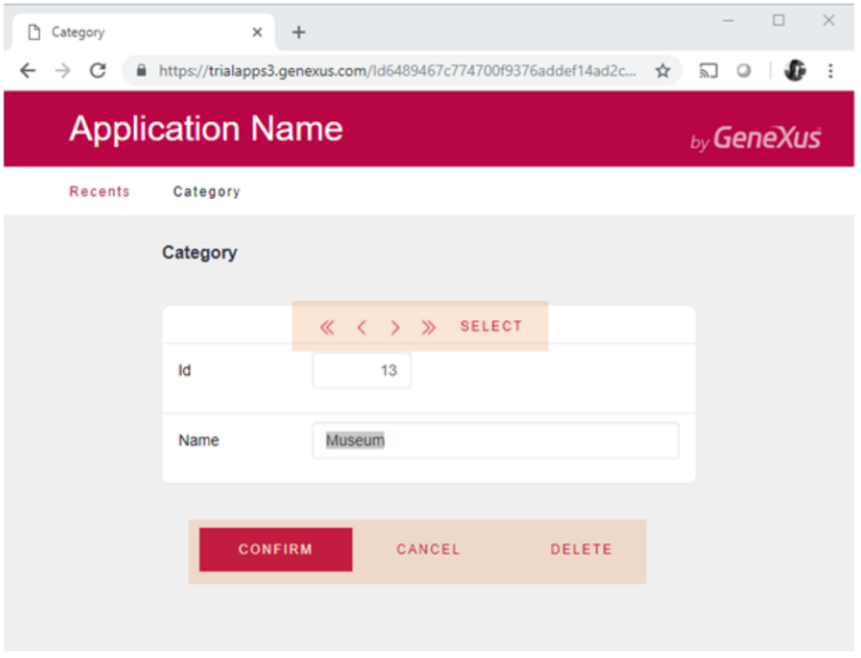
Menus and Master Pages

Interactive Screens

GeneXus 16

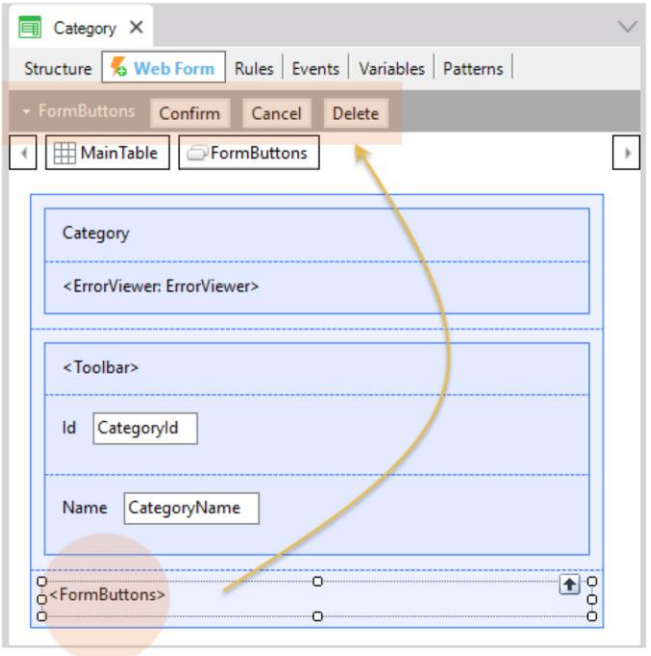
Action Groups



If we open any transaction at runtime –for example, Category– we see on the screen the navigation buttons before the attributes, and below them the buttons to confirm, cancel, or delete the category with which we are working every time. Where is this configured in the transaction object?

Action Groups

- Action Group: UI element used to group controls (buttons, images, text blocks, and variables)
- It helps to easily create a toolbar
- It allows inserting several controls within the same cell of a Responsive Table. It will be displayed in a single row.

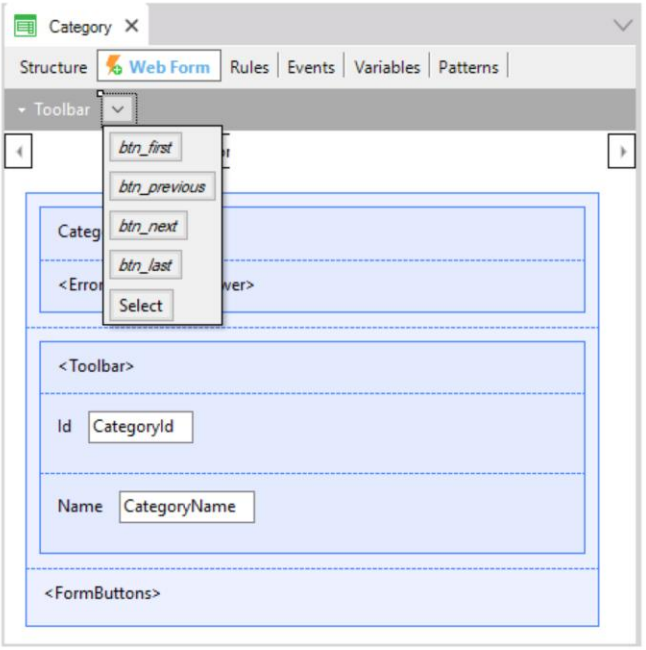


If we open the Web form, we can see that in addition to the attributes that make up the transaction structure, GeneXus adds two controls known as Action Groups.

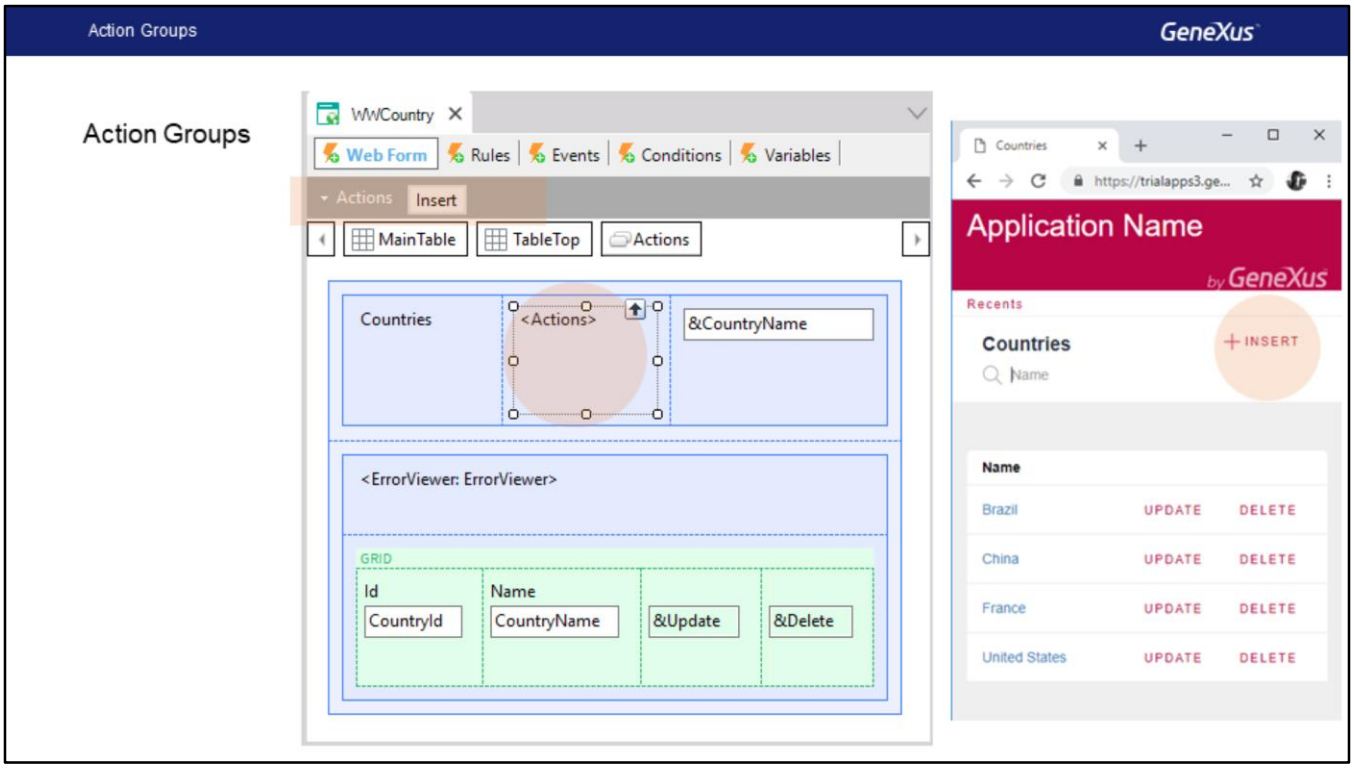
In the image, we're editing the control <FormButtons>. Note that when we do so, above the Web form, in the section called Action Bar, we see the controls that make up this action group. They will correspond to the buttons to confirm, cancel or delete the category.

Action Groups

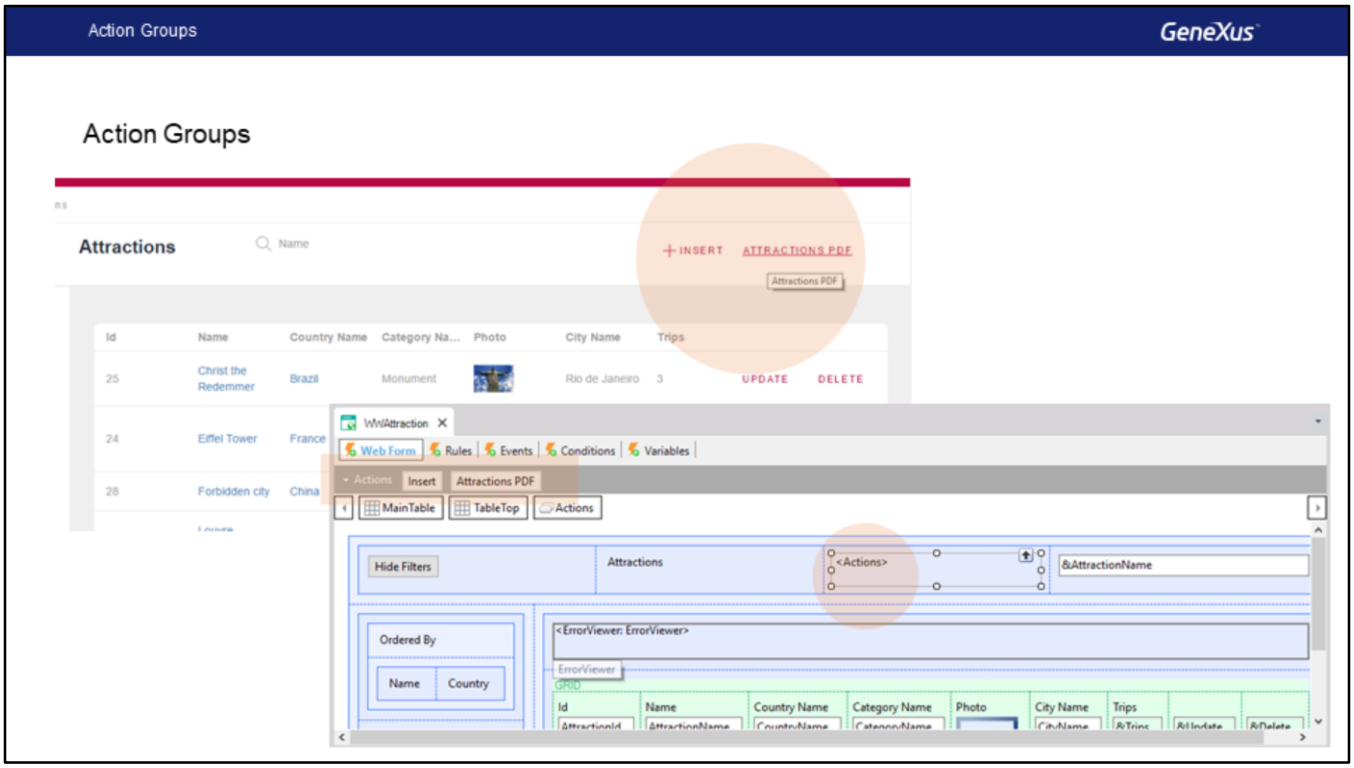
- An Action Group can contain, among other controls, another Action Group control.



If now we select the <Toolbar> action group control in the form, we can see that the group of actions called Toolbar is edited above, in the Action Bar. Also, we see that it has a sub-action group, that is, an action group within another action group. This action group contains the buttons to browse the various records of the transaction and a Select button that, as we know, will call the category selection list, automatically created by GeneXus.



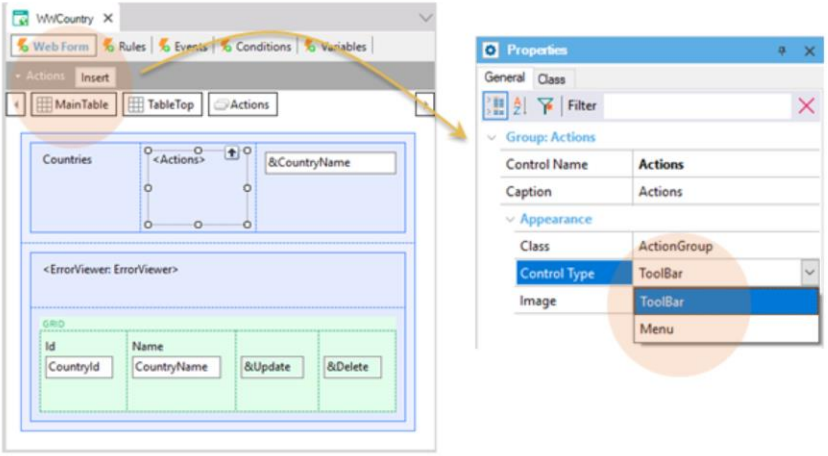
Also, in the Work With web panel implemented by GeneXus, when applying the pattern an action group will be displayed. At runtime, it corresponds to the Insert button.



In the Work With pattern instance applied to the Attraction transaction we had added an action to print the attractions in PDF format. When doing so, we had seen how GeneXus incorporated that action to the corresponding group in the web panel object generated.

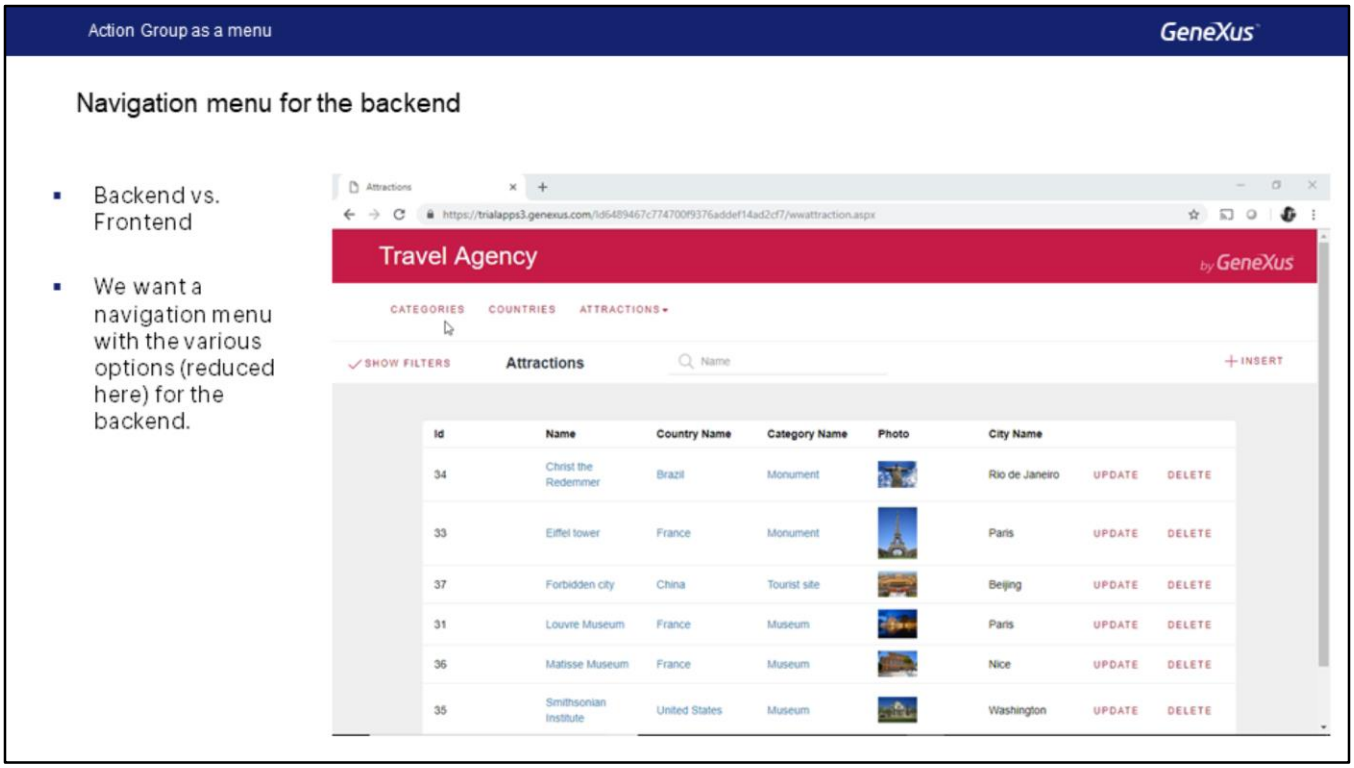
Action Groups

- We can create our own action groups in web panels and transactions, and add them anywhere to the form.
- Two types of action groups:
  - ToolBar
  - Menu



The examples of action groups we've seen before are of ToolBar type. We can see that by selecting the Action Bar and clicking on its properties.

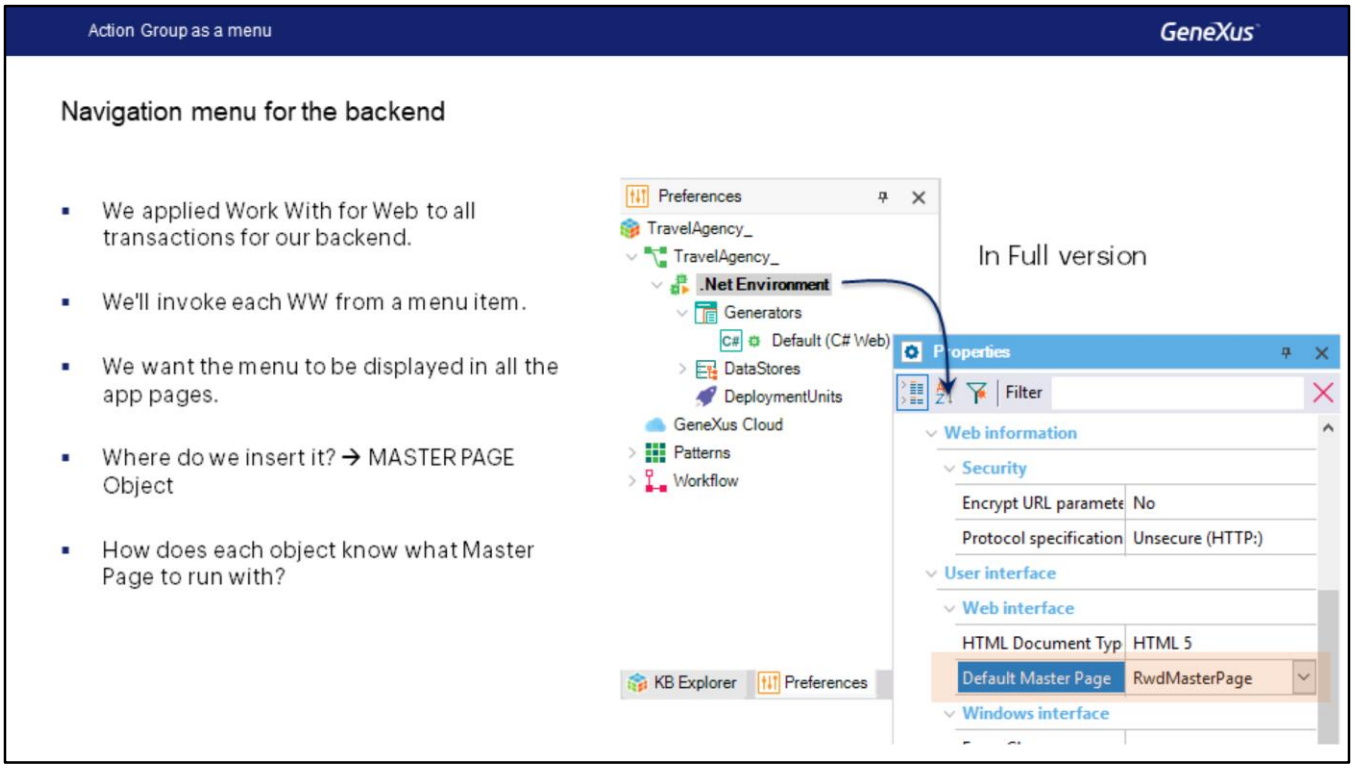
Next, we'll implement an action menu.



[DEMO: https://youtu.be/yiM-ss0_NSY]

We will want to add a menu that is displayed horizontally when the screen size allows it, and like the typical (dropdown) hamburger menu when it doesn't (for example, when the web app runs on phones). In the first demo we see what we want to achieve. It's a simplified version, where we haven't added all the items needed by our app (for example, we haven't added the panels we created in previous classes).

Apps generally have what is known as a **frontend**, that is, the application that clients will use, the app itself, and a **backend**, which is also a web app, but which will be used by those in charge of maintaining the application data; that is, it is an internal application of the company. So far, we have been working in an isolated manner. We haven't paid much attention to those matters. Next, we'll start to integrate them a bit. Most of what we've done will be used in the backend. For example, the Work With elements will be used by the Travel Agency staff to load data, not by the customers who want to look for tours, countries to visit, etc. Even the web panels and reports we've implemented look more appropriate for the backend. There is no functional difference between the backend and the frontend, except for security and permissions. We decide what is implemented in the backend and what is available in the frontend.



So, we will add a menu that will be displayed in all pages of the backend. We have been using the Developer Menu, but obviously it can't be the application homepage. We'll start by applying the Work With pattern to the transactions to which it hadn't been applied.

The menu, as we said, will be a special type of action group. In which web object do we insert it so that it appears as a menu in all the pages we visit inside the backend?

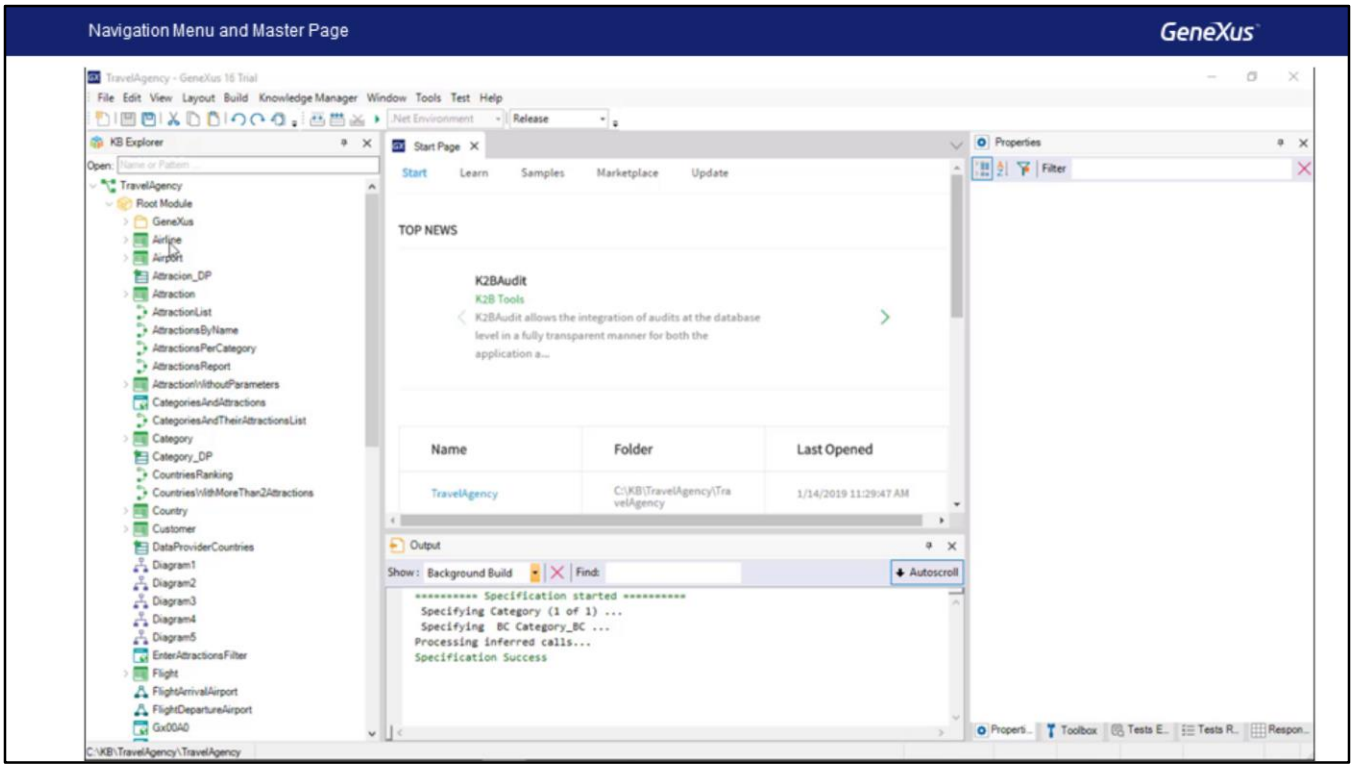
We had already talked about master pages. We had said that all KBs are created with these pages, which are the framework within which all running pages are loaded.

By default, each Environment (later on we'll talk about environments, but for now let's consider it as the instance of the app we're developing) has a Master Page associated with all its web objects (including those we develop from scratch).

In the following demo we'll see, for example, that the Attraction transaction is of Web Page type and that its Master Page is the one called RwdMasterPage. Other Master Pages are predefined in the KB, for other purposes we will not see here.
If we look at the WWAttraction object created by the pattern, we also see it has that Master Page defined as its master page. We'll see the same in the properties of a web panel created by us. Where is this Master Page located? Within the GeneXus folder, Web subfolder.

If we were using the full version of GeneXus instead of the Trial version, we could access the Preferences tab, and replace this default Master Page in all our web pages. Anyway, it is always

possible, for a certain object, to replace the Master Page with any other defined in the KB. In the Trial version we can't change the default Master Page. It will always be the one displayed on top.



[DEMO: https://youtu.be/eCuD6QQsnyU]

In the demo we can see how all pages, those of work with elements, as well as those of transactions and even web panels we've created, are running within this framework where the entire bar at the top remains unchanged. We will want the menu to be in that area.

We open the RwdMasterPage master page and change, in the first place, the app name to Travel Agency. We could add an image there, as an app icon, and everything we want.

We can also see that there is a control of Component type, which has already been mentioned in this course. This component is created by invoking another web panel object in charge of displaying the objects browsed. At runtime, we saw this as Recents.

Then, the most important one is the control <ContentPlaceHolder>. This control can only be placed within another **web panel of Master Page type**. Every page invoked will be loaded there. The rest is the framework that this page will have.

We delete the <component> because we don't want the route of the most visited links, and in its place we will place our menu.
To do so, we open the Toolbox and drag the New Action Group control to its place. A window is opened to enter a name for the action group we're creating. We call it NavMenu. It could be a dynamic menu, but we won't see it here.

Where do we create the menu items? At the top, in the Action Bar shown in gray above the web form. We right-click there and insert whatever we want; for example, a Text Block. We change its properties by changing the control's name, because we will have to name it later in the events. Also, we change the Caption because we want to show it at runtime.
We do the same for each option we want to include.

Next, we open the events and comment or delete within the Refresh event the loading of the component we had deleted. Otherwise, an error will occur when trying to save, because this component no longer exists in the form.

Then, in the Start event we create the links we want for each one of the menu options. This is done through the Link property of the TextBlock control, and then invoking the desired object with the Link method. We will not explain the differences between invoking with and without Link here.

By pressing F5 and opening any web object, the menu is displayed. Here, only the attractions, categories and countries are included for simplicity reasons. But this isn't a navigation menu, it is a ToolBar. It is not displayed as we wanted. The hamburger menu is not shown when the screen size is small. The reason is that we haven't changed its Control Type to Menu.

After doing this and pressing F5, we see it as we wanted. Actually, if we press F12 in Chrome, we can vary the screen to see how it would look not only in desktop computers, but also in mobile devices. It's a good way to test its responsiveness. In an iPhone, for example, we see that the hamburger menu is being displayed.

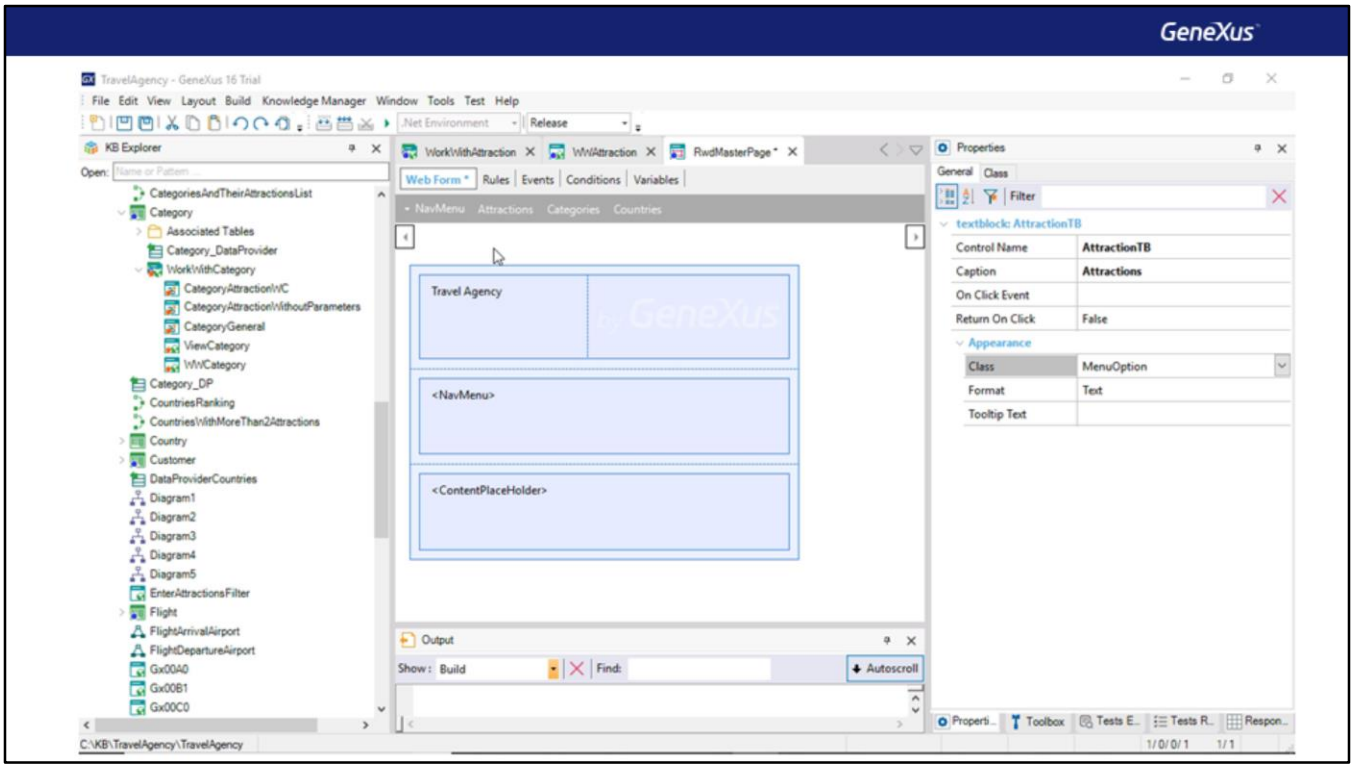
We've seen the following elements:

- **Master Page** property of each web object.
- Master Page Object
  - ContentPlaceholder Control in its web form
  - How to insert an action group
  - How to make it of menu type.

Menu with sub action group

- Action groups can be nested within other action groups
- To change the controls' design or behavior, we associate them with classes, which are defined within a Theme object of the KB (only an introduction is provided here; we'll talk about them later).

Watch the demo in the next page.



[DEMO: https://youtu.be/zPFotMPpw6Q]

To add an action group to another, just right-click / Insert Group. We'll create a submenu to show tourist attractions: the Work With and a couple of reports we had implemented in our KB. They could also be web panels.

There we drag the text block we had in the parent, Attractions, and add two new text blocks, to invoke each report. We need to add a suitable Caption for the Submenu, because it will be shown before displaying it. We enter Attractions as Caption.

At runtime, we see that when the menu is horizontally displayed –not as a hamburger– the submenu is overlapping with the content below. That is to say, its background color is transparent. If we wanted it to be white so this wouldn't happen, where would we change it?

If in GeneXus we edit the properties of the action group that implements the menu (we would see the same by editing those of the submenu), below the Appearance group we see a class property, by default the one called ActionGroup. What is a class?
We've already said when we talked more about web panels that the relationship between a form control and its class is similar to the relationship between the data type of an attribute or variable control and its domain. That is to say, the class centralizes several properties, to be able to change them from a single location, and automatically apply them to all the controls that have it as a class. Note that GeneXus offers a shortcut to edit the properties of the control class we're editing. It's the Class tab, next to General. There we change the Background color to White.

When we do it, an object called Carmine is automatically opened. It's an object of Theme type, and it's the predefined theme of the KB. When we changed the previous value to White, we didn't do it only for the action group control we were editing, but for any other action group which has the one called ActionGroup as a class. The theme object stores the properties of all the possible classes in the KB.
Note that the ActionGroupItem class allows setting the design and behavior of the menu items. The red color we saw in menu options at runtime comes from the Forecolor property of this class.

GeneXus™

The power of doing.

Videos	training.genexus.com
Documentation	wiki.genexus.com
Certifications	training.genexus.com/certifications