

Using patterns

*GeneXus*<sup>™</sup>

## Pattern Work With for Web

Application Name

by GeneXus

Recents Country — Countries

Countries

Q Name

+ INSERT

Id	Name		
6	Argentina	UPDATE	DELETE
1	Brazil	UPDATE	DELETE
9	Chile	UPDATE	DELETE
3	China	UPDATE	DELETE
2	France	UPDATE	DELETE
7	Jamaica	UPDATE	DELETE
5	Japan	UPDATE	DELETE
10	Kingdom of Saudi Arabia	UPDATE	DELETE
11	Mexico	UPDATE	DELETE
8	Uruguay	UPDATE	DELETE

Patterns allow us to empower applications by easily adding new features.

When we apply a pattern, GeneXus creates all the necessary objects to provide the desired behavior without the need for us to program them.

For example, let's suppose that even though we have the Countries transaction to enter, change and delete countries... we also want an attractive page to query countries and show all the existing countries, filter by country name, support paging, etc.

Application Name by GeneXus

Recents Country — Countries

Countries


🔍 Name

+ INSERT

Id	Name		
6	Argentina	UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE

RecentsCountries — Country

**Country**

Id	0
Name	<input type="text"/>
City Id	0
Flag	

**City**

Id	Name
0	<input type="text"/>

Also, it should enable us to go to the Countries transaction to add a new country...

Application Name by GeneXus

Recents Country — Countries

Countries

Q Name


+ INSERT

Id	Name	UPDATE	DELETE
6	Argentina	UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE
		UPDATE	DELETE

Application Name by GeneXus

RecentsCountries — Country

Country

Id	6
Name	Argentina
City Id	1
Flag	

City

Id	Name
x 1	Buenos Aires

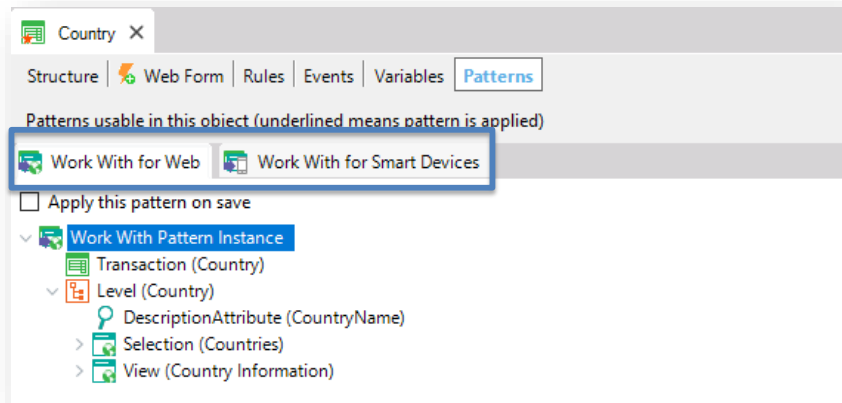
...change an existing country...



## Patterns

**Work With**

for Web  
for Smart Devices

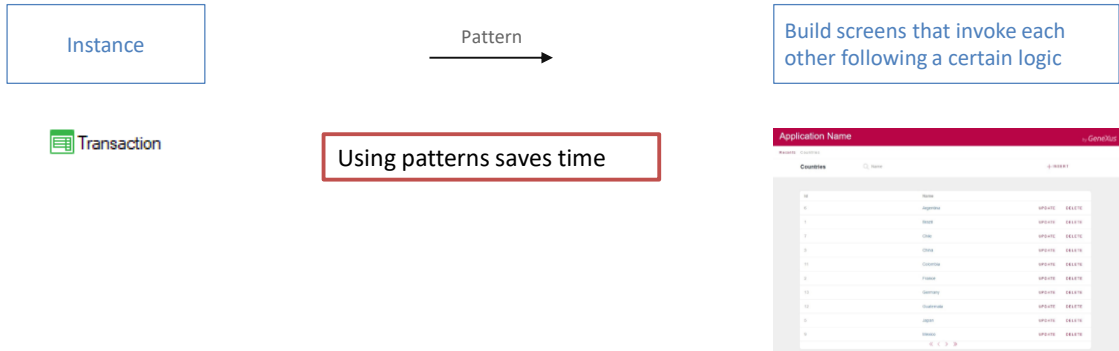


To quickly implement this, we have a pattern called “Work With” and we will show how to apply it to the Country transaction.

This pattern is available for web applications, such as the application in the example, or for Smart Device applications.

In the Patterns tab of the Country transaction, we can see both patterns.

Why are they called patterns?

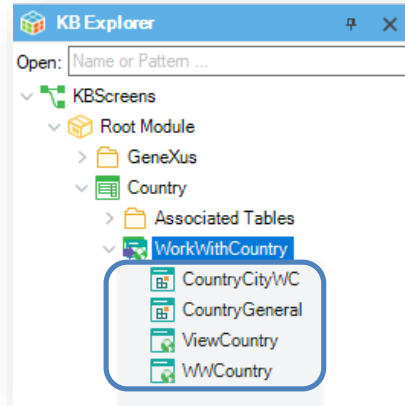
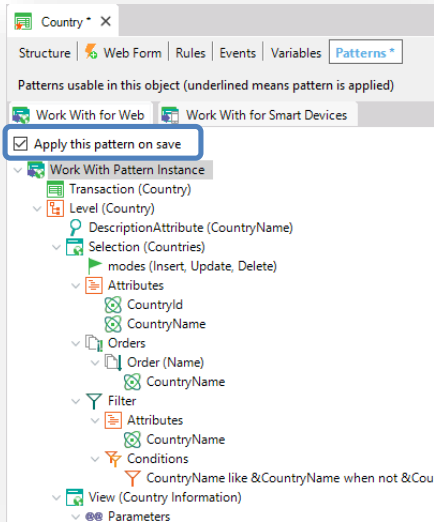


Why are they called patterns? Because from one instance, in this case from a transaction object, it builds a series of screens that invoke each other following a certain logic, which for this particular pattern has to do with presenting information in a hierarchy, where first all the info of the corresponding table is listed and it is possible to work with each element, either just to show it, or to change it as well.

This is an extremely common pattern of interaction, so not having to implement it manually saves us a lot of time.

## How to obtain it

- In the Country transaction:



**Automatically generated by GeneXus**

Now we will see how to apply the Work With pattern for Web to the Country transaction.

In the Country transaction, we select the “Patterns” section, choose the Work With tab, click on “Apply this pattern on save” and save.

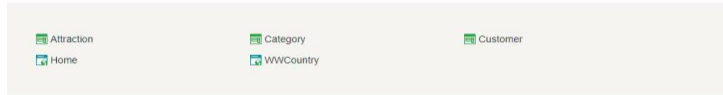
Now, if we place the Country transaction, we can see that below the transaction there are several objects that were created by GeneXus when the Work With pattern was applied.



## DEMO



## Browse Web Objects



[ DEMO: <https://youtu.be/XuzVrb6TWDE> ]

Now we press F5 to run the application and look at what has been automatically generated just by selecting an option and saving.

First of all, note that a link is displayed with the text “Work With Country” for us to work with a wider range of features.

We click on this link...

...and see that a page is opened to show all the countries that we have entered.

For each line with a country there are 2 actions available: Edit the details of the country in the line... we see that the transaction is opened and allows us to change the information related to the selected country. Let's add a city for France and confirm.

This other action clearly offers us the option to delete the country in the line.

And the 'INSERT' action also allows inserting a new country. Running it opens the Countries transaction, ready for us to add a country and all its related data:

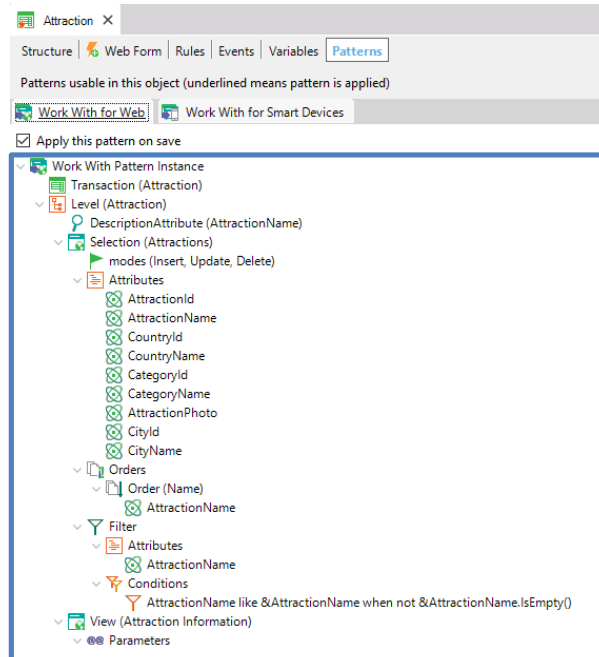
We enter “United States”... and a city, New York... We confirm.

Note that we can search by country name: if we type “F” it shows all the countries that begin with this letter. In this case, we only have France.

Note that the country name has a link. If we click on it, all the details of the selected country are displayed in the first tab and another tab labeled City shows the list of cities that belong to this country.

We choose to go back to Work With Countries.

## Pattern instance



We easily applied the Work With pattern to the Country transaction and now we will apply it to the Attraction transaction.

We go back to GeneXus, open the Attraction transaction, open the “Patterns” section and in the Work With for Web tab we select the option “Apply this pattern on save”.

Upon saving, all the necessary objects are generated to implement all the features that we’ve seen, in this case, to work with Attractions.

We confirm that they have been generated, and press F5.

Note that just like in the previous case, we’re offered to “work with attractions” and from there the transaction is called.

We run “Work With Attractions”...

Again, we see that if we change the screen size, the information displayed is adjusted. In this case, instead of showing all the Attraction attributes, only the name is displayed.

Here we have the same query features that we saw for “Work With Countries”... and we will use it to enter more tourist attractions.

Let's insert the Great Wall... in China... we select the path to the image... and indicate that it is in Beijing.

Now we will add the Eiffel Tower.

We type “Eiffel Tower” ... It's in France... and it is a monument. We select the path to the image... and indicate that the Eiffel Tower is in Paris.

Now, suppose that we are asked, in addition to filtering by attraction name, to provide the ability to filter the attractions in a certain country.

Let's see how to add another filter to "Work With Attractions"... Another request is that the grid shouldn't display the country or city codes... or the category... We go back to GeneXus.

So far, we have only selected "Apply this pattern on save" and we've seen all the features that are automatically generated... what we haven't seen so far is this tree of configurable values.

Let's take a quick look at this "Selection" node that also says "Attractions", because below it are all the configurable options for the "Work With Attractions" element that will be generated.

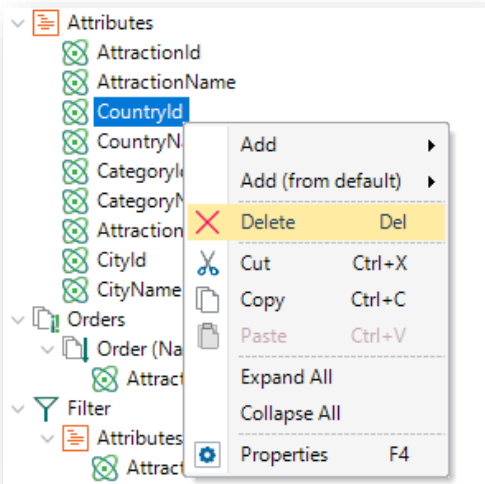
For instance, note that here we can see the operations that will be available to update the database, invoking the transaction to do so.

If you don't want to offer some of them, you can customize them in the properties window.

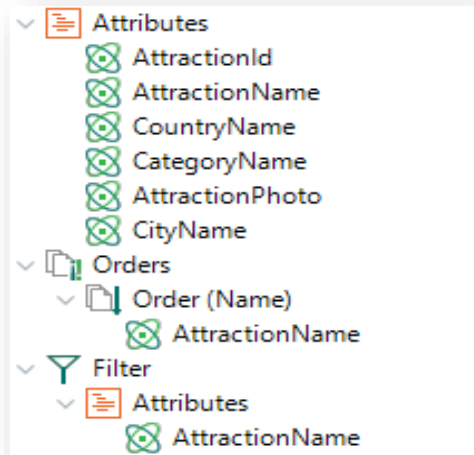
Right below is the Attributes node which contains the attributes that will be included in the "work with" grid.

## Working with the instance

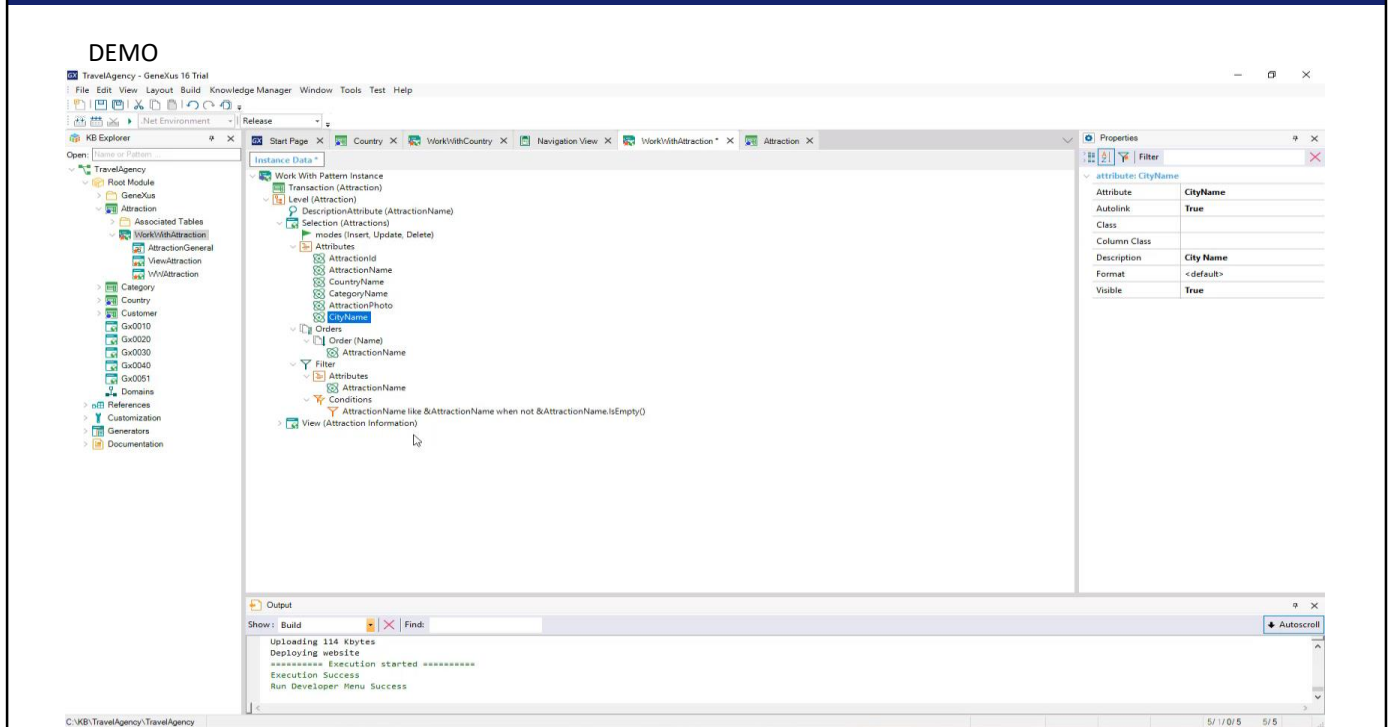
## BEFORE



## AFTER



Since we were asked not to display the country or city identifier in each line, nor the category, we will remove them.



[ DEMO: <https://youtu.be/5RFOW4Q63ZQ> ]

We have also been asked to make it possible to filter all the attractions in a certain country. To do so, under “Filter”, we right-click on the “Attributes” node, and “Select Attributes...”.

In this selection dialog, we choose CountryName.

We are asked if we want the filter condition to be defined and we select Yes.

This automatic filtering condition is generated over the grid's data, which will add a field outside the grid (it will be a variable) for the user to enter characters in order to display only the countries that contain them.

We press F5 to see the result of the customizations we’ve made.

We see that the main filter by attraction name continues to be displayed over the grid, as before (also, the grid no longer displays the country, city and category identifiers). Now a column has been added to the left, which can be hidden and displayed again, and where all the extra filters added to the pattern will be displayed. In this case, there is only one: Country Name. If we press "F", we see that only the attributes of France are displayed.

If we also type “E” in Name the only data item that matches both conditions is the Eiffel Tower.

Note that if we type “E” in the country filter: “E”, there are no records to display that match these conditions.

And if we indicate that the country name starts with a “C” but don’t enter any requirements for the attraction name, the Great Wall of China is displayed.

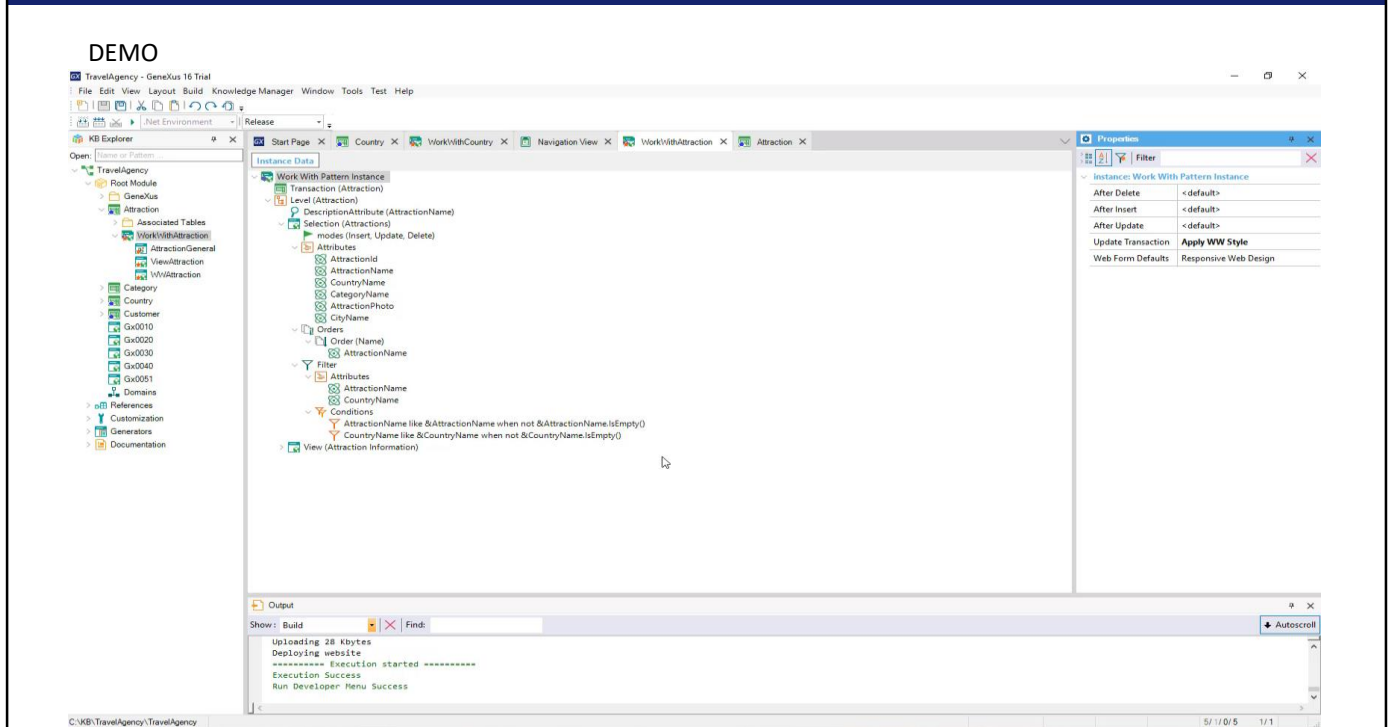
Now suppose that we are asked to make it possible to order the data in the grid, whether in alphabetical order by attraction name as we’ve been doing so far, or by country name. If we click on the column by which we want to order.

We see that it is already done, and that it even allows us to order in descending order.

That's why we wouldn't have to make any changes to the pattern. However, clicking on the column only orders the data that had already been loaded to the grid.

In this case, we only have 3 attractions in the database, but if we had 20, not all of them would be displayed. As we've said, the grid supports paging [go back to the countries ppt with paging shown at the beginning]. This means that only an X number of records are displayed at once. When the screen is opened, the first X are displayed, and it is possible to show the next X, and so on.

Therefore, the records displayed on each page will depend on the total ordering. To order all the records before loading them in the pages, we must do something in the pattern.



[ DEMO: <https://youtu.be/wDQYh4VgYng> ]

We go back to GeneXus... and in this Orders node, we can see the ordering by attraction name is automatically provided.

We right-click on "Orders" and select "Add" / "Order".

We give this order a name, in the properties section, such as: Country.

We right-click on the node of the new order and select "Add" / "Attribute".

We select the CountryName attribute and press F5 to see what we're offered.

Here, it indicates that the information is being ordered by Name; that is to say, by attraction name.











If now we want to do it by Country: the results will be displayed in alphabetical order by country.

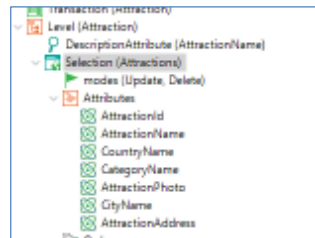
Application Name GeneXus

Records Attractions Name

HIDE FILTERS

Ordered By: Country + INSERT

Md	Name	Country Name	Category Name	Photo	City Name		
2	British Museum	England	Museum		London	UPDATE	DELETE
6	The National Gallery	England	Museum		London	UPDATE	DELETE
5	Musée d'Orsay	France	Museum		Paris	UPDATE	DELETE
1	Musée de Louvre	France	Museum		Paris	UPDATE	DELETE
13	Tour Eiffel	France	Monument		Paris	UPDATE	DELETE
14	Notre Dame	France	Monument		Paris	UPDATE	DELETE
15	Arc de Triomphe	France	Monument		Paris	UPDATE	DELETE
16	Basilique du Sacré-Coeur	France	Monument		Paris	UPDATE	DELETE
19	Rijksmuseum	Holland	Museum		Amsterdam	UPDATE	DELETE
12	Van Gogh	Holland	Museum		Amsterdam	UPDATE	DELETE



Properties

Filter

selection Selection (Attractions)

Caption	Attractions
Description	
Is Main	False
Master Page	<default>
Paging	
Rows per page	<custom>
Custom Rows	
Paging Mode	<default>
Show Current Page	<default>



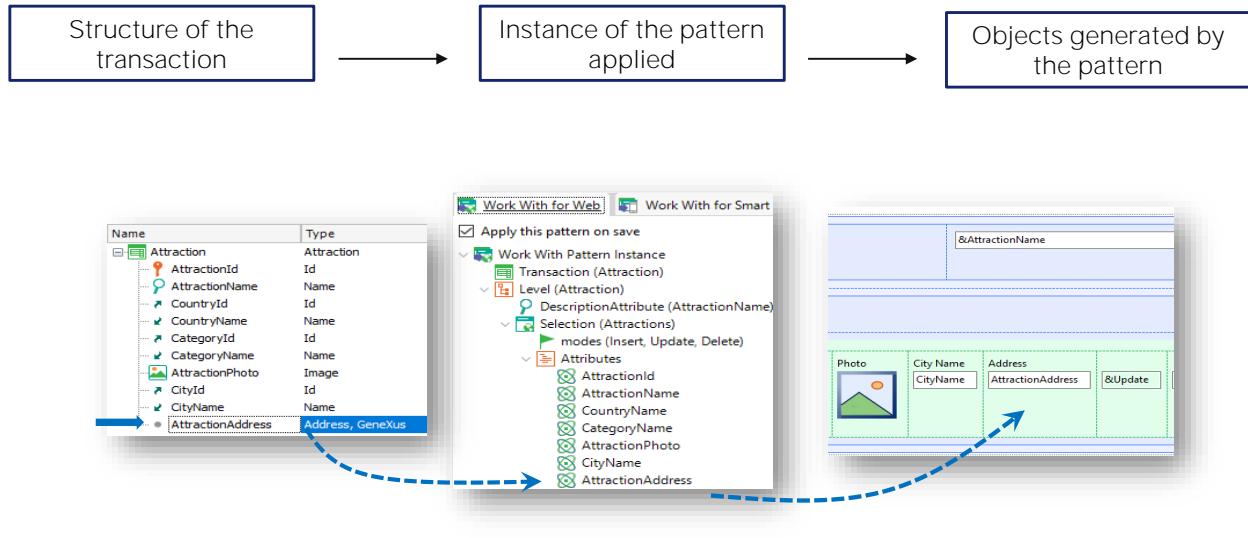
Even if there were more than one page of attractions, this sorting is made on the total number of attractions that have been entered, and not only those that are being shown on the current page. This happens when we click on the name of the row. This way, only the data of the current page is sorted.

We can configure the number of records to be displayed per page in a very simple way. All we have to do is open the Pattern section of the transaction, select "Selection" and set the "Rows per page" property with the "Custom" option. A new property called "Custom Rows" will be displayed; there we have to enter the number of records to be shown per page.



## Maintaining dynamism

- **Example:** New attribute AttractionAddress



Let's talk about something else:

What happens if, after applying the pattern, attributes are added to or deleted from the transaction structure?

Once the Work With pattern has been applied to a transaction, a "dynamic relationship" (synchronization) is created between the transaction structure, the applied pattern instance and the generated objects.

This means that if we add a new attribute to the transaction structure, it will be automatically shown in the pattern instance, and it will be added in the objects generated by the pattern. For example, the object that implements the screen with the grid displayed at runtime.

Therefore, it will be shown when the application is run [press F5]. Since we have added an attribute, we need to reorganize.

Likewise, if an attribute is deleted from the transaction structure (let's delete `AttractionAddress`), it will be automatically deleted from the pattern instance applied and will be removed from the GeneXus objects generated. Let's reorganize once again.

## DEMO

The screenshot shows the GeneXus IDE interface. The main window displays a table of the 'Attraction' pattern structure. The table has columns for Name, Type, Description, Formula, and Nullable. The 'Attraction' type is highlighted in blue. The 'Attraction' type is defined with the following attributes:

Name	Type	Description	Formula	Nullable
AttractionId	Id	Attraction Id		No
AttractionName	Name	Attraction Name		No
CountryId	Id	Country Id		No
CountryName	Name	Country Name		No
CategoryId	Id	Category Id		Yes
CategoryName	Name	Category Name		No
AttractionPhoto	Image	Attraction Photo		No
CityId	Id	City Id		No
CityName	Name	City Name		Yes

The Properties window on the right shows the 'Transaction1.web: Attraction' properties, including Description, Last Modified Date, Logically Deleted Attribute, Name, and Type.

The Output window at the bottom shows the following messages:

```

Show | Build | Find:
-----
Uploading 204 Kbytes
Deploying website
***** Execution started *****
Execution Success
Run Developer Menu Success
  
```

[ DEMO: [https://youtu.be/d-7elSwJ5\\_o](https://youtu.be/d-7elSwJ5_o) ]

When is this dynamic relationship lost?

For example, if we open one of the objects generated by the pattern [WWAttraction] and edit its form (we add a control or change something; for example, delete the attraction's photo), the dynamic relationship between the instance and the generated object is lost. That is to say, if we add the AttractionAddress attribute again to the transaction structure, it is added to the pattern structure, but not to the generated object. Let's remove it.

The same happens if we change the events, rules, or any other section of the object generated by the pattern.

So, losing the existing dynamic relationship means that the definitions made later in the pattern instance will no longer be applied to the generated objects, and they will have to be defined manually. For this reason, changing the objects generated by the pattern is not recommended.

Can the dynamic relationship be recovered? Yes, by opening the object that lost this relationship in edit mode, through the Apply Default options in the menu. Note that we can recover the dynamic relationship of the form, or of all the object's parts (Form, events, etc).

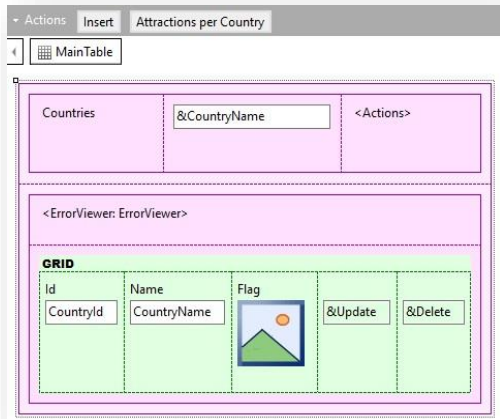
Let's retrieve the form that the pattern builds by default.

Note that the photo is displayed again.

## Maintaining dynamism

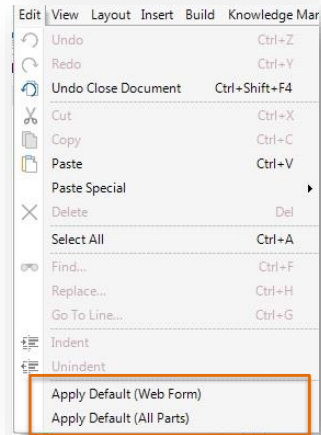
Is dynamism lost?

**Yes**, when the objects generated are edited.



Is it possible to recover it?

**Yes**, through the Edit menu:



## General Settings

The screenshot illustrates the configuration of the 'Insert' property in GeneXus. It shows three windows: 'Preferences', 'Pattern Settings', and 'Properties'. The 'Preferences' window is set to 'TravelAgency' > '.Net Environment' > 'Patterns' > 'Work With for Web'. The 'Pattern Settings' window shows 'Standard Actions' > 'Insert' selected. The 'Properties' window shows the 'Insert: Insert' property with 'Enabled by Default' set to 'True'.

Insert: Insert	
Caption	GXM_insert
Tooltip	GXM_insert
Enabled by Default	True
Default Condition	
Style	
Image	ActionInsert
Disabled Image	ActionDisabled
Disabled Class	
Button Class	BtnAdd
In Grid Class	

Full version only

If now we want the "work with attractions" screen to no longer offer the possibility to insert a new attraction:

We open the pattern instance. Note that the Insert property is included in the Modes node properties.

As we can see, the possible values for this property are True, False and Default. As we saw at runtime, the Default value is True.

So as not to offer the possibility to insert a new attraction from the screen, we will only have to select False. We try it by pressing F5.

The option to insert new attractions is no longer displayed.

Where is the "Default" value of the Insert property defined? As for the values of the Update, Delete, Display, and Export properties, what's their purpose?

Suppose that we wanted all the "work with" elements (attractions, countries, categories if we applied it, customers as well) not to offer the possibility to make insertions. We have two options: open each transaction and in their pattern instance change the value of the Insert property as we just did for the Attraction, or go to where this "default" value is set and change it to False once; this will affect all the work with elements that still have "default" as their value.

In the GeneXus Trial version, that location is not accessible. In the Full version, it is in the Preferences window, below Patterns / Work With for Web.

The values defined here are used to initialize the properties of each pattern, when the instances are created.

Modification of properties in a specific or general instance

Example: Not allowing the entry of countries (in its instance)

Work With for Web | Work With for Smart Devices

- Apply this pattern on save
- Work With Pattern Instance
  - Transaction (Country)
  - Level (Country)
    - DescriptionAttribute (CountryName)
    - Selection (Countries)
      - modes **Insert, Update, Delete**
      - Attributes
        - CountryId
        - CountryName
      - Orders
      - Filter
      - View (Country Information)

Properties

Filter




modes: Ins: false, Upd: default, Del: default, Dis: de...

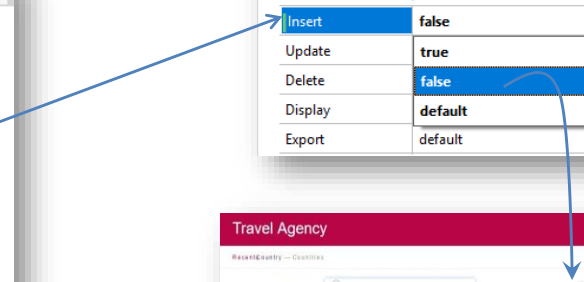
Insert	false
Update	true
Delete	false
Display	default
Export	default

Travel Agency

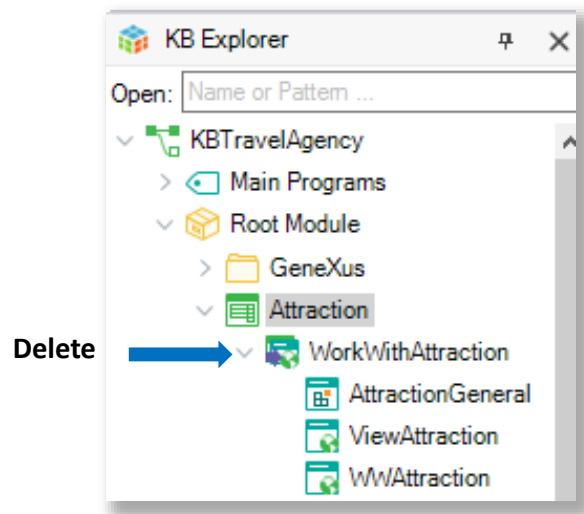
Reset/Empty - Countries

Countries

id	Name	Flag	UPDATE	DELETE
4	Brazil		UPDATE	DELETE
6	China		UPDATE	DELETE
2	France		UPDATE	DELETE



## Deleting the pattern

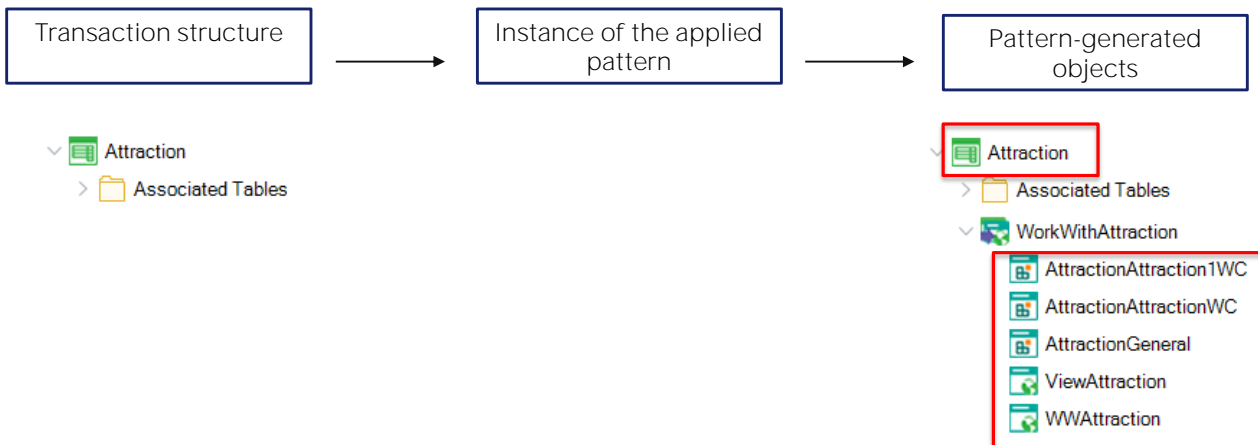


Another question that we can ask ourselves is if we can undo the application of the pattern to a transaction; that is to say, if we can delete all the objects and features of the pattern as if it had never been applied.

To undo the application of the pattern, we must select the instance below the transaction icon and delete it.

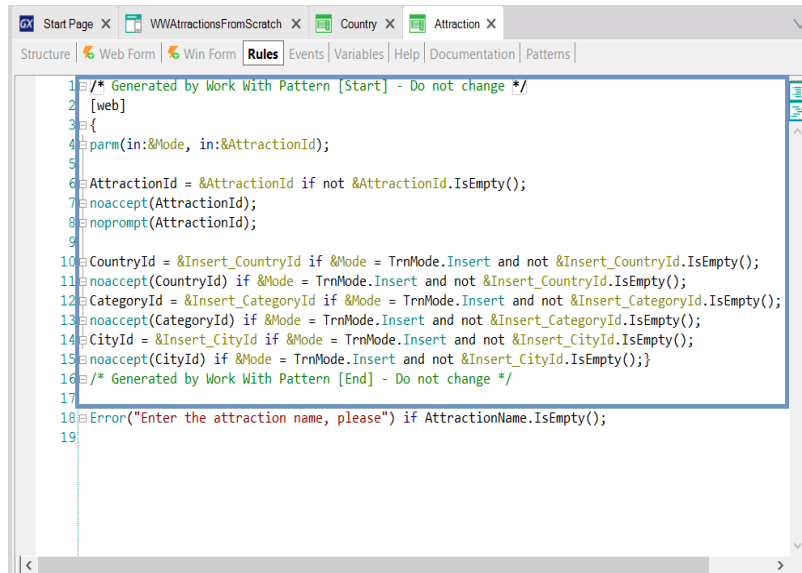
This deletes all the generated objects and automatically clears the box "Apply this pattern on Save" in the Patterns tab of the transaction. To apply the pattern again, we only need to check the box again and click on Save.





Applying the pattern to the transaction not only generates the objects that implement the new screens provided by the pattern, but also modifies the transaction's behavior so that it can follow this new interaction flow.



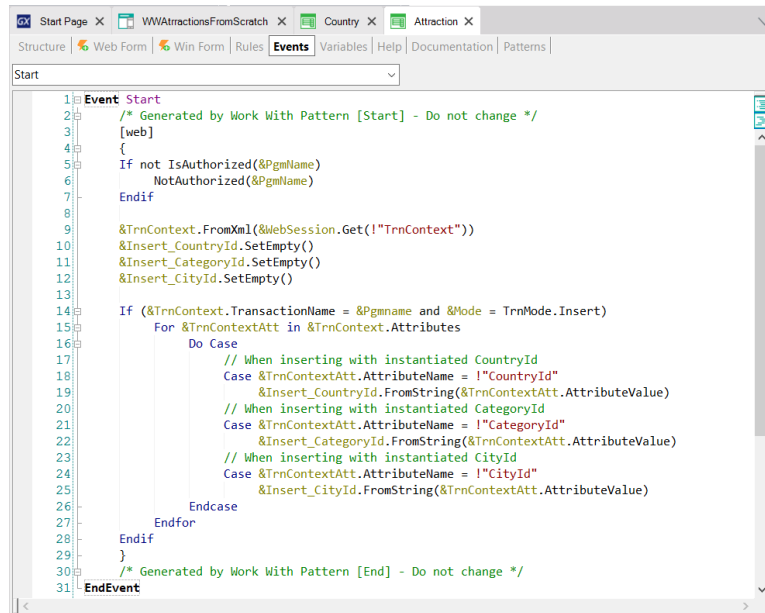


```
1 /* Generated by Work With Pattern [Start] - Do not change */
2 [web]
3 {
4   parm(in:&Mode, in:&AttractionId);
5
6   AttractionId = &AttractionId if not &AttractionId.IsEmpty();
7   noaccept(AttractionId);
8   noprompt(AttractionId);
9
10  CountryId = &Insert_CountryId if &Mode = TrnMode.Insert and not &Insert_CountryId.IsEmpty();
11  noaccept(CountryId) if &Mode = TrnMode.Insert and not &Insert_CountryId.IsEmpty();
12  CategoryId = &Insert_CategoryId if &Mode = TrnMode.Insert and not &Insert_CategoryId.IsEmpty();
13  noaccept(CategoryId) if &Mode = TrnMode.Insert and not &Insert_CategoryId.IsEmpty();
14  CityId = &Insert_CityId if &Mode = TrnMode.Insert and not &Insert_CityId.IsEmpty();
15  noaccept(CityId) if &Mode = TrnMode.Insert and not &Insert_CityId.IsEmpty();
16 /* Generated by Work With Pattern [End] - Do not change */
17
18 Error("Enter the attraction name, please") if AttractionName.IsEmpty();
19
```

For example, let's look at the Rules section. As we can see, it has added this whole set of rules, leaving comments to warn the developer not to change them. Here is the Error rule that we had declared before. That one is left intact, outside of that block of rules added by the pattern.

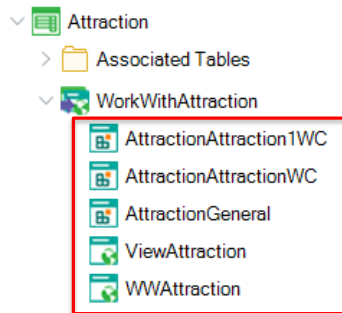
If we look at the first rule in the code block added by the web pattern, it's the rule that allows you to define the parameters that the object will exchange with whoever invokes it. Something you might have noticed is that once we apply the pattern, the transaction no longer appears in the Developer Menu. It can no longer be directly invoked for the user to enter one attraction after another, changing some of their values, or deleting others. The reason is that the transaction will now always be invoked by passing parameters to it, i.e. telling it in which mode it will be opened and instantiated according to which identifier value. We will dwell on this later, when we study communication between objects.

Likewise...



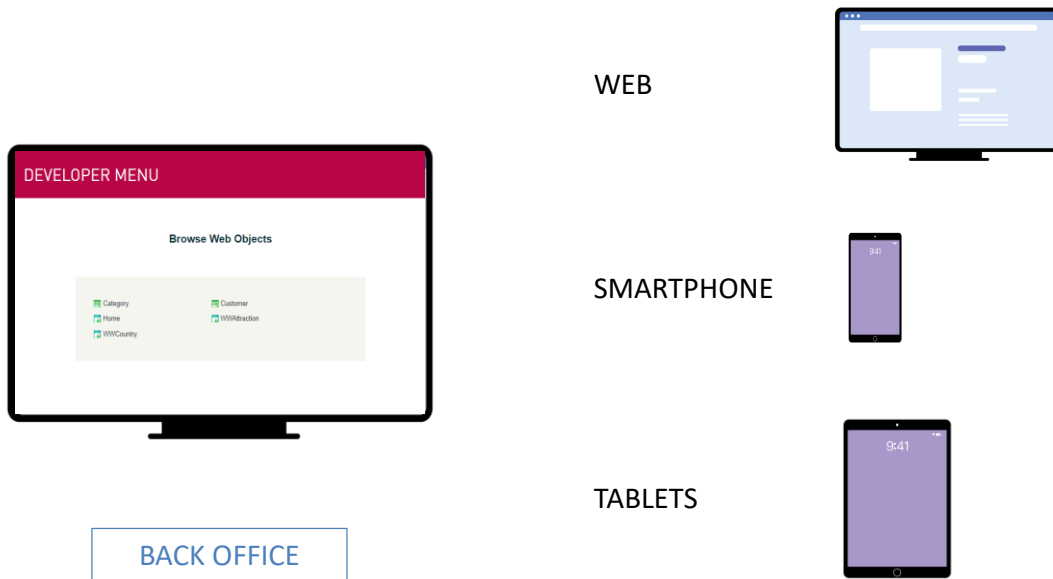
```
1 Event Start
2 /* Generated by Work With Pattern [Start] - Do not change */
3 [web]
4 {
5   If not IsAuthorized(&PgmName)
6     NotAuthorized(&PgmName)
7   Endif
8
9   &TrnContext.FromXml(&WebSession.Get(!"TrnContext"))
10  &Insert_CountryId.SetEmpty()
11  &Insert_CategoryId.SetEmpty()
12  &Insert_CityId.SetEmpty()
13
14  If (&TrnContext.TransactionName = &Pgmname and &Mode = TrnMode.Insert)
15    For &TrnContextAtt in &TrnContext.Attributes
16      Do Case
17        // When inserting with instantiated CountryId
18        Case &TrnContextAtt.AttributeName = !"CountryId"
19          &Insert_CountryId.FromString(&TrnContextAtt.AttributeValue)
20        // When inserting with instantiated CategoryId
21        Case &TrnContextAtt.AttributeName = !"CategoryId"
22          &Insert_CategoryId.FromString(&TrnContextAtt.AttributeValue)
23        // When inserting with instantiated CityId
24        Case &TrnContextAtt.AttributeName = !"CityId"
25          &Insert_CityId.FromString(&TrnContextAtt.AttributeValue)
26      Endcase
27    Endfor
28  Endif
29 }
30 /* Generated by Work With Pattern [End] - Do not change */
31 EndEvent
```

... If we go to the events section that we haven't mentioned yet for the transactions, we see that the pattern has incorporated quite a bit of code.



## Web panels / Web components

The objects corresponding to the screens that organize and display the Work With element information are called Web Panels (or a specialization corresponding to a little piece of a web panel, which is called web component). We will study this object later, as we need to know how to develop our own application screens, which will be an important part of the UI. GeneXus has implemented them for us here, but we must learn how to do it on our own.



First with transactions and now with the screens created by the Work With pattern applied to those transactions, we can see how the back office of our application begins to take shape; that is, the specialized application to handle the data of the other application, which we will release for the end customer. Remember that the latter are called customer-facing applications.

In general, the back office application is only a web app, but it is not mandatory. It may also run on other types of devices, such as phones or tablets. The solution is similar, using the other Work With pattern.

Applying the Work With pattern to the Attraction transaction:

The behavior of the Attraction transaction was modified

New objects —web panels – were created that implement the Work With screens

The Developer Menu was modified.

Pressing F5:

The Database didn't have to be reorganized

Every new or modified object was programmed in the corresponding language and compiled, uploading everything to the cloud

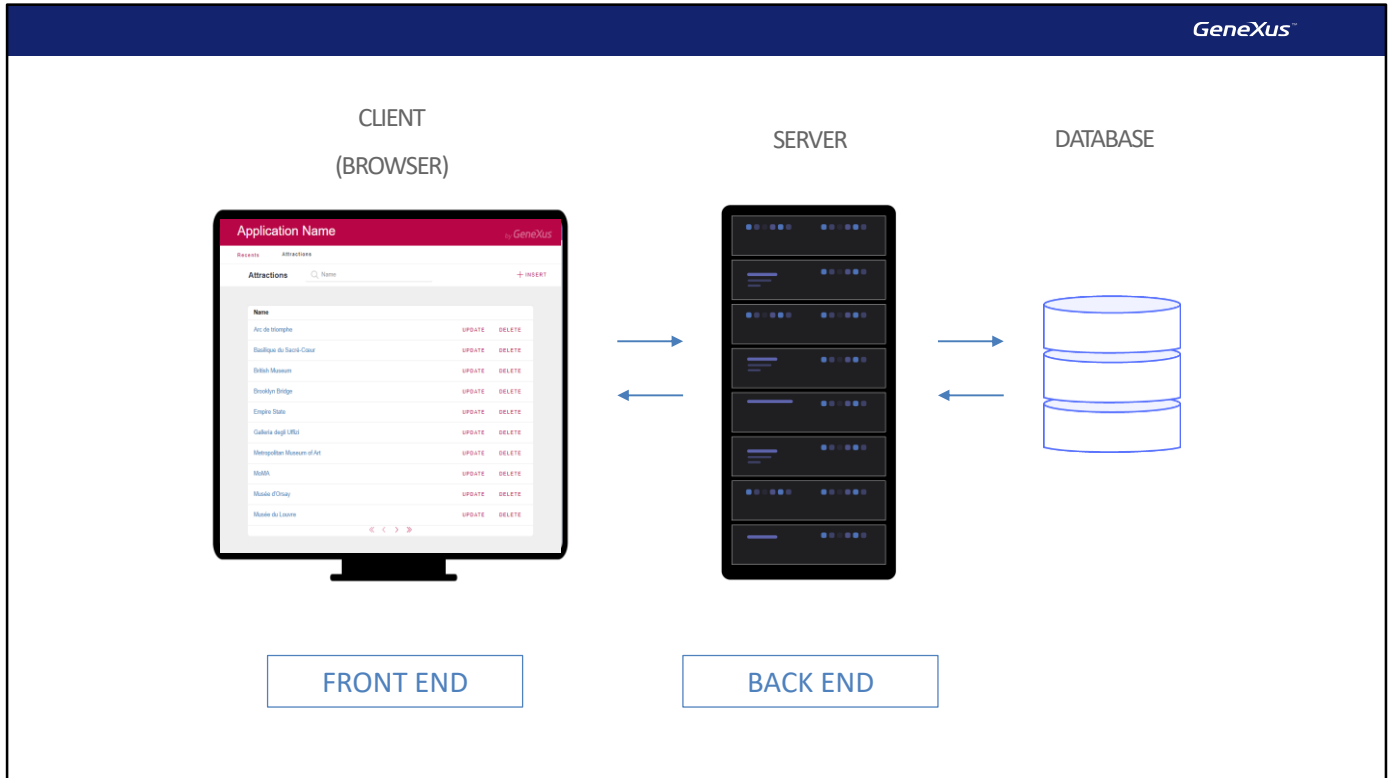
GeneXus ran the modified Developer Menu.

If we look back at what happened when we applied the Work With pattern to Attraction:

- Some parts related to the behavior of the Attraction transaction have been modified.
- New objects have been created; in particular, web panels that implement the Work With screens.
- The Developer Menu has been modified, so as not to invoke the transaction anymore, and instead invoke the associated Work With component. To achieve this, some more objects were modified/created, which do not matter now.

Next, when pressing F5:

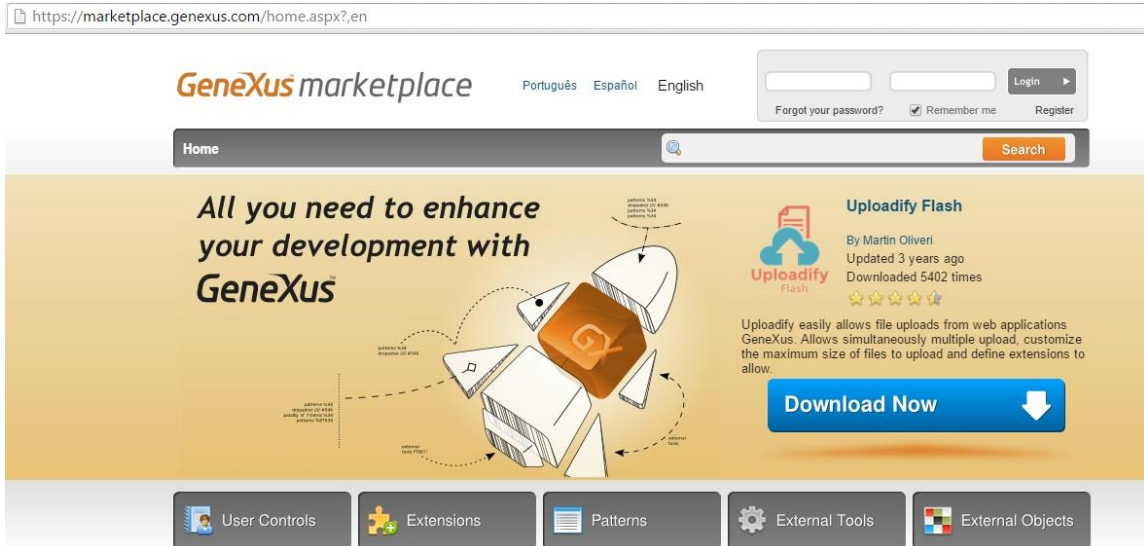
- GeneXus understood that there were no structural changes in the data, so it did not have to reorganize the database.
- Also, GeneXus programmed each new or modified object in the target language and compiled it, uploading it all to the cloud server.
- Lastly, GeneXus ran the modified Developer Menu.



Then, when the user runs WWAttraction:

The front end part of the program (that is, the part of the object that controls the User Interface, reacts to actions performed, communicates with the server), which asks the server for the information to be displayed on the grid (all attractions), is loaded in the client's browser. The back end part of the program requests the information from the database, and returns it to the browser, which displays the loaded page.

## Other patterns...



The screenshot displays the GeneXus marketplace website. At the top, there is a navigation bar with the GeneXus logo and language options: Português, Español, and English. A search bar is located on the right side of the navigation bar. Below the navigation bar, the main content area features a large banner for "All you need to enhance your development with GeneXus". The banner includes a 3D illustration of a cube with various icons on its faces, representing different patterns. To the right of the banner, there is a featured pattern called "Uploadify Flash" by Martin Oliveri, which has been updated 3 years ago and downloaded 5402 times. A "Download Now" button is visible below the featured pattern. At the bottom of the page, there is a horizontal menu with five categories: User Controls, Extensions, Patterns, External Tools, and External Objects.

Lastly, we go back to GeneXus and save the changes in GeneXus Server.

In addition, there are other patterns that generate useful functionalities and they can be viewed in this page.

# GeneXus™

**The power of doing.**

Videos

[training.genexus.com](http://training.genexus.com)

Documentation

[wiki.genexus.com](http://wiki.genexus.com)

Certifications

[training.genexus.com/certifications](http://training.genexus.com/certifications)