

# Design of an Angular application

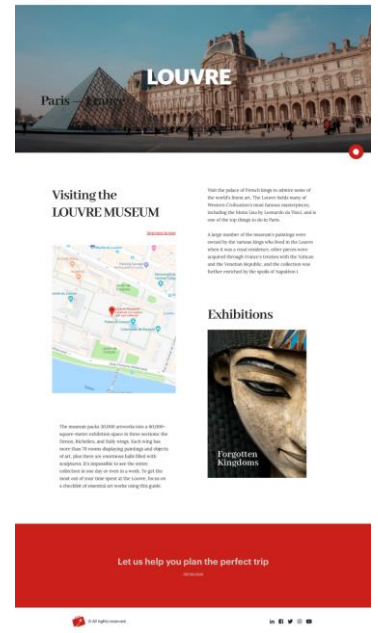
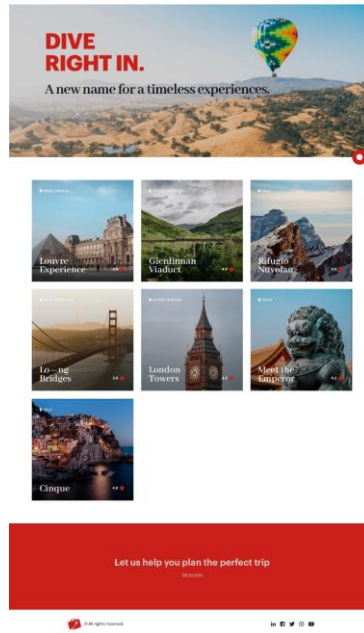
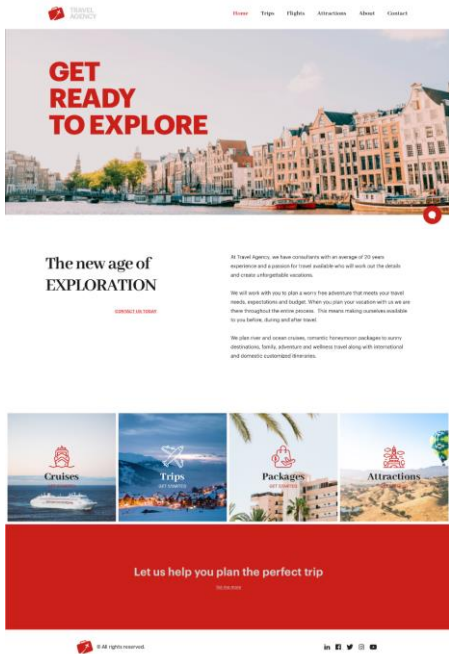
Importing a design from Sketch

*GeneXus*<sup>™</sup>

In a previous video you saw how you to determine your own design definitions for your app using the Design System Object.

This video will show you how to import –into your GeneXus app– the design of full screens, specifying colors, fonts, etc., as defined by digital app design professionals in Sketch (their usual working tool),

## Design from the designer



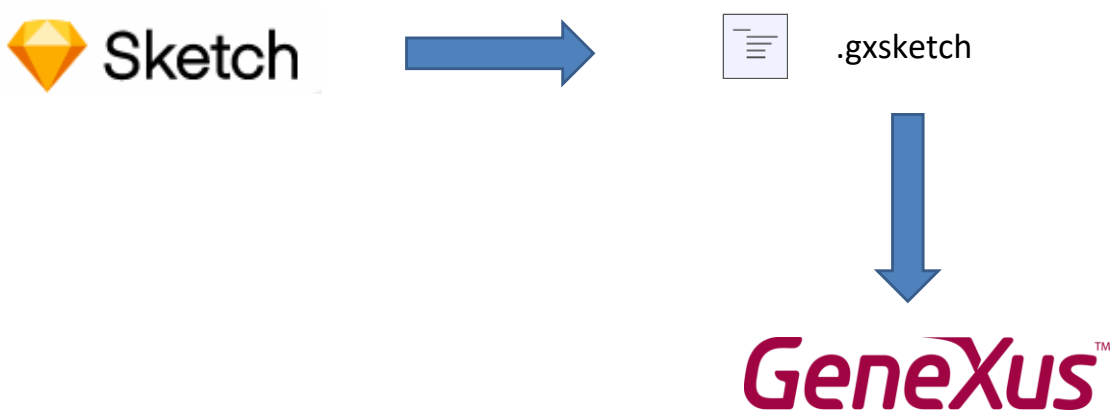
You have agreed with the designer that you will need an initial screen as the baseline for the remaining screens in the app, another screen to view the list of tourist attractions available, and a third screen with details of a tourist attraction.

These screens are defined by the designer. To the left, on top of the main screen, you will see the app's logo and the menu with actions. Further below there is a background image with a highlighted title on it, and below and to the right is a button to start a chatbot. Then there is a section with data on the travel agency and underneath four buttons to access the data on the entities that your app will be handling. Below the buttons there is a footer with a red frame that includes a message and the links to social networks.

In the screen in the middle, the logo is on top, together with the menu and a context image that includes a title and a chatbot button, followed by a grid with the tourist attractions available, and the footer. You will see that many of these elements are also included in the third screen that shows the detail of one attraction (the Louvre in this case).

It is obvious that the designer has done a good job with the uniform appearance of all the screens that show a clear and clean way of presenting the information.

## Importing from Sketch

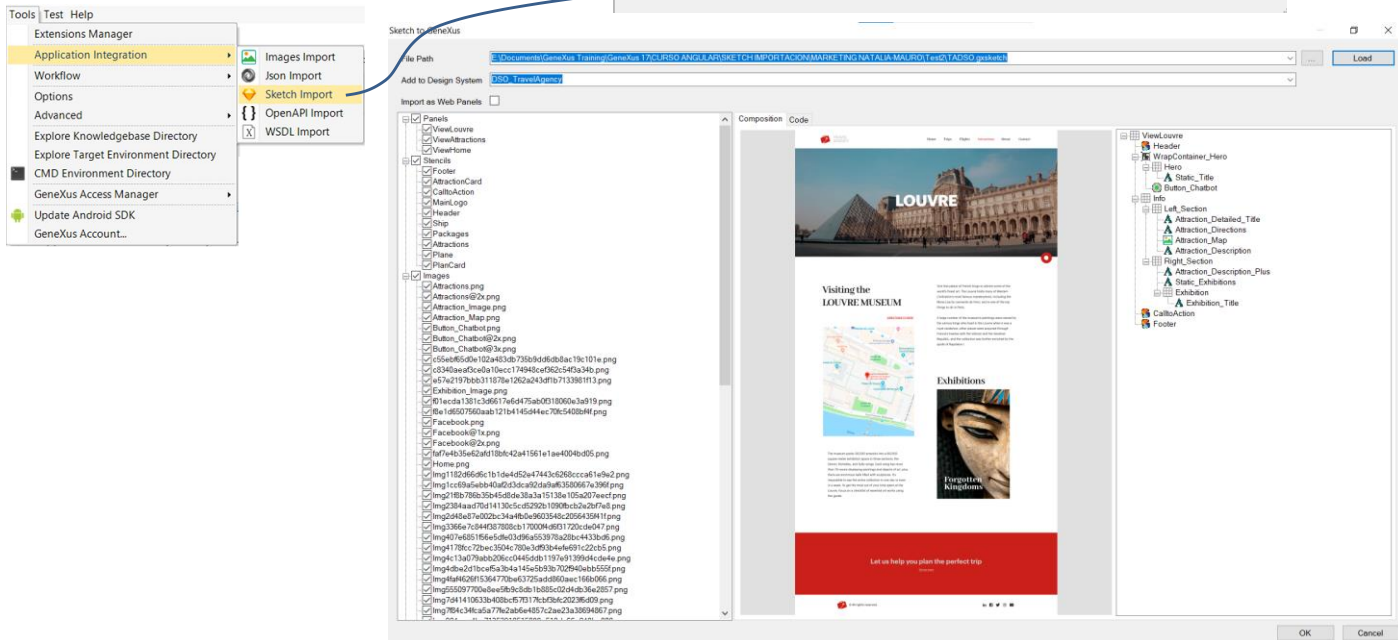


This design was done in the regular tool, that is, Sketch. Then a GeneXus plugin installed in Sketch is used to create a file with extension gxsketch, which is the one you receive.

GeneXus allows you to import this file created in the designer's tool, directly into your KB, with all design elements automatically created in a Design System Object. Even the corresponding GeneXus objects will be created with all the controls required for the app to function with the screens shown.

---

## Importing the .gxs sketch file



To import the .sketch file sent by the designer go to Tools/Application Integration and select Sketch import.

Add the name of the Design System Object where you want to save all the design definitions.

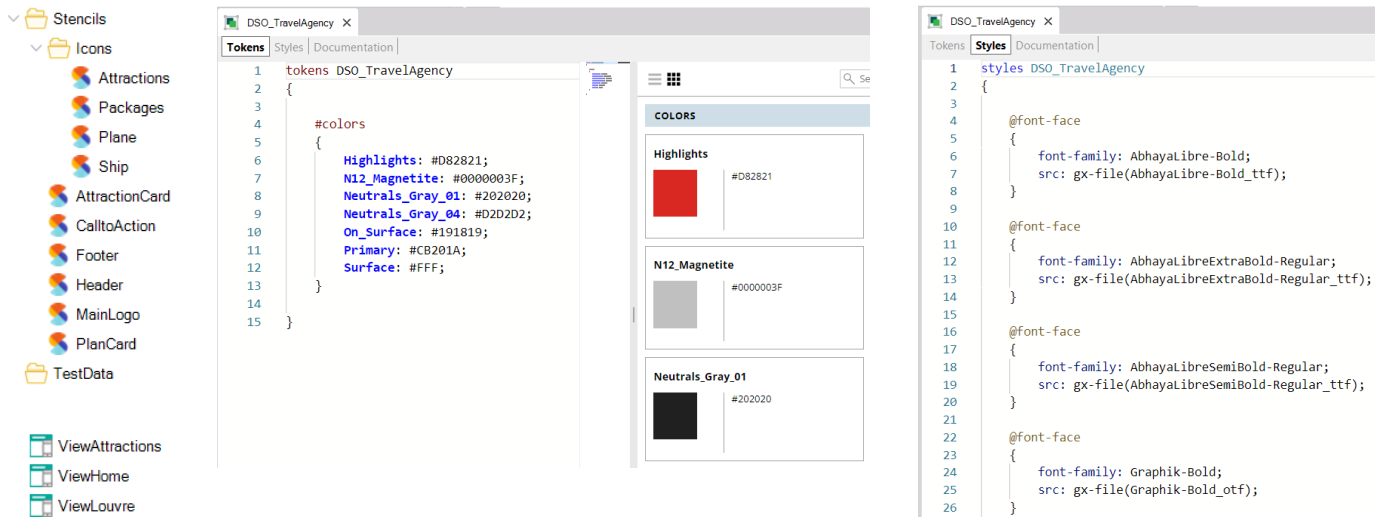
The check box Import as Web Panels should be unchecked because you need to create panels and not webpanels.

If you click on any of the panels that the import will create you will have its preview. where you can confirm the match with the design validated. To the right, you are informed about the controls that it will contain.

Note that also the images will be imported to execute the panel with fixed data allowing you to execute and verify that it is the functionality desired. Then you must substitute these fixed data with the data stored in your app's database.

You agree, so you press OK.

## Objects created in the import



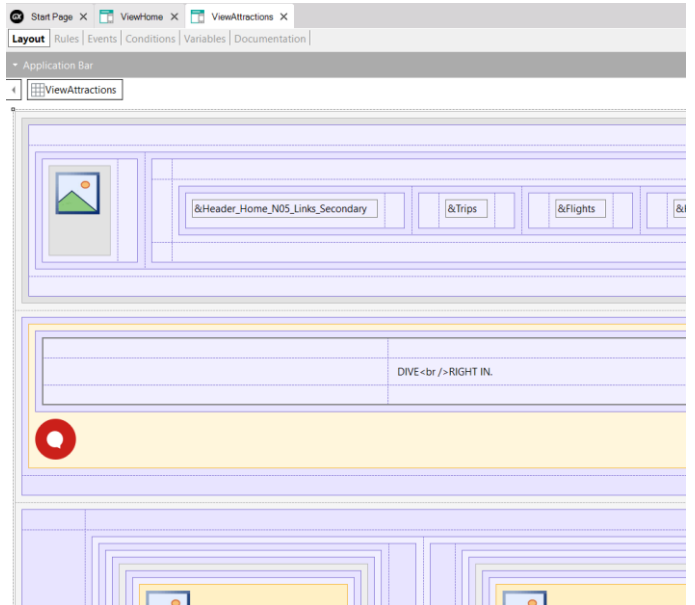
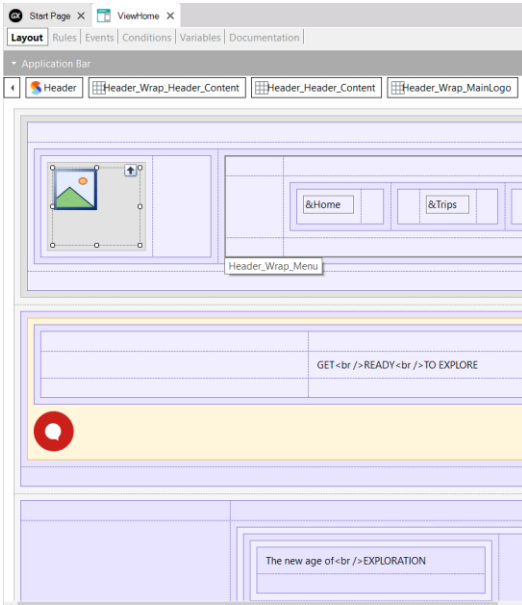
The Output window shows the import's progress and the error-free results.

In KB Explorer, you will see that two folders were created: `Stencils`, containing several stencils used to encapsulate design in the app, and `Test Data`, containing data providers and sdt to load fixed data, which will enable you to test the app and verify if the design is the one expected.

Further down, you will see that the `DSO_TravelAgency` Design System Object has been created. If you open it, you will see the `Tokens` and `Styles` that defined the appearance of controls on screen.

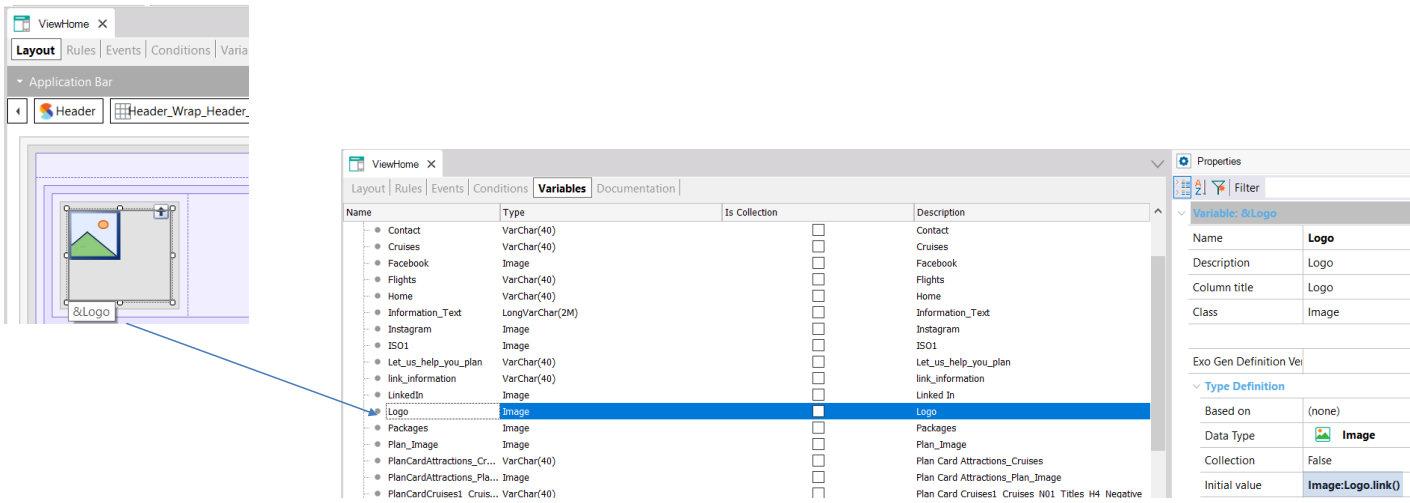
On the bottom, you will find the panel objects you saw on the import wizard screen, which were created automatically, like the one of the home page (`ViewHome`), the one that shows the attractions (`ViewAttractions`), and the detail of an attraction (`ViewLouvre`).

## Panel objects created automatically in the import



In the ViewHome form you will see that all components for the visual content -like tables, variables, and images, among others- are already included. All this was automatically created in the importing process. And the same goes for the ViewAttractions panel.


## Objects created in the import



The screenshot displays the GeneXus development environment. On the left, a small preview window shows a UI element labeled '&Logo'. The main window is titled 'ViewHome X' and has tabs for 'Layout', 'Rules', 'Events', 'Conditions', 'Variables', and 'Documentation'. The 'Variables' tab is active, showing a list of variables with columns for Name, Type, Is Collection, and Description. The 'Logo' variable is selected and highlighted in blue.

Name	Type	Is Collection	Description
Contact	VarChar(40)	<input type="checkbox"/>	Contact
Cruises	VarChar(40)	<input type="checkbox"/>	Cruises
Facebook	Image	<input type="checkbox"/>	Facebook
Flights	VarChar(40)	<input type="checkbox"/>	Flights
Home	VarChar(40)	<input type="checkbox"/>	Home
Information_Text	LongVarChar(2M)	<input type="checkbox"/>	Information_Text
Instagram	Image	<input type="checkbox"/>	Instagram
ISO1	Image	<input type="checkbox"/>	ISO1
Let_us_help_you_plan	VarChar(40)	<input type="checkbox"/>	Let_us_help_you_plan
link_information	VarChar(40)	<input type="checkbox"/>	link_information
LinkedIn	Image	<input type="checkbox"/>	LinkedIn
<b>Logo</b>	<b>Image</b>	<input checked="" type="checkbox"/>	<b>Logo</b>
Packages	Image	<input type="checkbox"/>	Packages
Plan_Image	Image	<input type="checkbox"/>	Plan_Image
PlanCardAttractions_Cr...	VarChar(40)	<input type="checkbox"/>	Plan Card Attractions_Cruises
PlanCardAttractions_Fla...	Image	<input type="checkbox"/>	Plan Card Attractions_Plan_Image
PlanCardCruises1 Cruis...	VarChar(40)	<input type="checkbox"/>	Plan Card Cruises1 Cruises N01 Titles H4 Negative

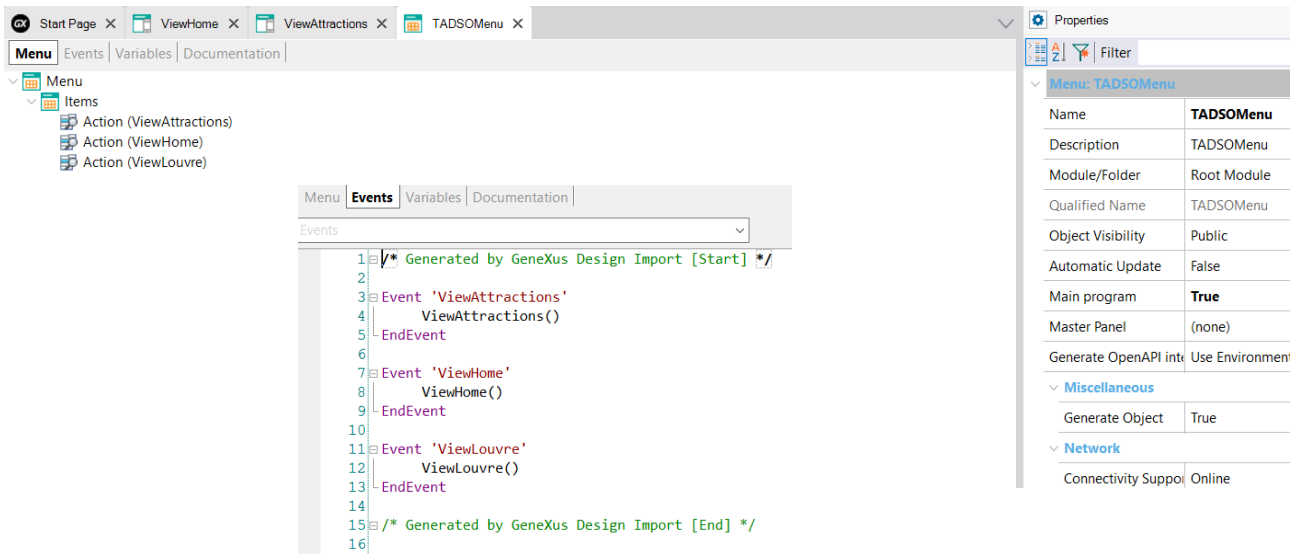
The Properties panel on the right shows details for the selected 'Logo' variable:

Variable: &Logo	
Name	Logo
Description	Logo
Column title	Logo
Class	Image
Exo Gen Definition Ve	
Type Definition	
Based on	(none)
Data Type	 Image
Collection	False
Initial value	Image:Logo.link()

If you open the ViewHome panel to see the variables, you will find that the images were loaded through the Initial Value property by making reference to the images loaded in the import process into the KB.

And if you go to events, you will find that the interaction between panels is also programmed, since pressing the attractions button will invoke the ViewAttractions panel.

## Startup object in the app



The screenshot shows the GeneXus IDE interface. The main window displays the menu configuration for the application. The menu items are listed as follows:

- Menu
- Items
  - Action (ViewAttractions)
  - Action (ViewHome)
  - Action (ViewLouvre)

The Events tab is active, showing the following code:

```

1 /* Generated by GeneXus Design Import [Start] */
2
3 Event 'ViewAttractions'
4   ViewAttractions()
5 EndEvent
6
7 Event 'ViewHome'
8   ViewHome()
9 EndEvent
10
11 Event 'ViewLouvre'
12   ViewLouvre()
13 EndEvent
14
15 /* Generated by GeneXus Design Import [End] */
16

```

The Properties panel on the right shows the details for the Menu: TADSOMenu:

Name	TADSOMenu
Description	TADSOMenu
Module/Folder	Root Module
Qualified Name	TADSOMenu
Object Visibility	Public
Automatic Update	False
Main program	True
Master Panel	(none)
Generate OpenAPI into Use Environment	
<b>Miscellaneous</b>	
Generate Object	True
<b>Network</b>	
Connectivity Support	Online

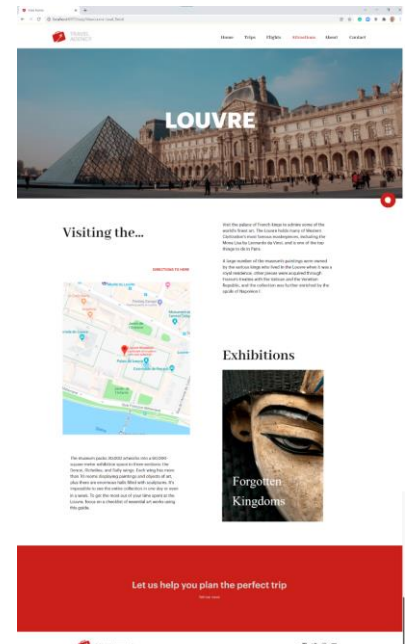
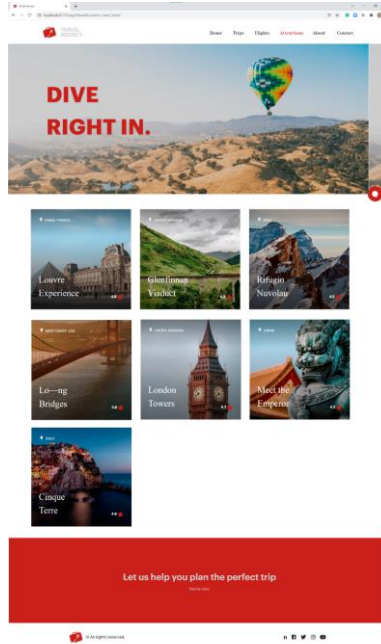
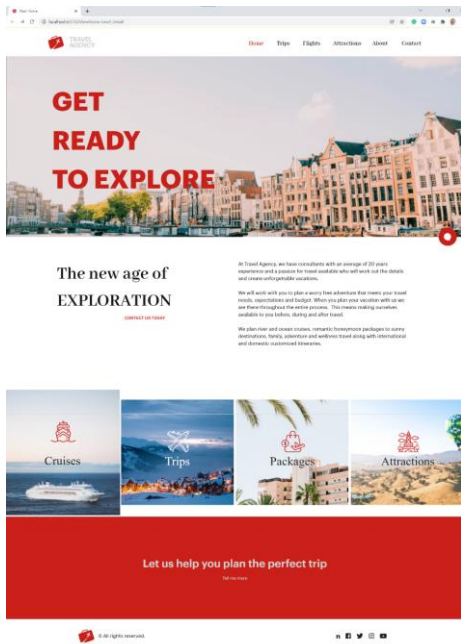
Here you can see that a menu object was also created. By default, it would be the main object to be executed. Note that the app's other panels are invoked in the events.

This object will only be seen in the generation for native platform, and in the case of Angular, the object seen first will be the ViewHome, which is the second option in the menu.

To execute, set the ViewHome object as Startup Object and press F5.



## Running the app with fixed data



Note that the initial screen is executed and it is just as the designer anticipated!

When you click on Attractions you will see the list of tourist attractions of the travel agency.

Remember that these you see are the fixed data loaded during the import.

If you click on Louvre, a page with detail information will appear, with a neat design that is aesthetically attractive.

This example showed the development process of an application and the significance of teamwork with individuals of varied profiles, where each contributes to a better outcome in the solution.

## Modifying the new objects to access data in the database

```

Layout | Rules | Events | Conditions | Variables | Documentation
Events
1  /* Generated by GeneXus Design Import [Start] */
2
3  Event Grid_Attractions.Load
4      Composite
5          For &ViewAttractions_Grid_Attractions_SDT in ViewAttractions_Grid_Attractions_DP()
6              Grid_Attractions.ItemLayout = &ViewAttractions_Grid_Attractions_SDT.Layout
7              load
8          EndFor
9      EndComposite
10 EndEvent
  
```

Structure		Documentation
Name	Type	
ViewAttractions_Grid_Attractions_SDT		
Layout	VarChar(40)	
Attraction_Image	Image	
Opacity_Adjustment_2	Image	
Attraction_Location	VarChar(40)	
Location_Icon	Image	
Opacity_Adjustment_1	Image	
Attraction_Name	VarChar(40)	
Star_Icon	Image	
Rating_Value	VarChar(40)	

```

Start Page X | ViewAttractions_Grid_Attractions_DP X
Source | Rules | Variables | Help | Documentation
1  ViewAttractions_Grid_Attractions_DP
2  {
3
4      ViewAttractions_Grid_Attractions_SDT
5      {
6          Layout = "AttractionItem4"
7          Attraction_Image = Image:e04d854f986edeff804aafef50f757e5120c94ce.Link()
8          Opacity_Adjustment_2 = Image:Opacity_Adjustment_2.Link()
9          Attraction_Location = Upper("Italy")
10         Location_Icon = Image:Location_Icon.Link()
11         Opacity_Adjustment_1 = Image:Opacity_Adjustment_1.Link()
12         Attraction_Name = "Rifugio" + NewLine() + "Nuvolau"
13         Star_Icon = Image:Star_Icon.Link()
14         Rating_Value = "4.5"
15     }
16 }
17
18 ViewAttractions_Grid_Attractions_SDT
19 {
20     Layout = "AttractionItem4"
21     Attraction_Image = Image:b8d1a9828c1a76bdbd62713259d82fc5055db4c1.Link()
22     Opacity_Adjustment_2 = Image:Opacity_Adjustment_2.Link()
23     Attraction_Location = Upper("United Kingdom")
24     Location_Icon = Image:Location_Icon.Link()
25     Opacity_Adjustment_1 = Image:Opacity_Adjustment_1.Link()
26     Attraction_Name = "Glenfinnan" + NewLine() + "Viaduct"
27     Star_Icon = Image:Star_Icon.Link()
28     Rating_Value = "4.9"
29 }
30 }
ViewAttractions_Grid_Attractions_SDT
  
```

So far, you have dealt with fixed data for tests that the designers added to show the app's behavior with the new design. Now you will make some changes in order to view the real data in your database.

If you go to the events of the ViewAttractions panel you will see that the grid is loaded using a variable of the SDT type called `&ViewAttractions_Grid_Attractions_SDT` and the `ViewAttractions_Grid_Attractions_DP` data provider.

Now substitute the DP load so that, instead of taking fixed data, it uses the attributes from the Attraction table and the Country table.

## Modifying the new objects to access data in the database

```

ViewAttractions x ViewAttractions_Grid_Attractions_DP x ViewAttractions_Grid_Attractions_DP_BD x
Source Rules Variables Help Documentation
1 ViewAttractions_Grid_Attractions_DP_BD from Attraction
2 {
3   ViewAttractions_Grid_Attractions_SDT
4   {
5     Layout = "AttractionItem"
6     Attraction_Image = AttractionPhoto
7     Opacity_Adjustment_2 = Image:Opacity_Adjustment_2.Link()
8     Attraction_Location = CountryName
9     Opacity_Adjustment_1 = Image:Opacity_Adjustment_1.Link()
10    Attraction_Name = AttractionName
11    Star_Icon = Image:Star_Icon.Link()
12    Rating_Value = str(AttractionRating)
13  }
14 }
15

```

```

ViewHome x ViewAttractions x Travel_Agency_App_MobileMenu x Navigation View x
Layout Rules Events Conditions Variables
Events
1 /* Generated by GeneXus Sketch Import [Start] */
2
3 Event Grid_Attractions_Small.Load
4   For &ViewAttractions_Grid_Attractions_Small_SDT in ViewAttractions_Grid_Attractions_Small_DP_BD()
5     Grid_Attractions_Small.ItemLayout = &ViewAttractions_Grid_Attractions_Small_SDT.Layout
6     load
7   EndFor
8 EndEvent

```

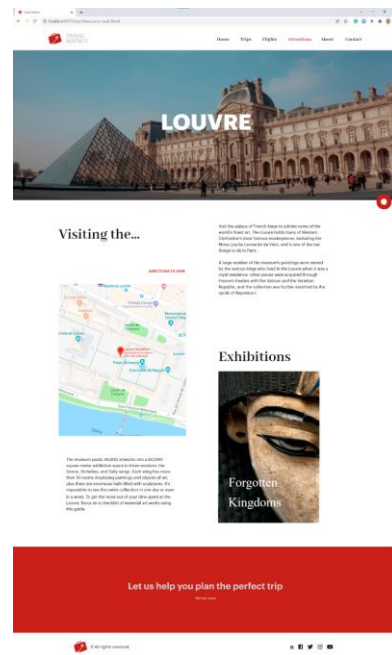
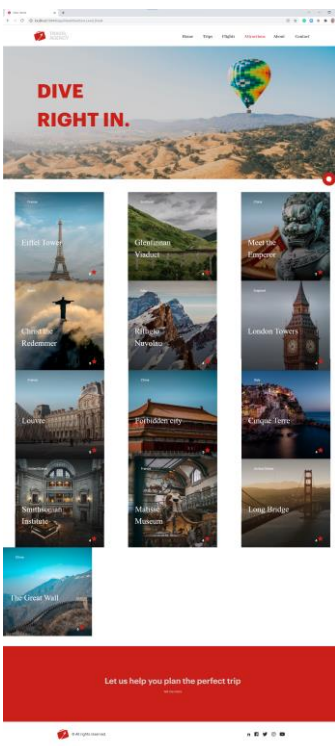
With the Data Provider you do Save As and save it under the name ViewAttractions\_Grid\_Attractions\_DP\_BD.

Then you add the clause from Attractions and load the Attraction\_image field with the value of the AttractionPhoto attribute, and the Attraction\_name field with the AttractionName attribute. And to Attraction\_location assign the CountryName attribute, while the value of the AttractionRating attribute should be assigned to Attraction\_rating . Then proceed to delete everything that is not needed.

Now, go back to the ViewAttractions panel and change the data provider that loads the attractions and replace it with the new one you built.

Then save, right click on the ViewHome panel and select Run.

## Running the app with real data



If you go to attractions, you will see that you are going over the attractions you had loaded in the database, including the Eiffel Tower, and the Glennfinn Viaduct, which were not in the fixed data. And if you page, you will find all the attractions. When you click on Louvre you will see the page that includes the attraction's details.

The Agency's request to show all tourist attractions has been fulfilled, but now the app has a more appropriate design.

There is no doubt that, when there is a designer available in the team, the work of writing the DSO from scratch by hand becomes absolutely unnecessary. Importing these definitions from Sketch solves the design in a more professional way, so you can focus on the functional part of your app.

This working method that combines the design by a professional into your GeneXus project implies several advantages, because it optimizes efforts, allowing each team member to contribute in accordance with that individual's specific profile in relation to what she/he does best.

Upcoming videos will deal with other aspects such as adding security to your app, or how to commission your app on the client's facilities.

# GeneXus™

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

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

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