CardRequestList list

```
Parm(in:CreditCardId, in:&CreditCardName);

output_file("List", "pdf");
```



```
print printblock1

For each CardRequest
    print printblock2
endfor
```

To make it easier, we have already defined the CardRequestList procedure that receives in its Parm rule the identifier and name of the credit card, and displays the registered requests.

Allow Selection and Allow Hovering Properties

Ranking			
GRID			
	Credit Card Id	Card Name	Requests
	&CardRequests.item(0).CreditCardId	&CardRequests.item(0).CreditCardName	&CardRequests.item(0).CreditCardRequests

Behavior	
Sortable	True
Allow Drop	False
Allow Drag	False
Notify Context Chan	False
Allow Collapsing	False
Allow Selection	**True**
Allow Hovering	True

OK, let's go back to our web panel.

We need to be able to select a grid line. So let's look at the properties of the grid, and enable the Allow Selection property with True value.

This also enables the Allow Hovering property that allows highlighting with color as we pass through the lines.
It's worth mentioning that the default colors can be modified through the applied Theme, editing the classes assigned to the Selected Row Class and Hover Row Class properties at the Grid class level;

OK. But we also need an event that captures the selected line and allows performing an action. This event is the OnlineActivate event associated with the grid.

We then go to the events sector, and type Event, and the name of the grid, which in this case is Grid1, period, and select the OnlineActivate event.

What we want to do is to call the list that we have seen previously, and pass it by parameter the ID and Name of the card selected in the line.

CurrentItem Property

```
Event Start
     &CardRequests = DPRequests()
     &CardRequests.Sort("[CreditCardRequests]")
Endevent

Event Grid1.OnLineActivate
     &CreditCardId = &CardRequests.CurrentItem.CreditCardId
     &CreditCardName = &CardRequests.CurrentItem.CreditCardName
     CardRequestLists.Popup(&CreditCardId, &CreditCardName)
EndEvent
```


How can we do this?

As the grid shows the loading of a collection, we must position ourselves on the selected item of the collection in order to obtain the ID and the name. For that, we are going to use the CurrentItem property.

So, we are going to define the &CreditCardId variable... we see that it was automatically based on the attribute of the same name, and we are going to assign to the collection variable &CardRequests period, CurrentItem, period, CreditCardId, to obtain in this way the ID of the card selected in the line.

We will do the same to obtain the Name.

We define the &CreditCardName variable, verify that it is based on the attribute of the same name, and define the corresponding assignment.

Lastly, we will call the defined list to open as a popup window, and we pass the necessary parameters: the variable &CreditCardId, and the variable &CreditCardName.

Let's see it at runtime. We press F5:

Collection SDT in the grid of a Panel object

- ➢ The collection is received in the Refresh event.

- ➢ The variable is inserted in the form and GeneXus will automatically create the grid with the selected fields.

- ➢ To perform any action on a selected line, we must define an event and use the CurrentItem property

- ➢ This event should be defined as the default action of the grid (Default Action property).

Finally, it should be mentioned that if you were working with an object of Panel type, the process would be similar:

In the Refresh event, the collection is received in a variable.
This variable is inserted in the form and G will automatically create the grid associated with the selected fields.
In case you also want to perform some action on a selected line, you must define an event and use the CurrentItem property
Finally, this event must be defined as the default action of the grid, using the Default Action property.

GeneXus™

training.genexus.com
wiki.genexus.com