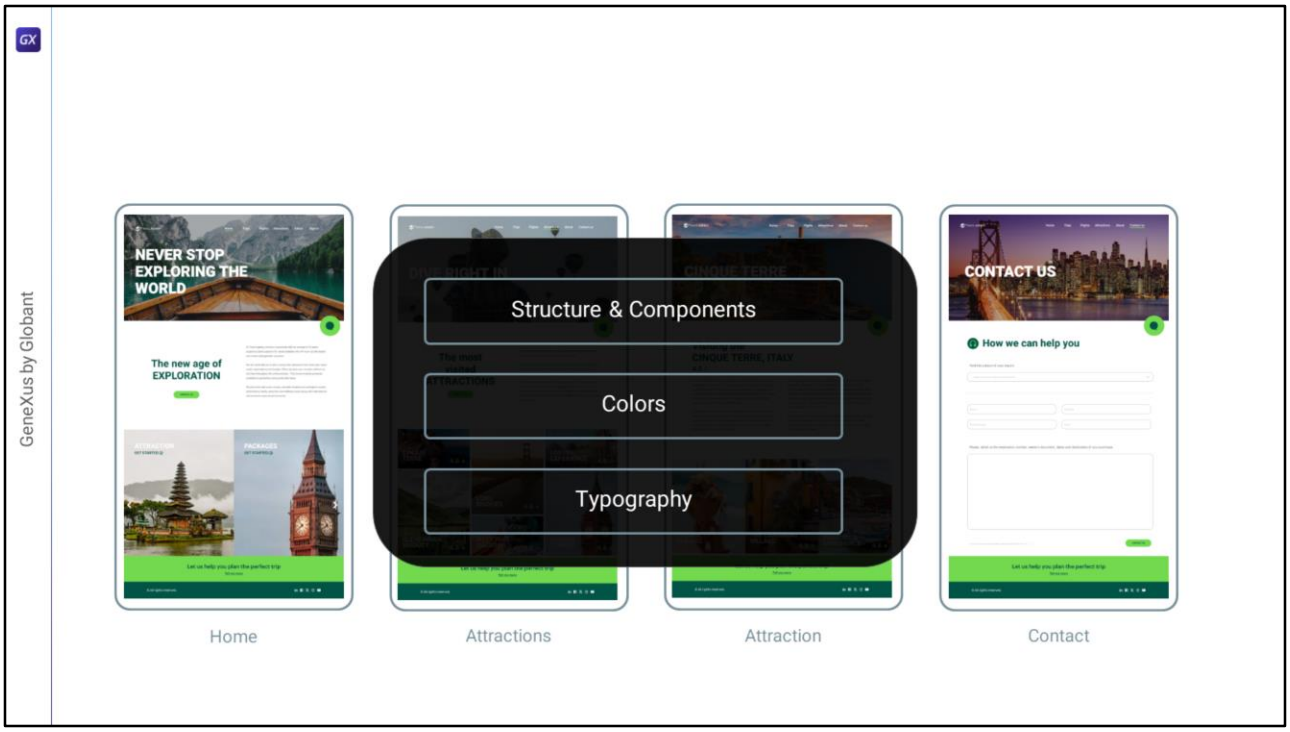
GeneXus by Globant

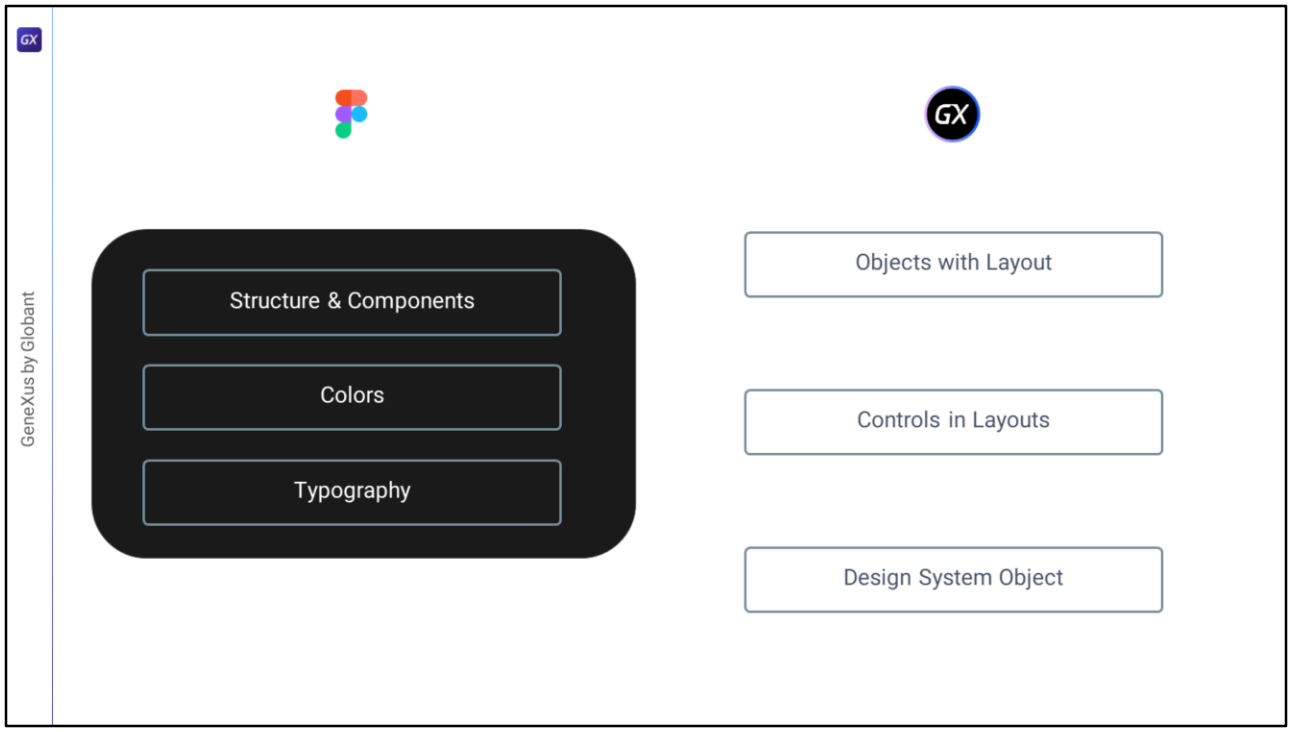
Design in GeneXus. Players

Cecilia Fernández



In the previous class, Chechu, our star designer, had analyzed these dimensions of the design system that she created to implement the design of the Travel Agency application that we wanted to develop.

The question that will drive not only this class, which is a kind of review, but the whole course in general will be how to apply all that she was talking about, which has to do with creating the design system, and how to take it to GeneXus.



So let's start by analyzing these points as a review that will introduce us to the course.

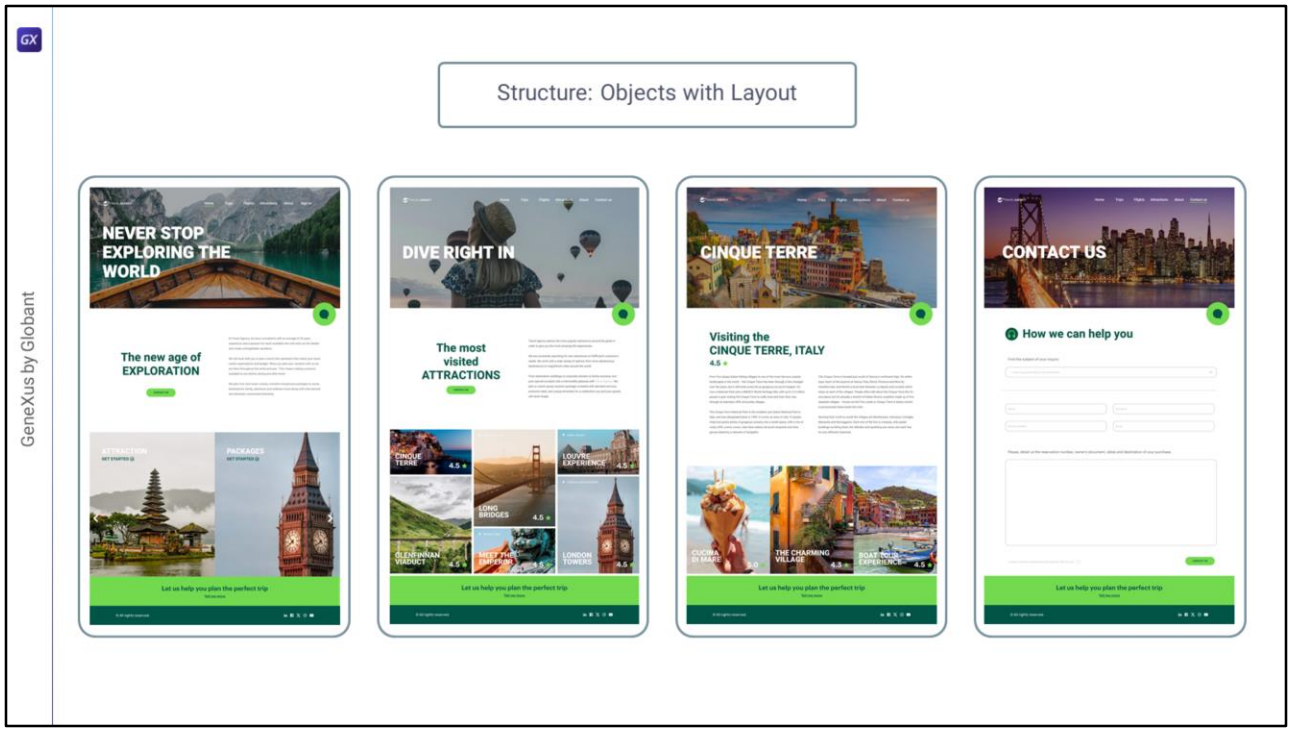
The structure and components of the first item have to do with how the application screens are structured and componentized. In GeneXus, this will be related, in a first dimension, with the identification of objects with a layout, which will represent each part of these screens.

And then in a second instance, which is more micro because it is internal to each object, we will have to see how the controls are organized within each layout, to reflect that structure.

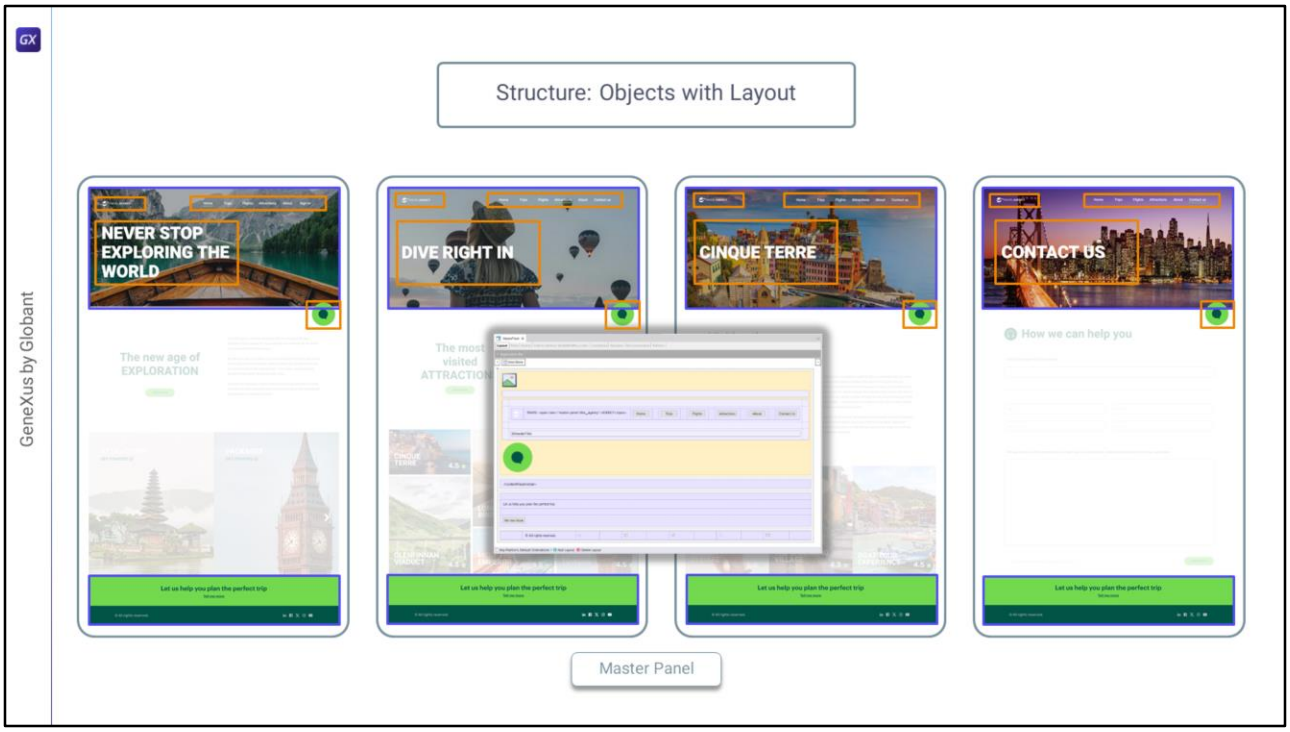
Next, we will have to get into the topic of the style itself, where colors, typography, and other aspects that we will discuss and that in GeneXus will be given by the Design System object.

So, in this class we are going to review all this before going deeper into each topic.

Let's start by viewing what we mean by the division of all screens into objects with a Layout.

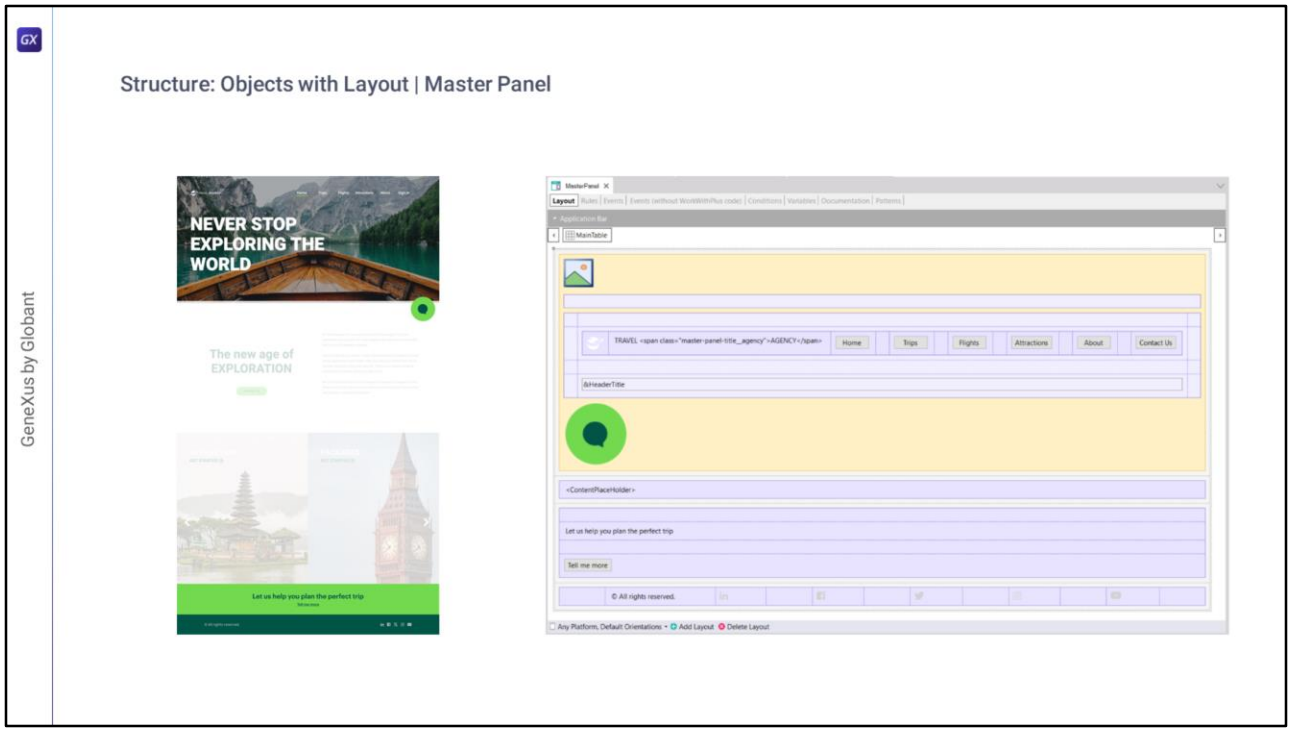


If we look at the 4 screens that Chechu defined...

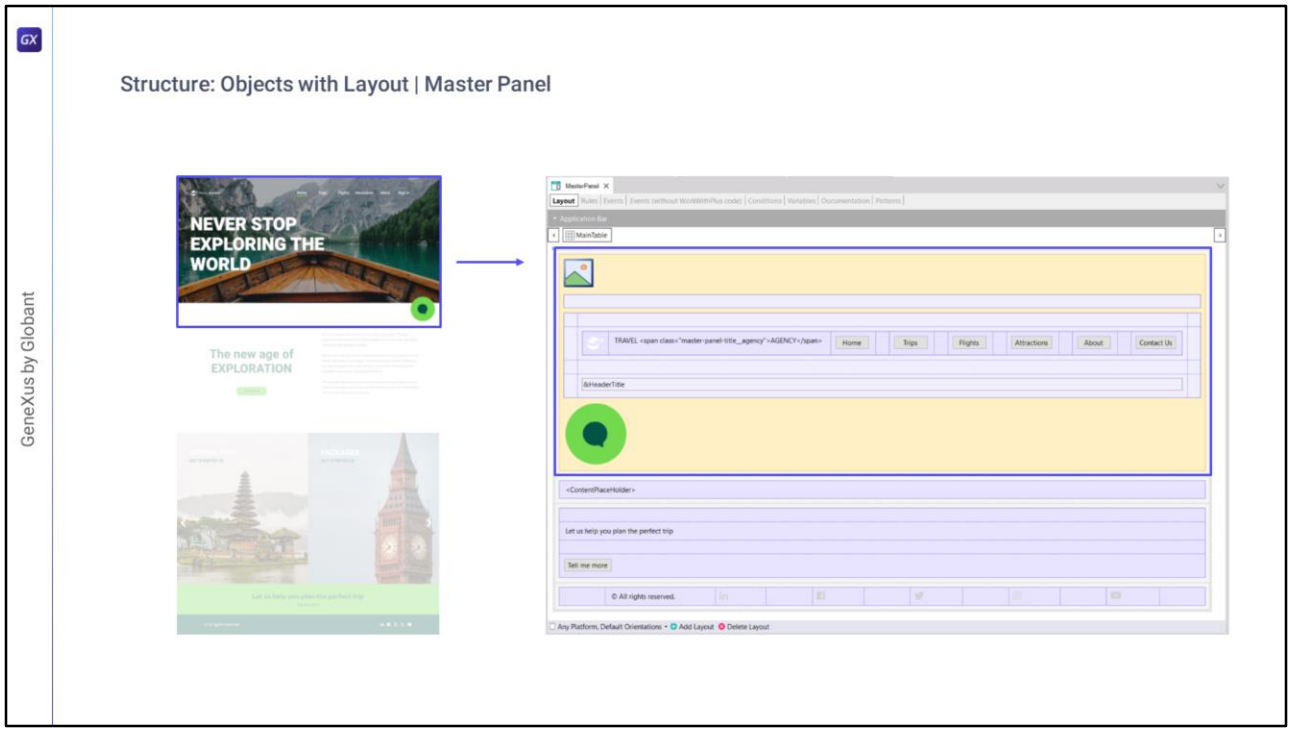


...we can see that all this is repeated in structure and style. Thus, all page headers will have a background image that takes up the same space on each screen (although the image itself varies, the size and type of image is the same); the image will be superimposed with the company logo and the menu, a text (which will also vary from screen to screen but not in style or structure) and an image for chatting. In addition, a footer will be displayed on each screen, and in this case the style and content will also be the same.

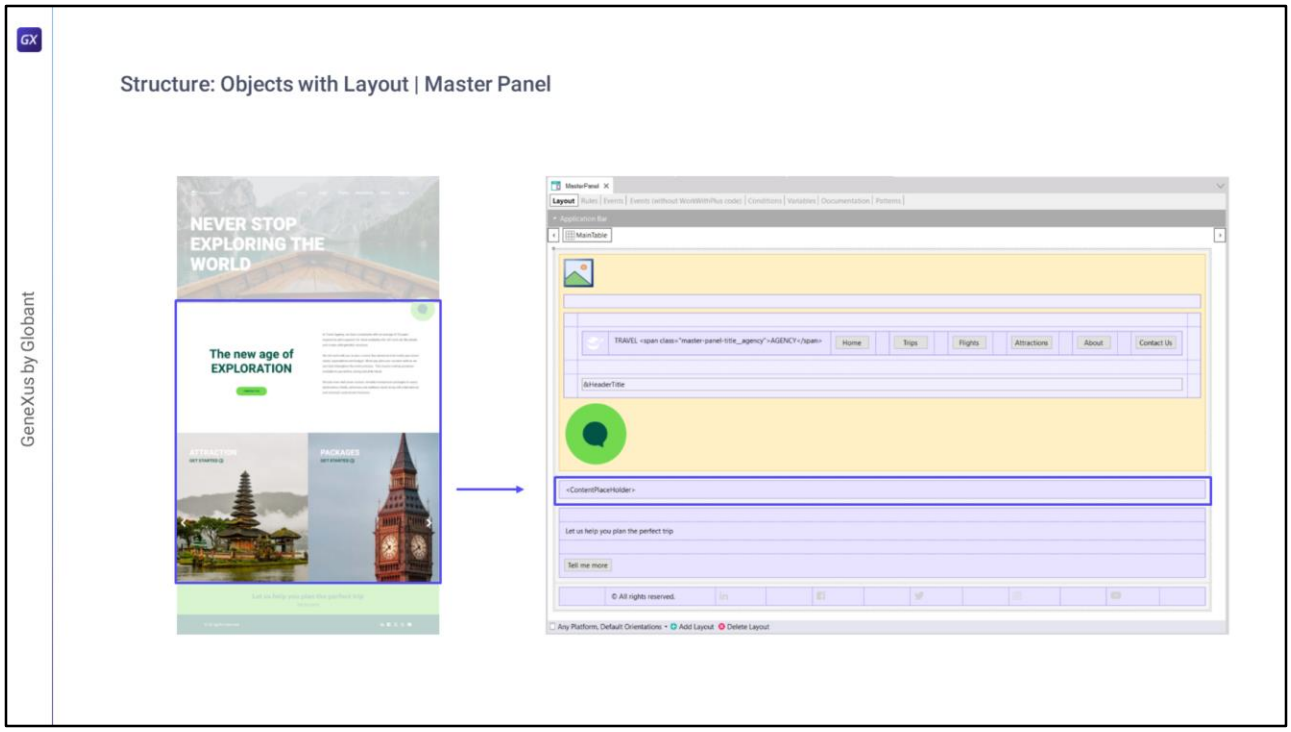
As we know, in GeneXus we have the **Master Panel** object for designing the appearance and behavior of the parts that will be common to all screens (or at least to a set of them: in this case, to these first 4 screens).



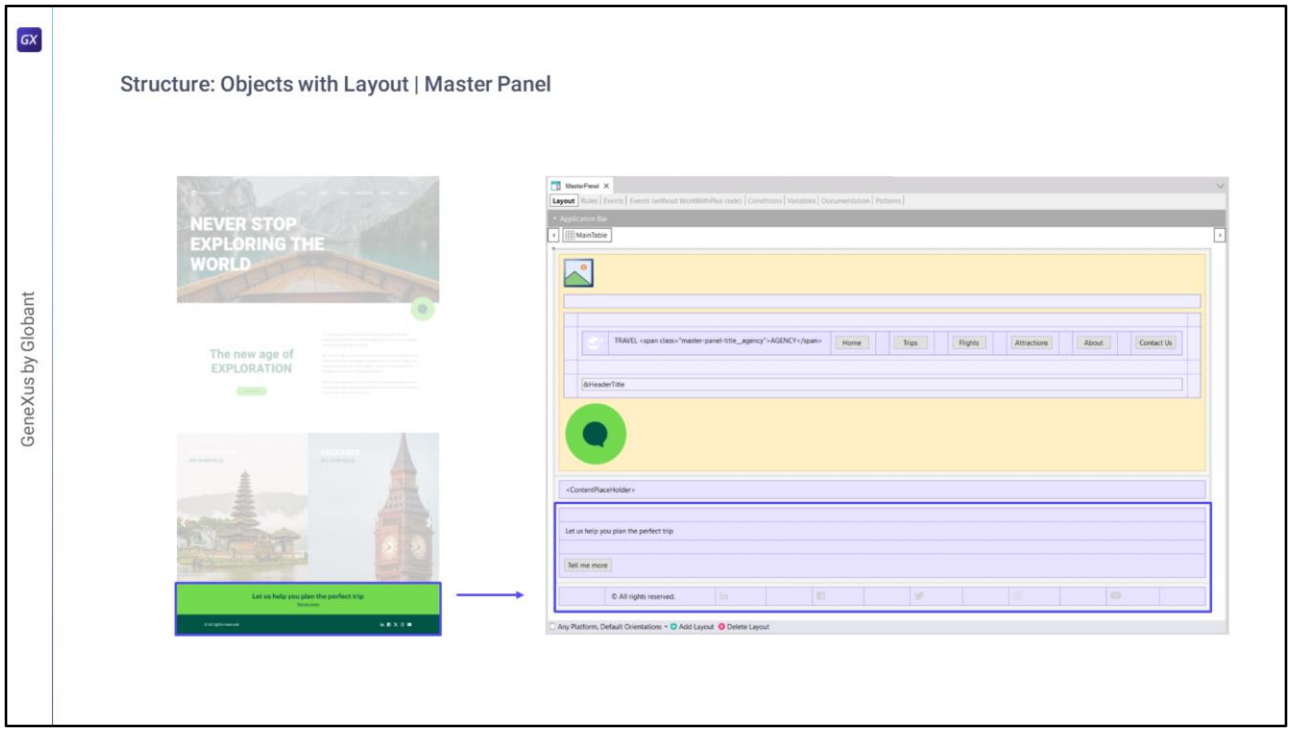
So we are going to implement all this that will be repeated in a Master Panel object.



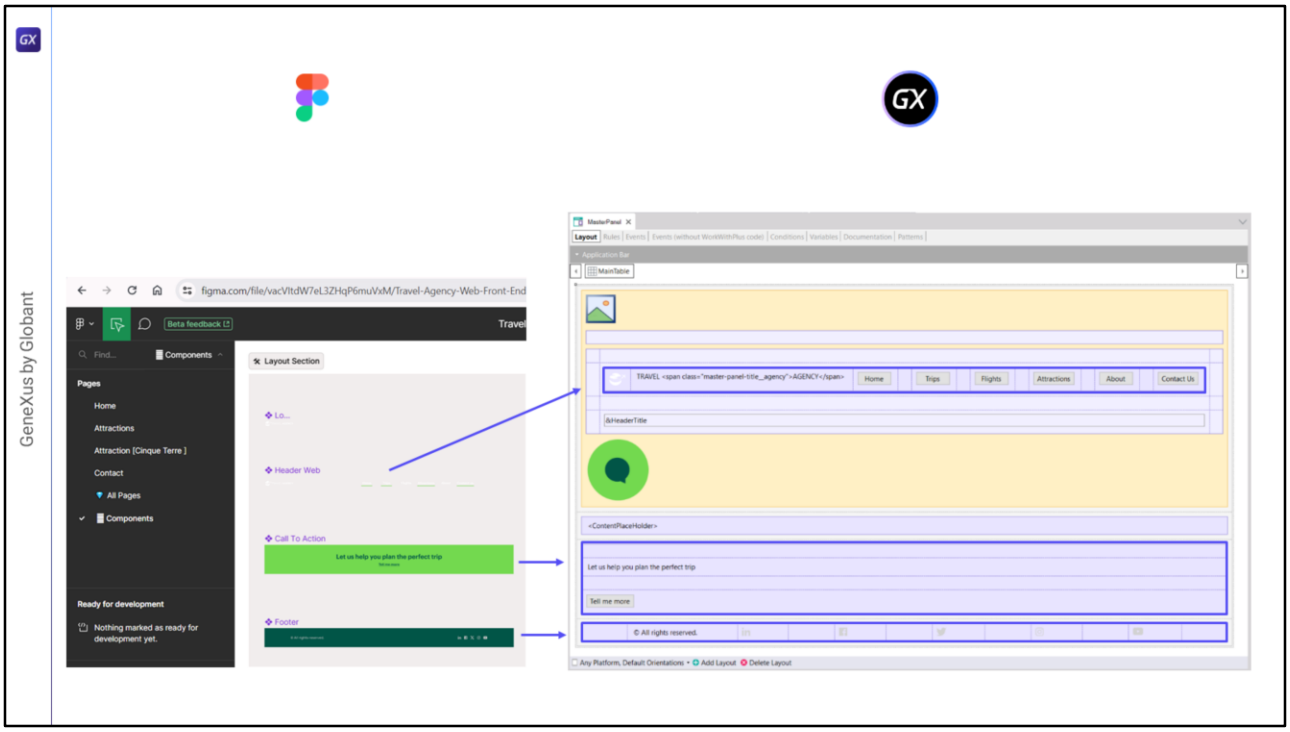
In the first row of the Master Panel table, we are going to model the header...



In the second row we will have this ContentPlaceholder control that is specific to the Master Panel object, which precisely indicates that every page with this Master Panel will have its information loaded there...

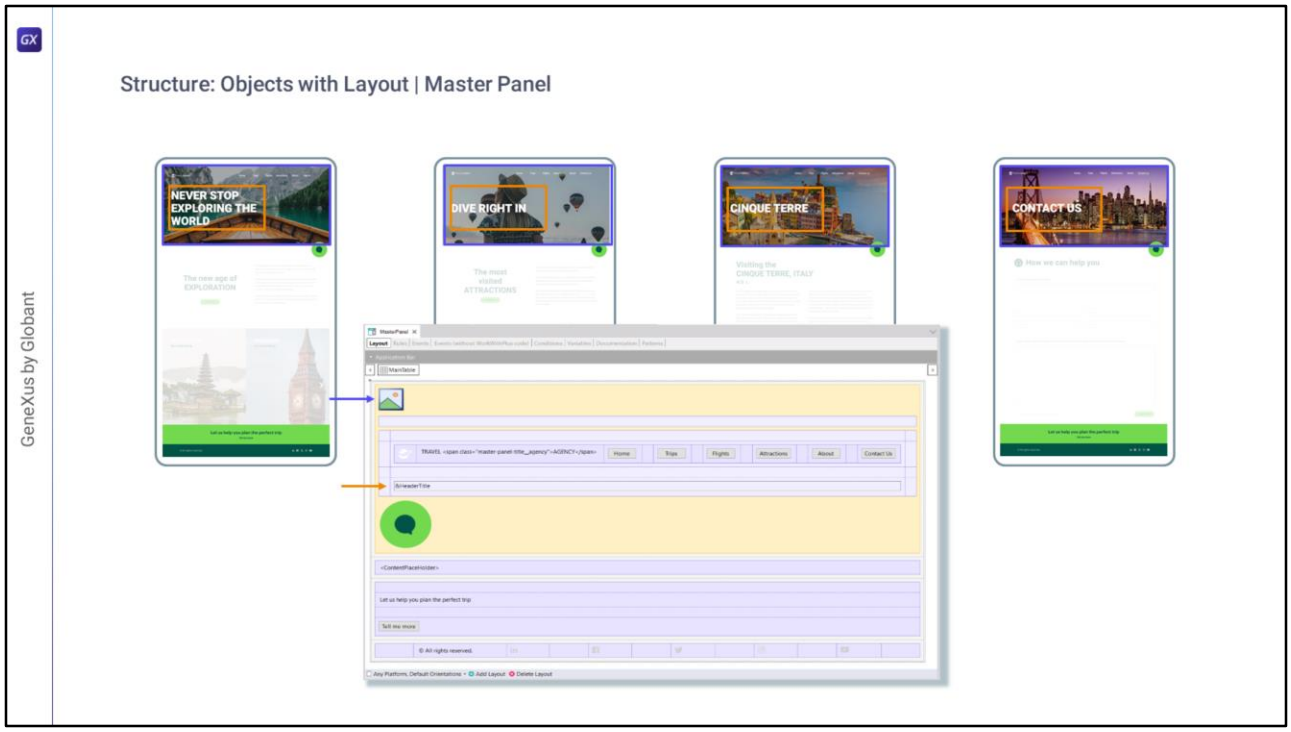


...and lastly –this is also specific to the Master Panel– in rows 3 and 4 we are going to model this footer.



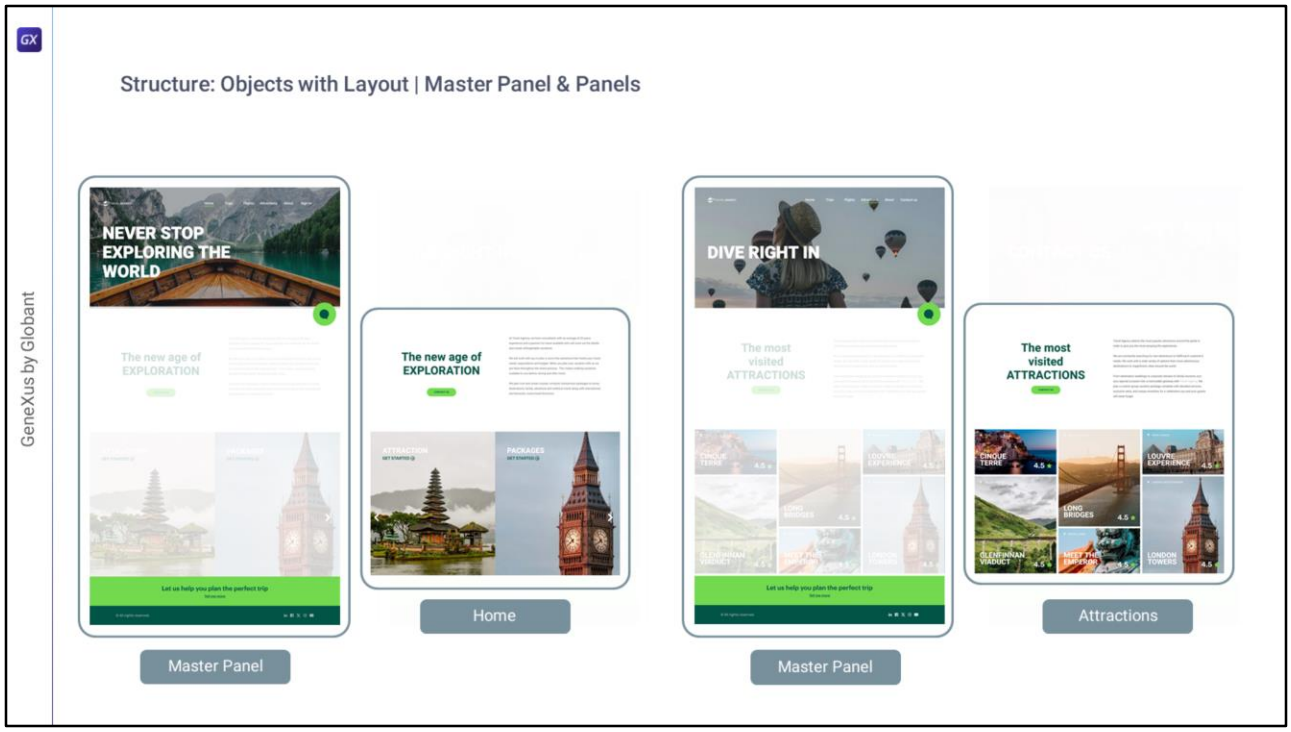
Good. If we look at what Chechu modeled in Figma, she had no way to express this. The most she could do was to model these parts (the menu and these parts of the footer) as components. She can't establish a relationship between them because she doesn't have a tool that allows her to express this same thing that we can express in GeneXus through a Master Panel object.

This is where we start to see that the design tools and GeneXus do not have exactly the same expressive capabilities, so it is necessary to understand a bit of both tools, in particular the design tool being used, to make the best combined use of them.



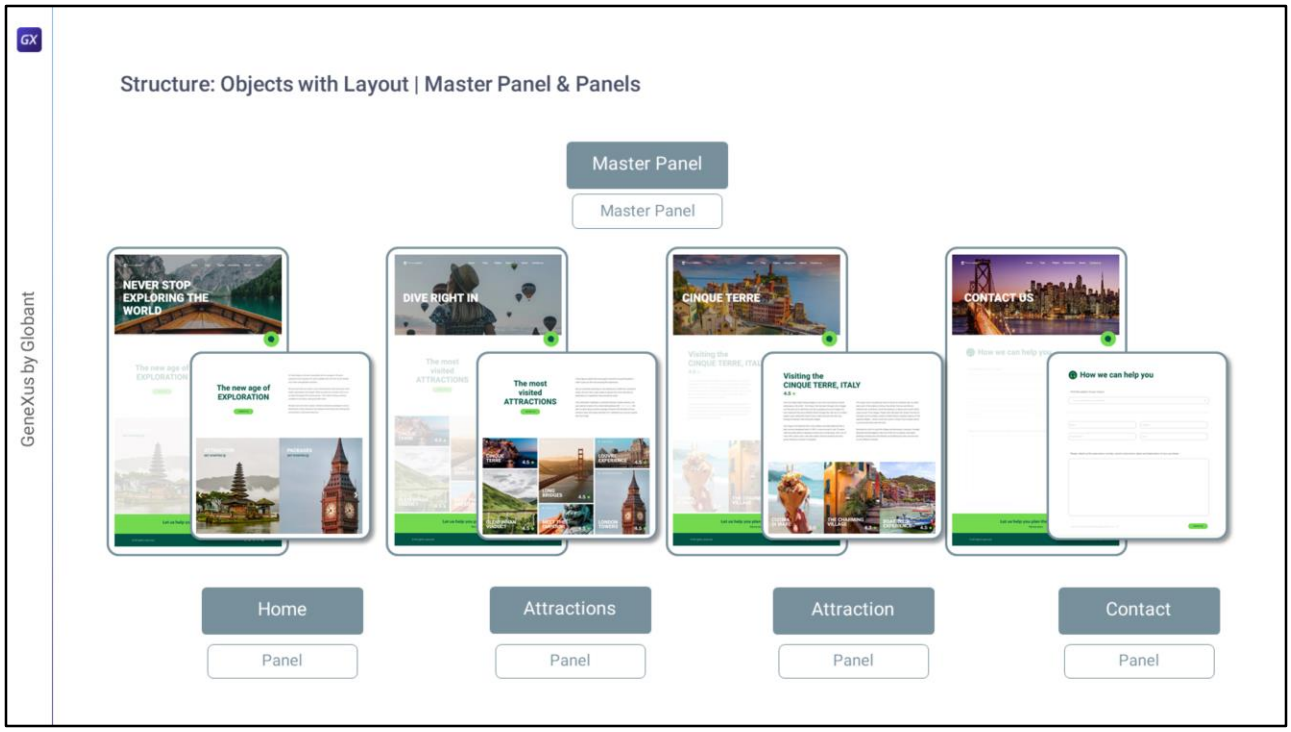
Well, before moving on to something else, you may still have a doubt about the part whose content varies from one page to another, but we said that it would still be implemented in a Master Panel, which is precisely the background image. We see it changes from screen to screen, and the text is superimposed on it.

Well, if we implement them in the Master Panel, these two elements, with this variable image and this variable control, then the answer is easy, it will come from varying the content of these controls according to the object that is being loaded each time in the ContentPlaceholder. You can already imagine that this is going to be done by code in the events section. We are going to see it later in the course.



Going back... with the Master Panel we can begin to make a first structuring: dividing each screen into two: one with everything that is repeated, in this case header and footer, and another with the specific content of each page, which will be modeled in a Panel object that I have called “Home” for this screen.

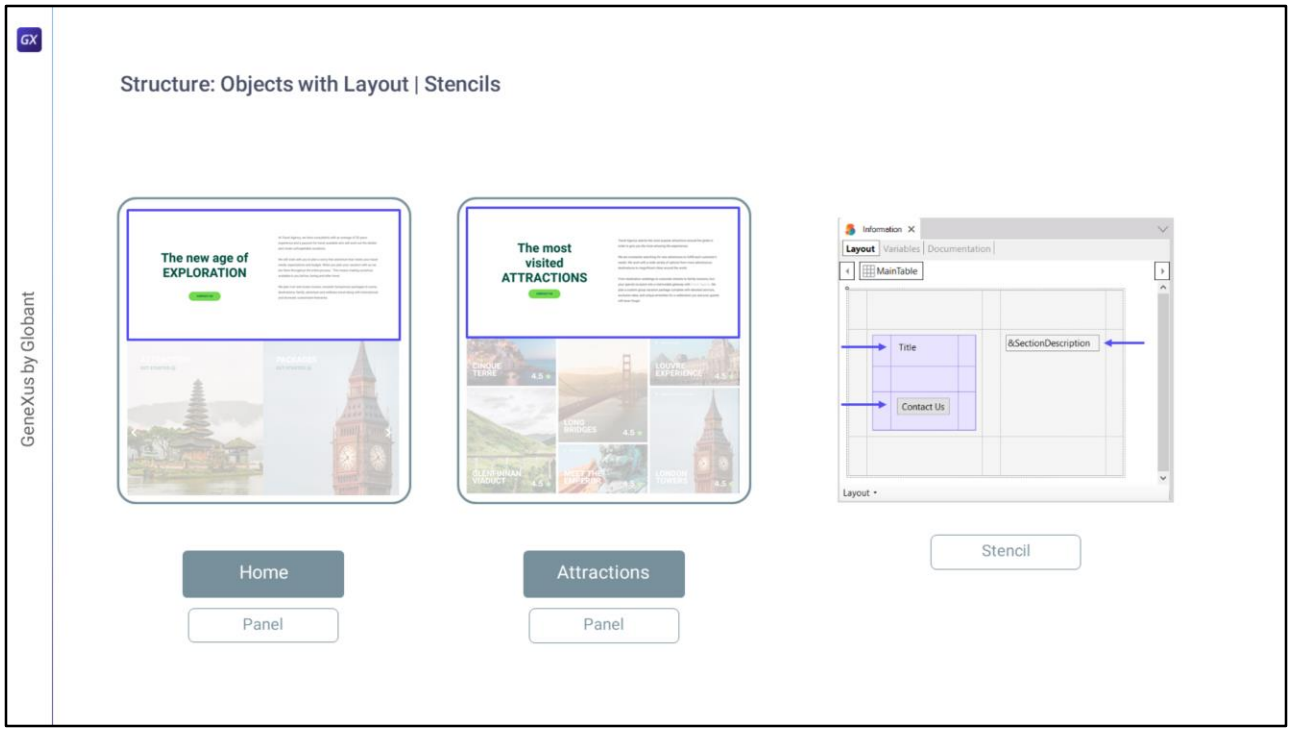
For each screen the same thing will be repeated: the division into the Master Panel and the Panel.



So, there will be only one Master Panel object left (here I named Master Panel the object that is of Master Panel type; I could have called it TravelAgency, or any other way, but I wasn't very creative), as well as 4 panel objects that will be loaded in the contentplaceholder of the Master Panel.

We've made this first division of the screens into Master Panel and Panels.

But we are not going to stop here, because if we continue analyzing the screens we can notice that there are more structures that are repeated, although the content varies, just as we saw before, and that, therefore...



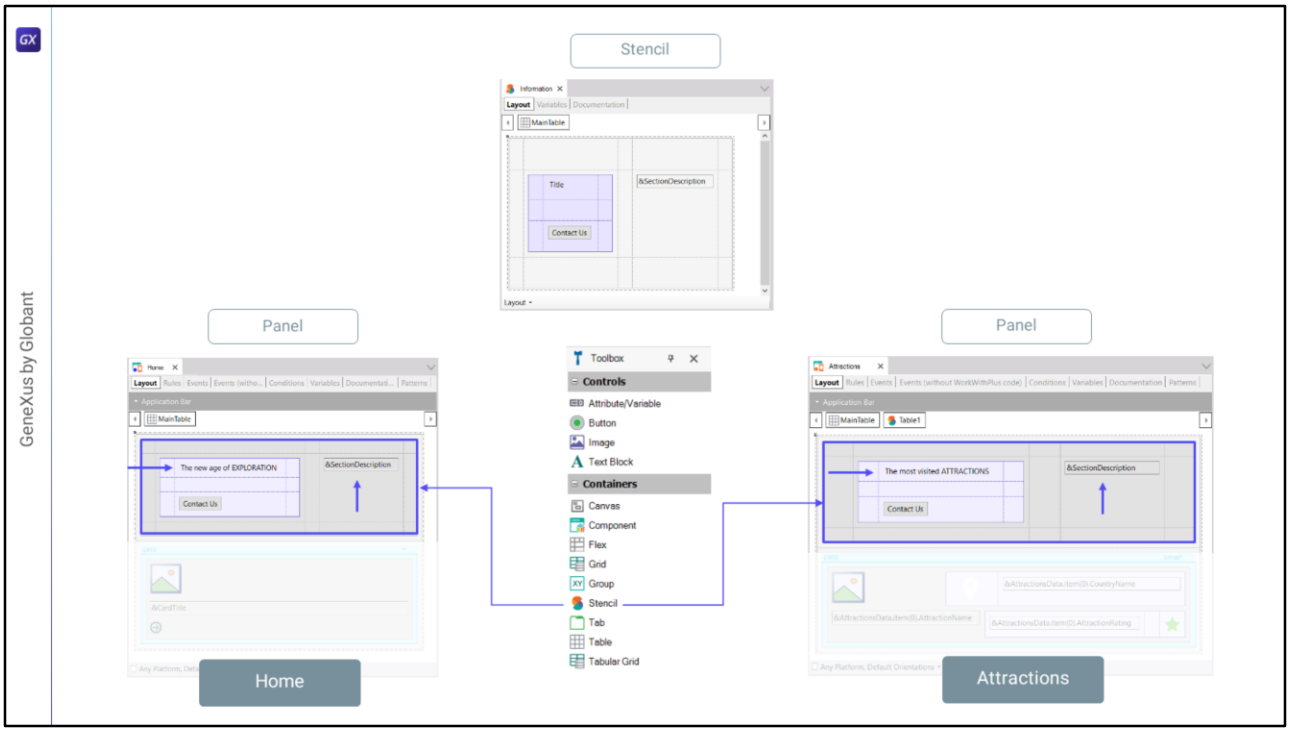
... these parts can be isolated to define them once and reuse them as many times as needed. It is the case of what happens between this part of the Home panel and this part of the Attractions panel.

They have the same visual structure and behavior (and I say behavior because here is a button that will call exactly the same screen in both cases), so I can implement all these sections that are repeated with a Stencil object, which is used just for that.

That is to say, it is a GeneXus object with a Layout, but unlike the other ones that we saw, this one cannot be run by itself. To be used and run it has to be instantiated.

In this way what I do is, in the stencil object, define the structure and appearance without worrying about the content, which will be specific to each instance.

That's why there we see a textblock, a button, and a text variable. The textblock has a generic caption, and the variable has nothing loaded, it is just a text variable.

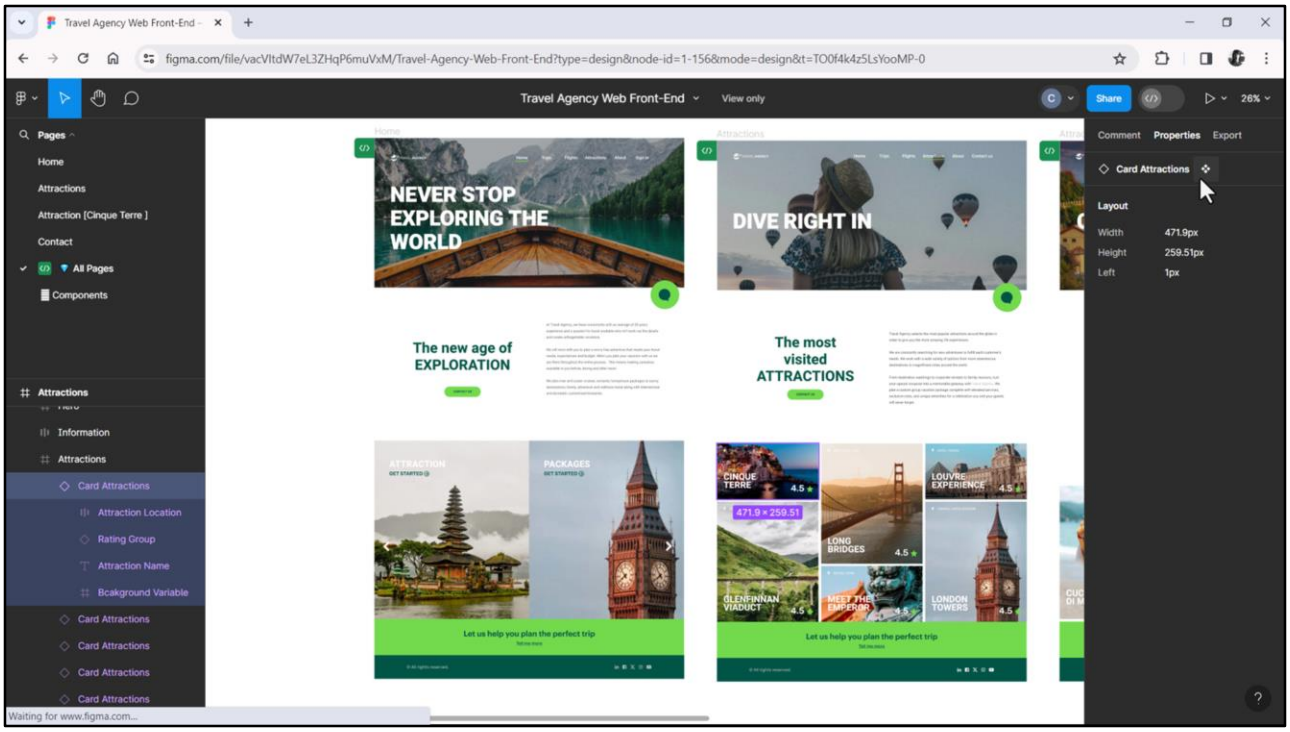


And then what I do is **instantiate** the stencil in the panels where I need it.

Thus, in the first row of both panels we insert a stencil **control**, where the content of the stencil **object** will be loaded, but where we will be able to modify those aspects that have to do with the content. For example, I'll change the caption of the textblock in each of these panels so that it adapts to what I need for each one. For example, this variable here will be loaded at runtime in the events with the particular and specific content.

So then we get another form of reuse. If we need to modify something of the structure or the appearance we do it in the stencil object, once, and that applies to the panels where that stencil is instantiated.

Chechu, our designer, did not identify this repetition...

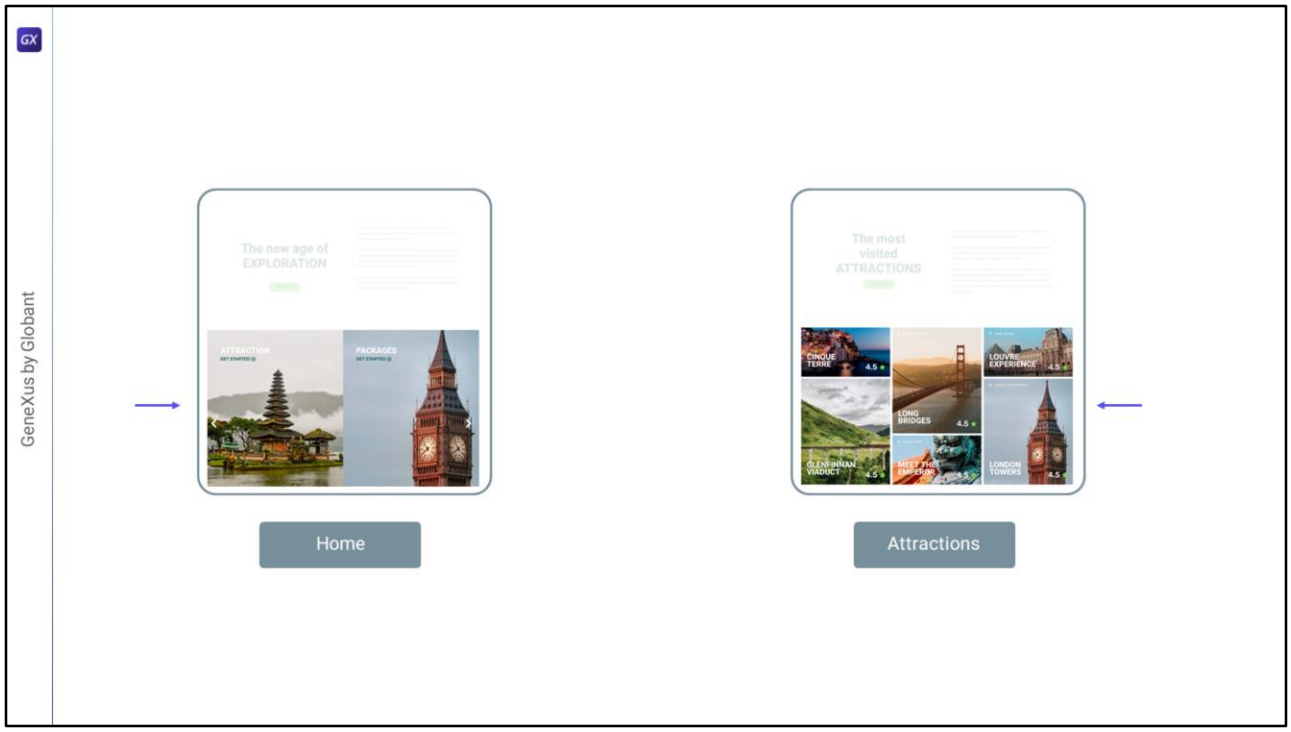


If we inspect both screens, we see that they have a group that she called Information; she called it the same in both cases, but did not componentize it... and that's why it doesn't appear in the components page, as we are going to see it will happen, for example, with these cards... Here it indicates that it is an instance of a component and there I access the component; in fact, it takes me to the components page, and I could even return to the instance from which I started. And here I return to this card on the Home page.

Likewise, if I select this other element we see that it is an instance of a component... this one here, which is in the components page.

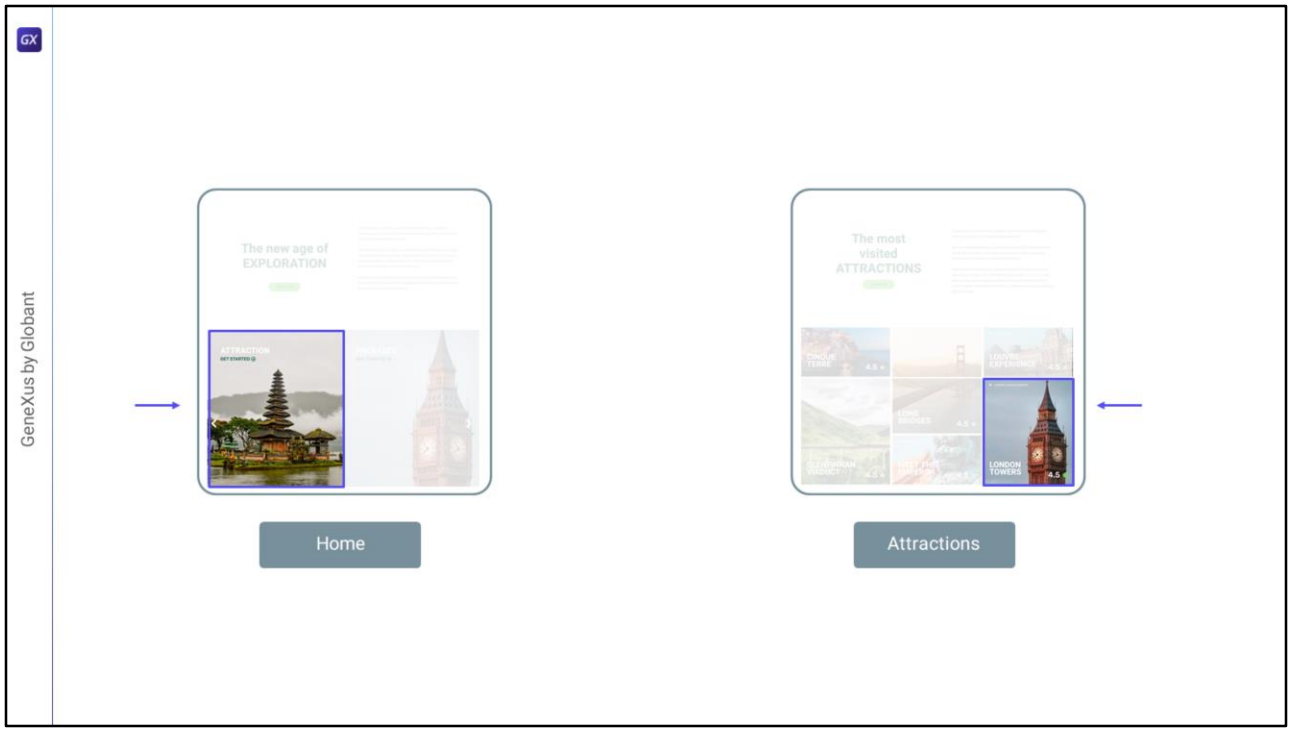
Chechu surely did not componentize the Information group that we were talking about because it was not so obvious to her. We can do it. It wasn't obvious to her, it was obvious to us.

However, it was obvious to her that she had to componentize these cards, so, having identified these cards as components gives us the clue that we may componentize them too, in our case by creating stencils to reuse them. So...



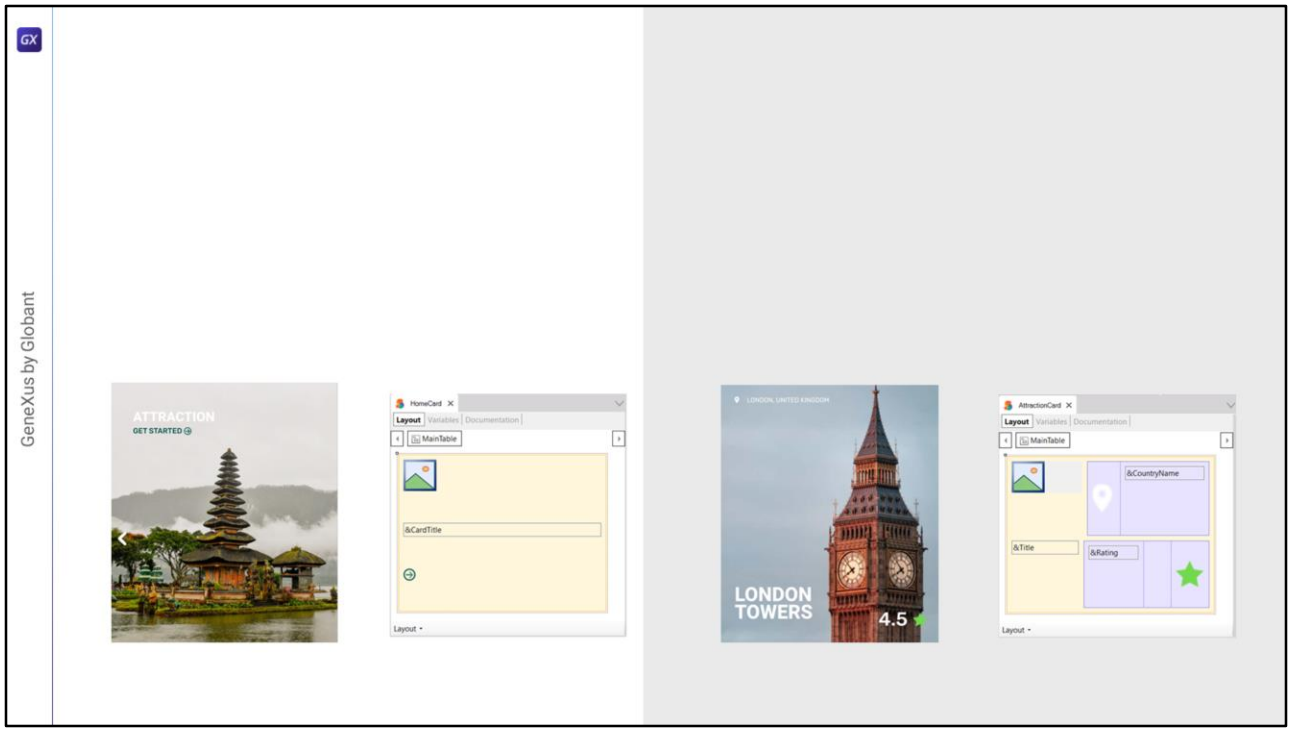
Looking at these two panels, we can see two carousels here where these cards are repeated.

In other words, we are going to need a repetitive structure to model the carousels, and we can already anticipate that they are going to be grids of different types: in one case it is going to be horizontal, in another case it is going to be a flex grid...

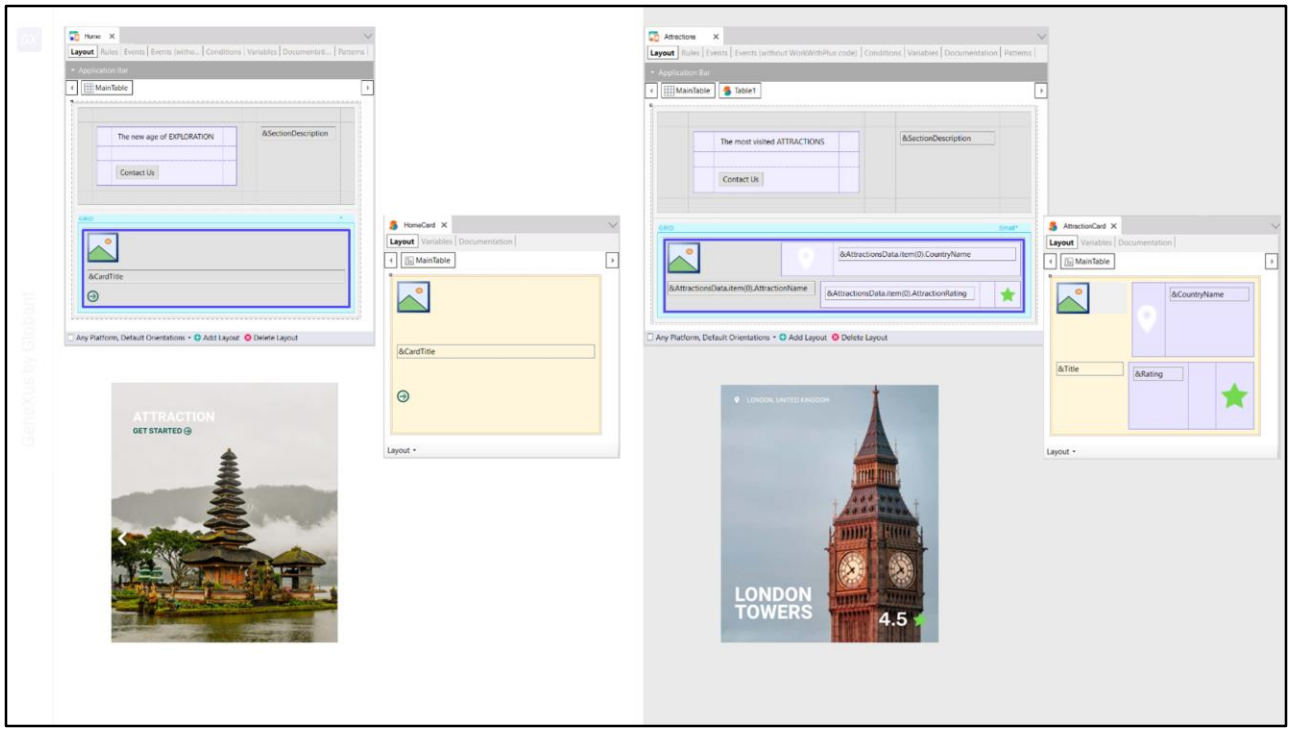


It will be a grid composed of elements that will be of these cards' type. Since their style and content are repeated, we can already anticipate that it would be interesting...

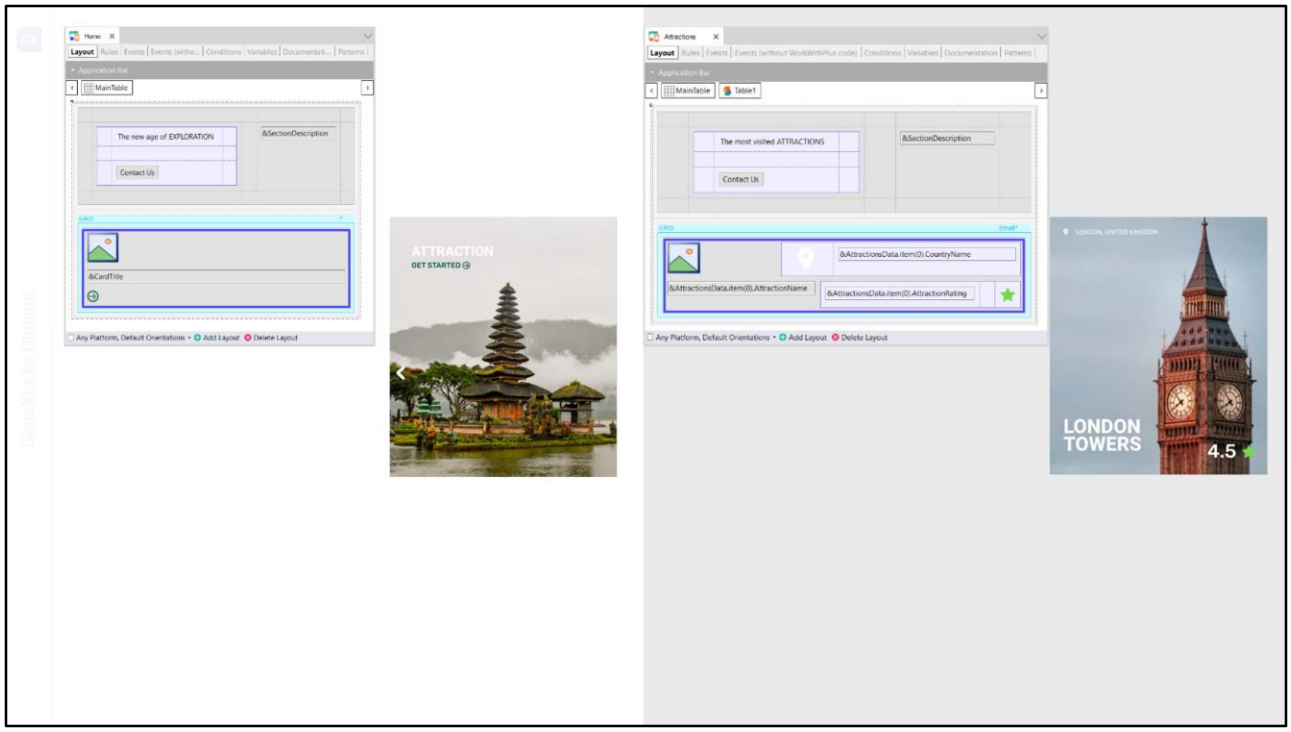
...to model them as stencils.



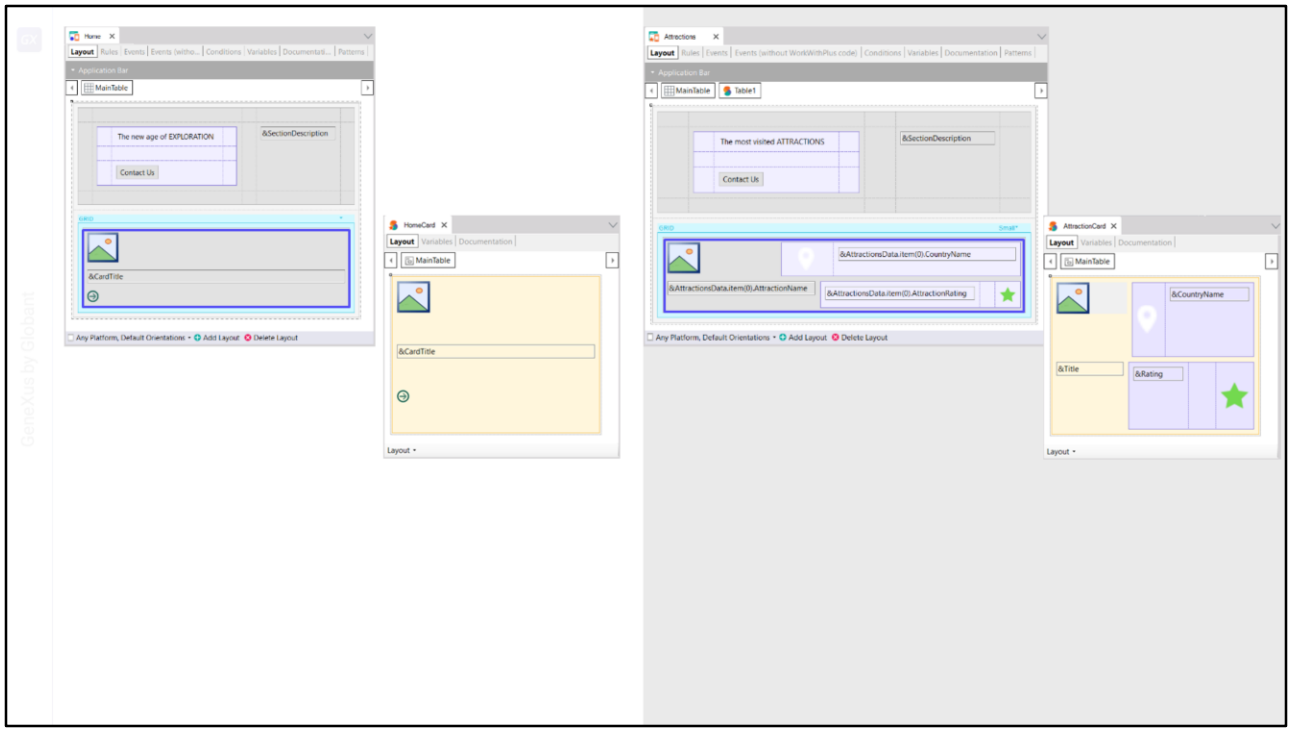
Therefore, what we are going to do is to create a stencil for each one; there we are going to model its layout, and then...

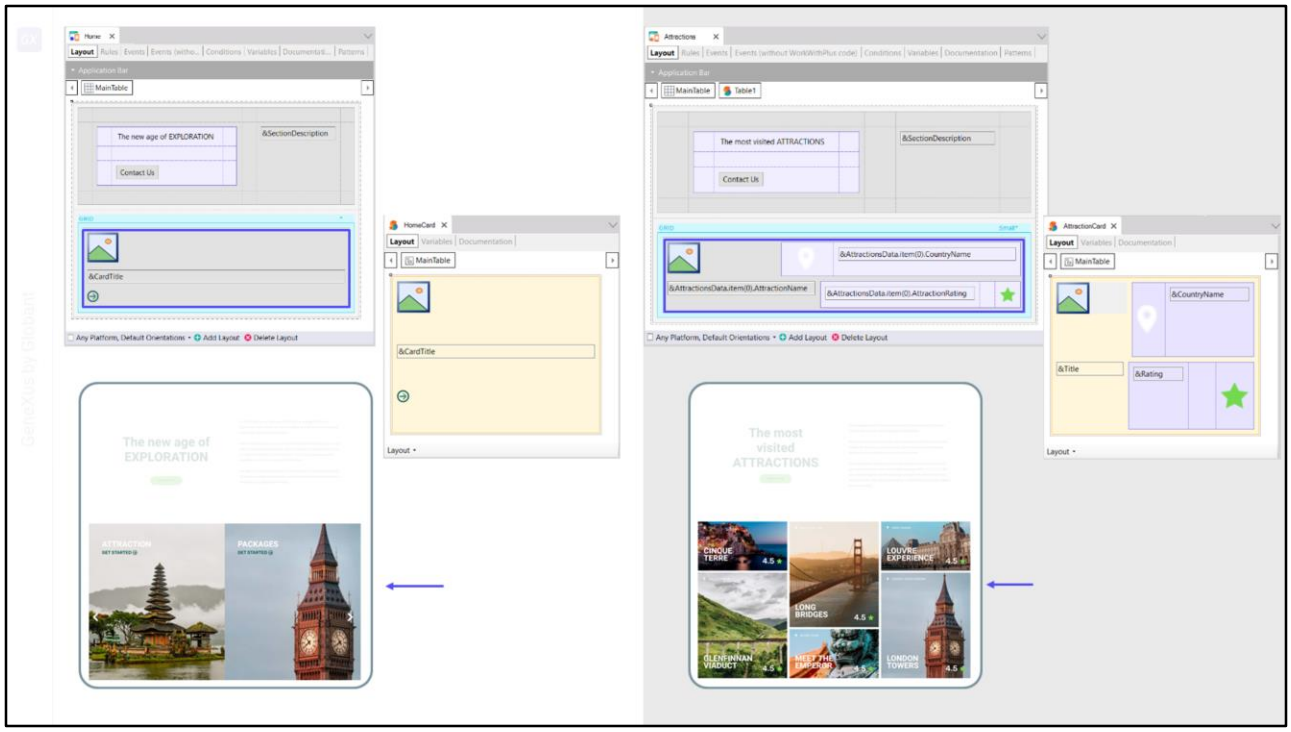


...simply as we saw before we're going to instantiate each stencil in the panel, as elements of each of those grids.

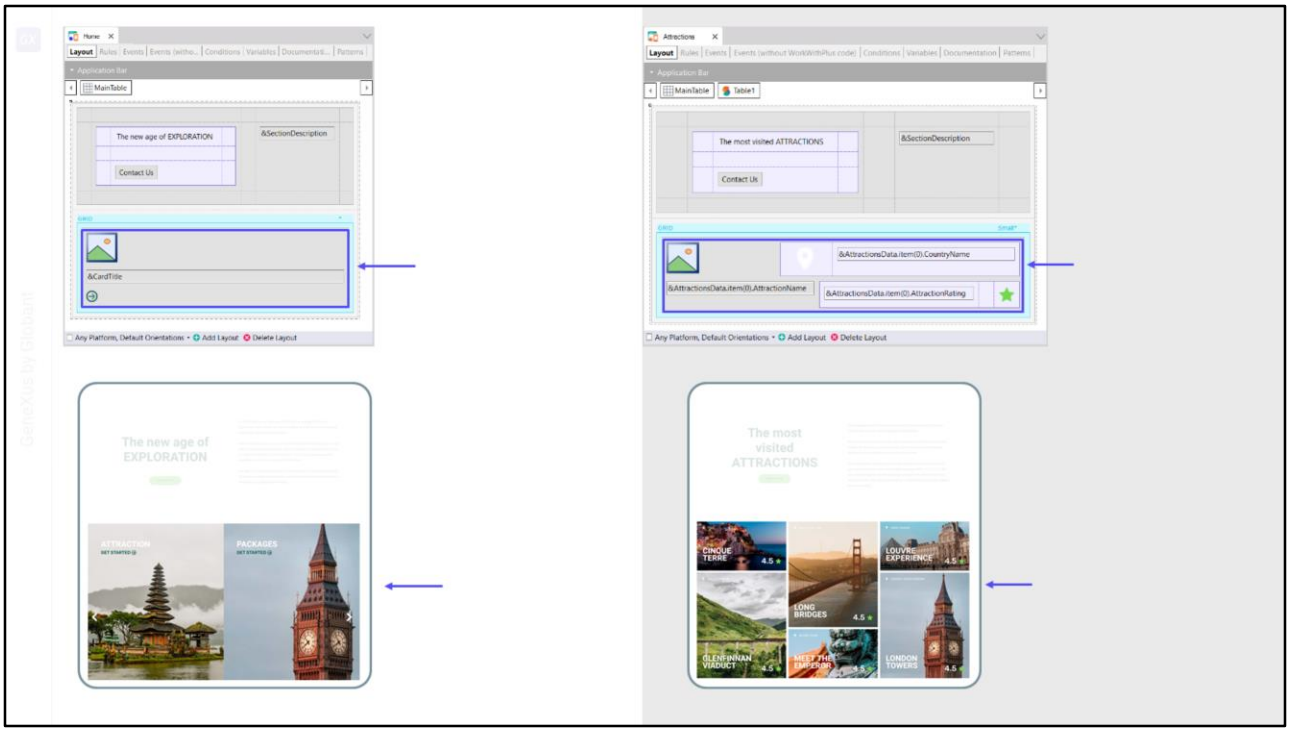


So, by doing this...

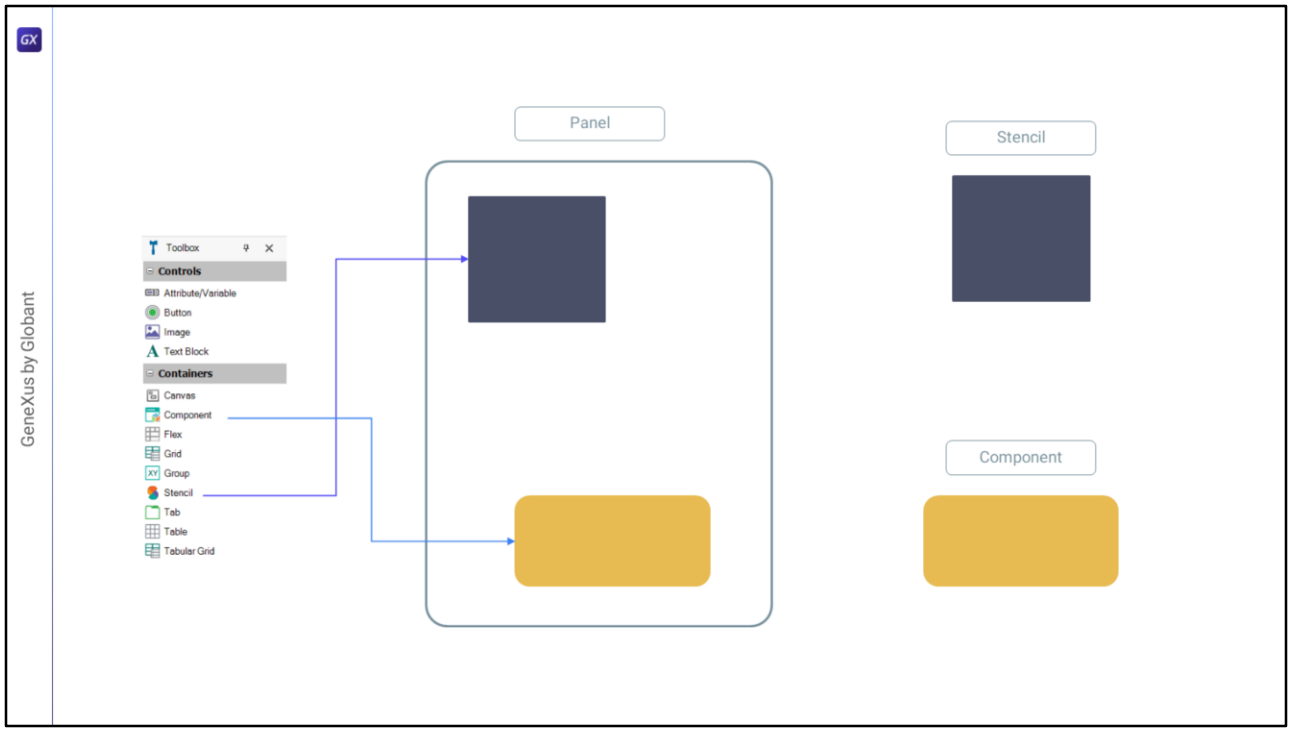

Home
Layout
Rules
Events
Events (witho...
Conditions
Variables
Documentati...
Patterns
Application Bar
MainTable
The new age of EXPLORATION
Contact Us
&SectionDescription
&CardTitle
Any Platform, Default Orientations
Add Layout
Delete Layout
HomeCard
Layout
Variables
Documentation
MainTable
&CardTitle
Layout
Attractions
Layout
Rules
Events
Events (without WorkWithPlus code)
Conditions
Variables
Documentation
Patterns
Application Bar
MainTable
Table1
The most visited ATTRACTIONS
Contact Us
&SectionDescription
&AttractionsData.Item(0).CountryName
&AttractionsData.Item(0).AttractionName
&AttractionsData.Item(0).AttractionRating
Any Platform, Default Orientations
Add Layout
Delete Layout
AttractionCard
Layout
Variables
Documentation
MainTable
&CountryName
&Title
&Rating
Layout



I have the grid modeled as a repetition of those stencils...



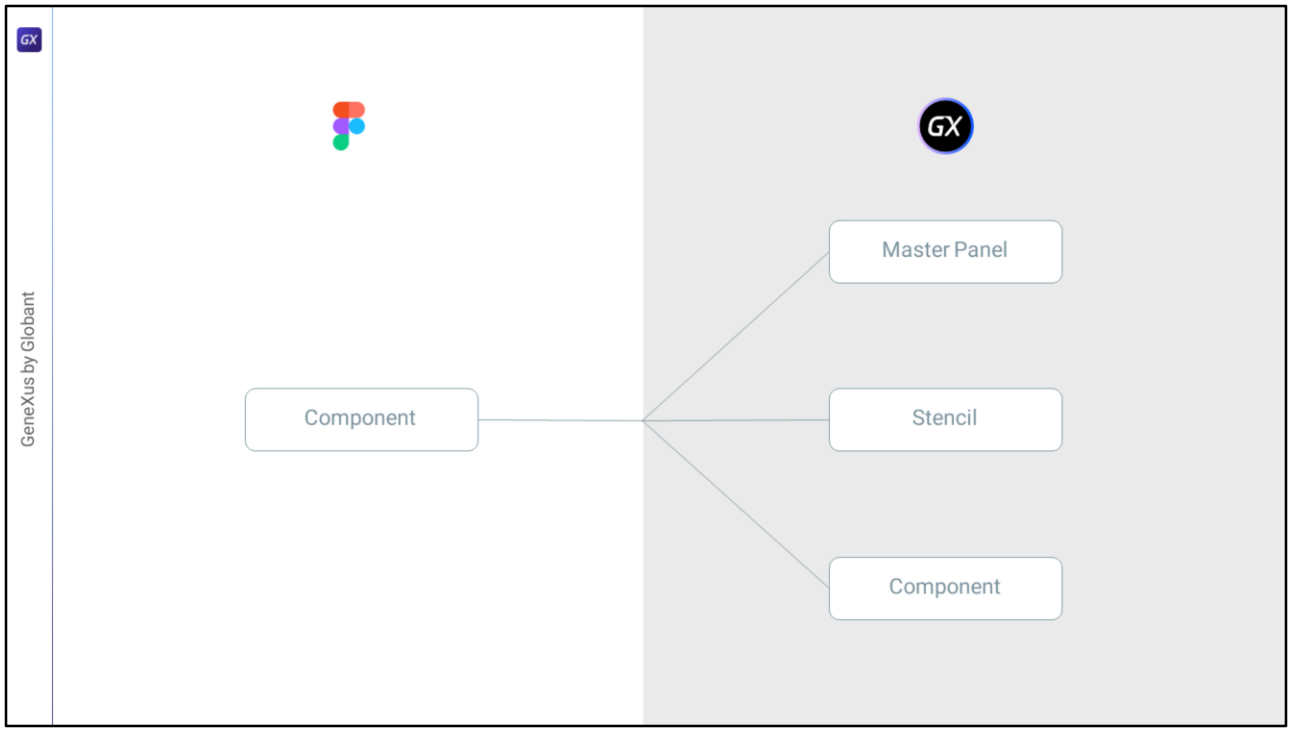
And then what I will need to do is simply, in the events, load each item, each content, with the respective information. In the second case, I'll have to take it from the database.



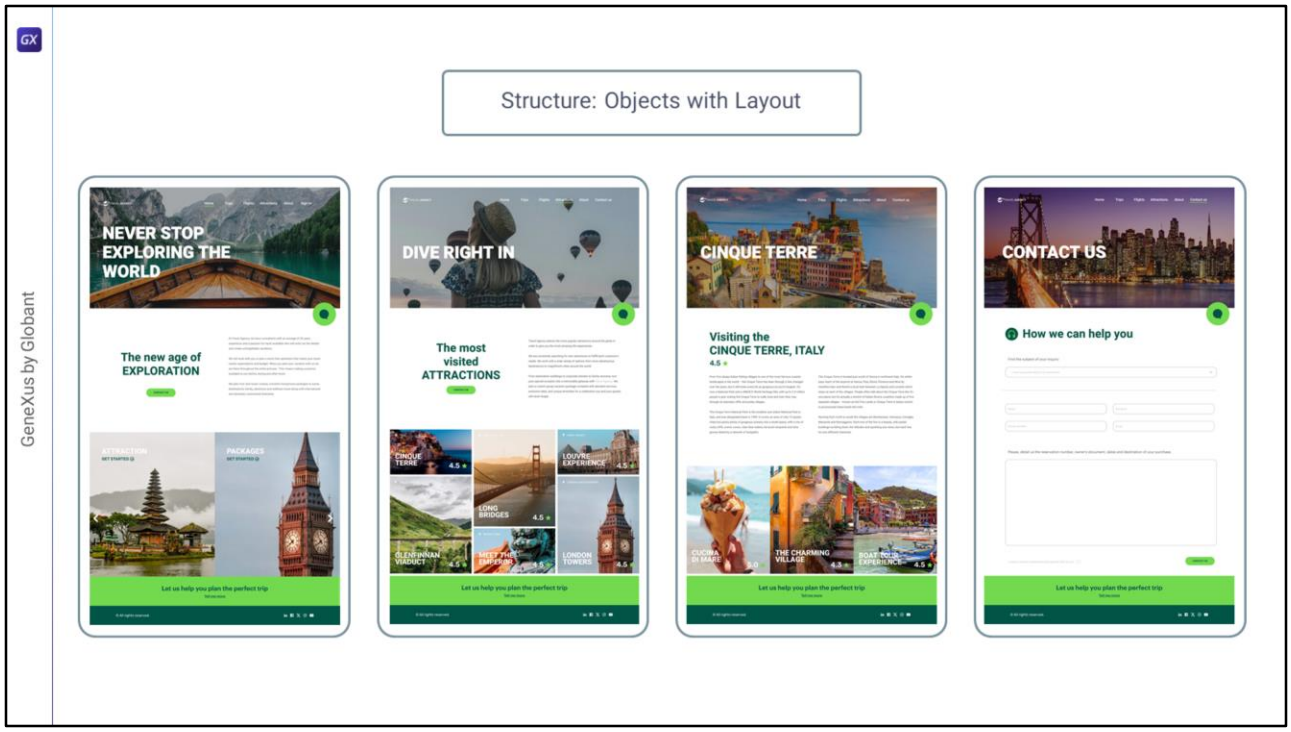
In GeneXus, besides being able to componentize a layout section with design after defining it in a stencil object, as we did so far, which we then instantiated in the layout of the object that needs it, we have another way, which is used especially when behavior is also involved (although stencils today allow some degree of behavior)...

If we are developing in the Web Panels world, it will be with the Web Component type object, and if we are developing in the Panels world it will be an ordinary panel. So there I've identified this as a Component but it will be implemented depending on the world: if it is the Web Panels world with a Web Component, and if it is the Panels world with another Panel. In either case, it will be inserted in the layout where you want to use it through a Component type control.

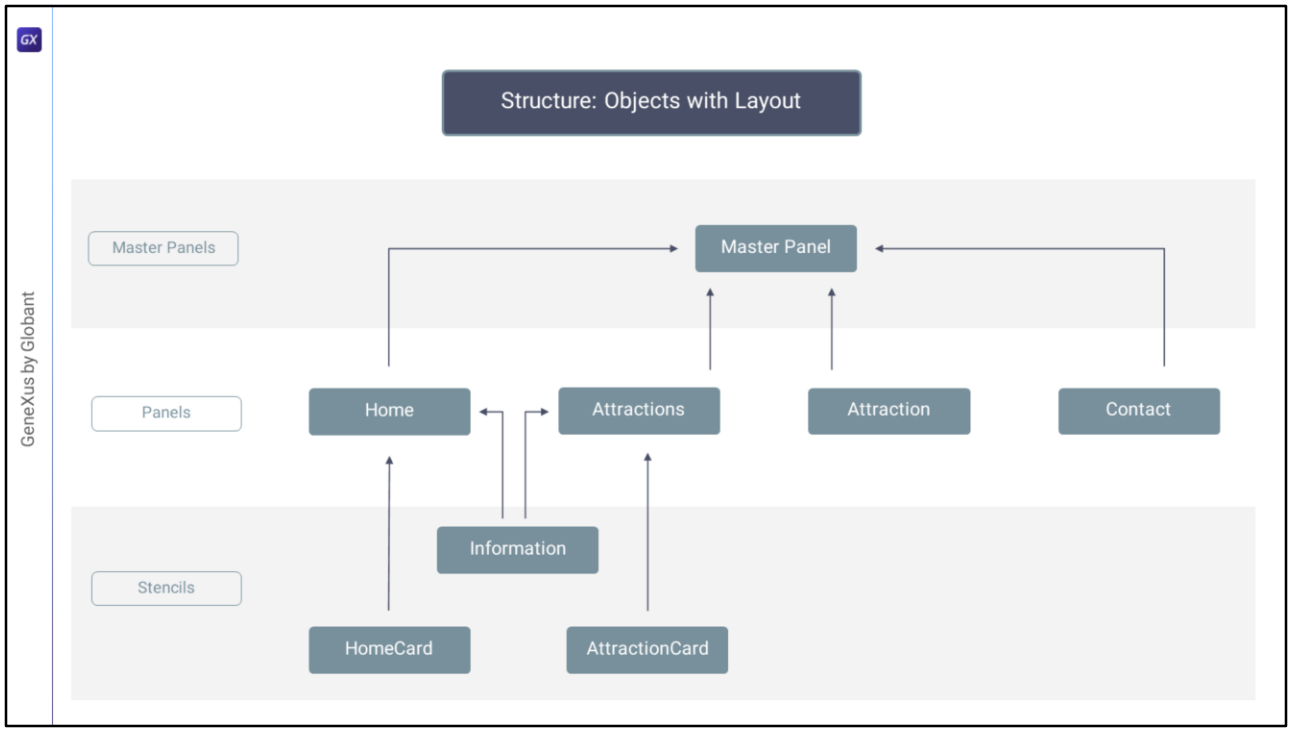
In short, what do I want to show with this? That GeneXus provides several options to isolate part of the screen and model it separately. Actually, we have one more which is the use of User Controls.



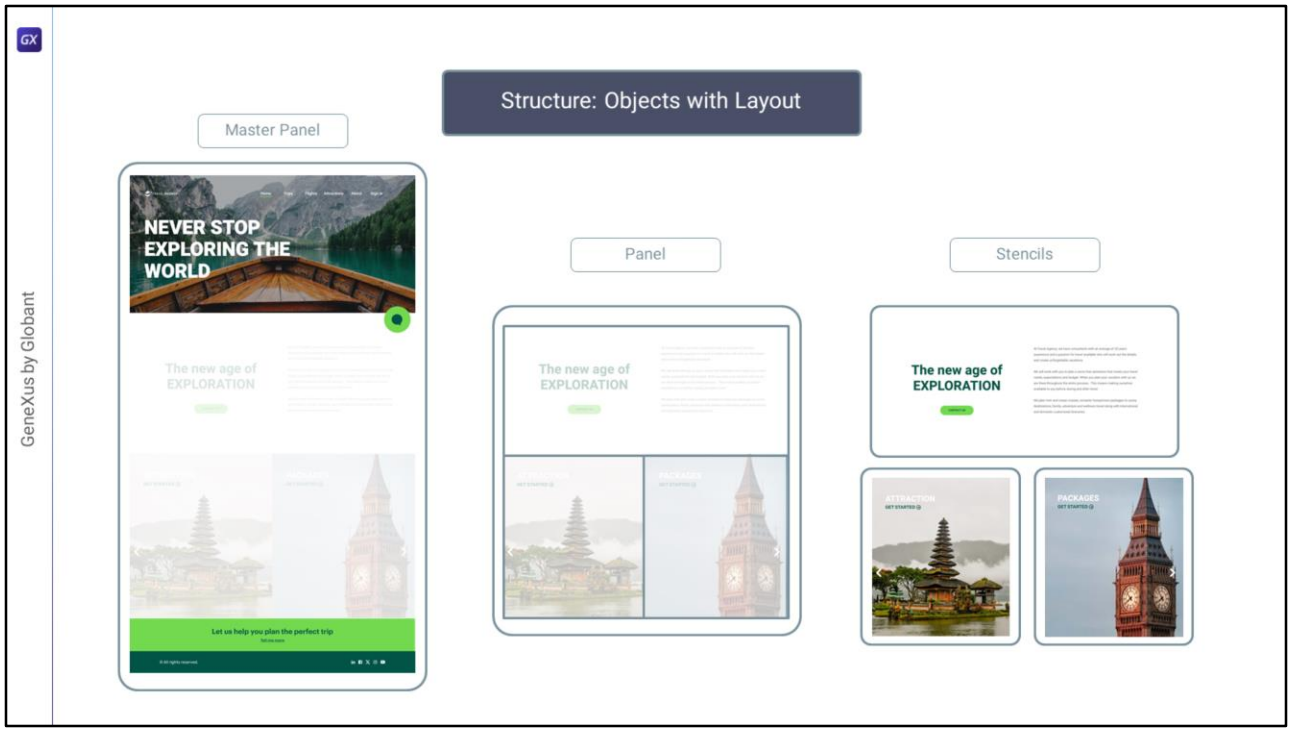
A component that Chechu has isolated in Figma can be represented in GeneXus in several ways: either inside a Master Panel, a stencil, or a component. And also a User Control.



In any case, this first structural division of the application into objects with Layout, which are interrelated, is now clear...

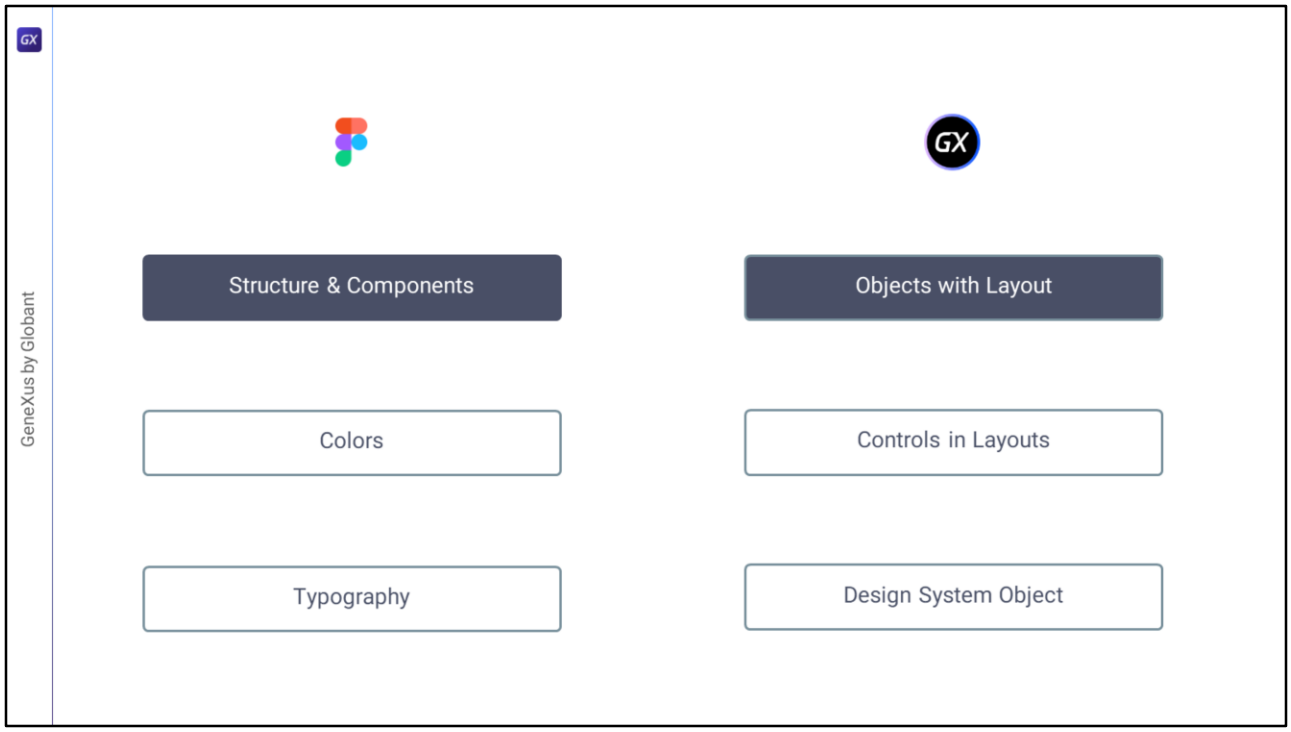


...as we can see in this scheme that I made here, showing the Master Panel, the four panels that we have identified so far, and those three stencils that we have seen.

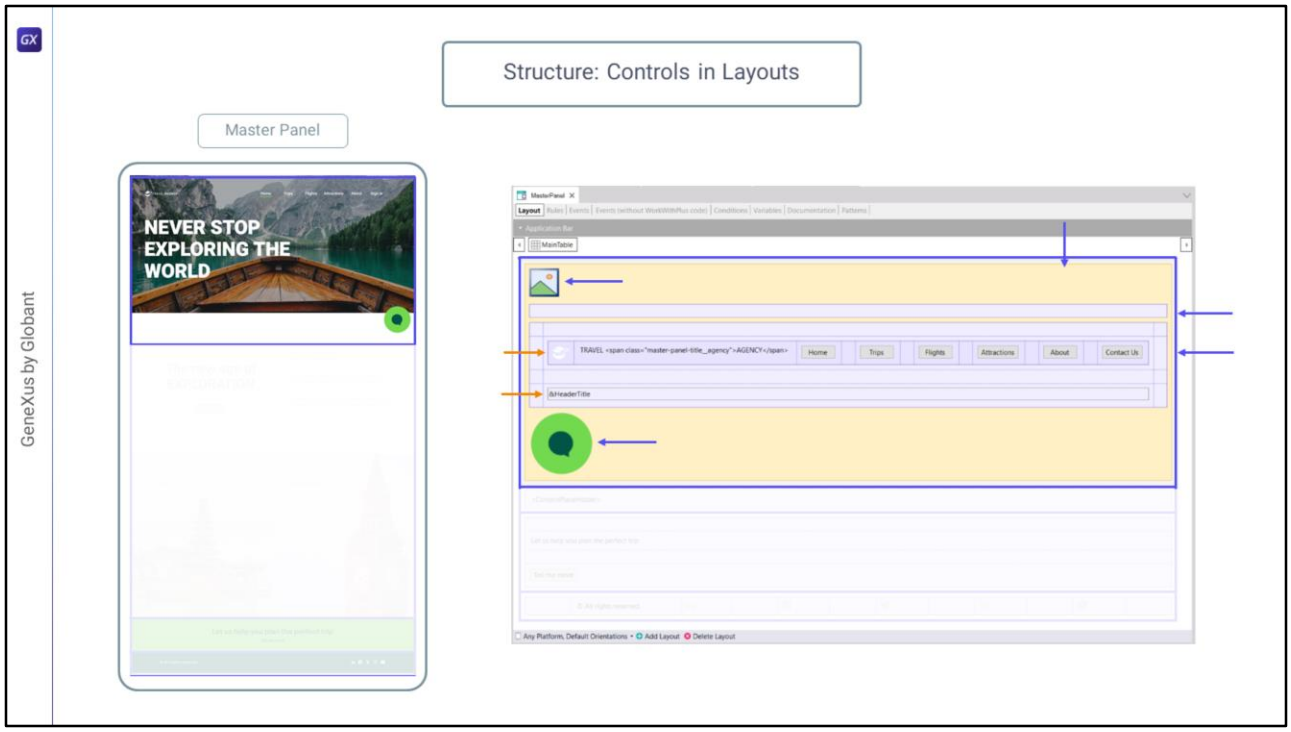


If we take the first screen... it is divided first into Master Panel and Panel; but in turn the Panel is also divided into its controls, such as the grid for example, and Stencils.

The problem, then, is now reduced to a smaller one but with the same complexity: how do I implement the layout of each of these objects?



And now we are going to move on to the second item, the controls in layouts. How do we organize the layouts so that the structure reflects what is going to be the structure of the screens?

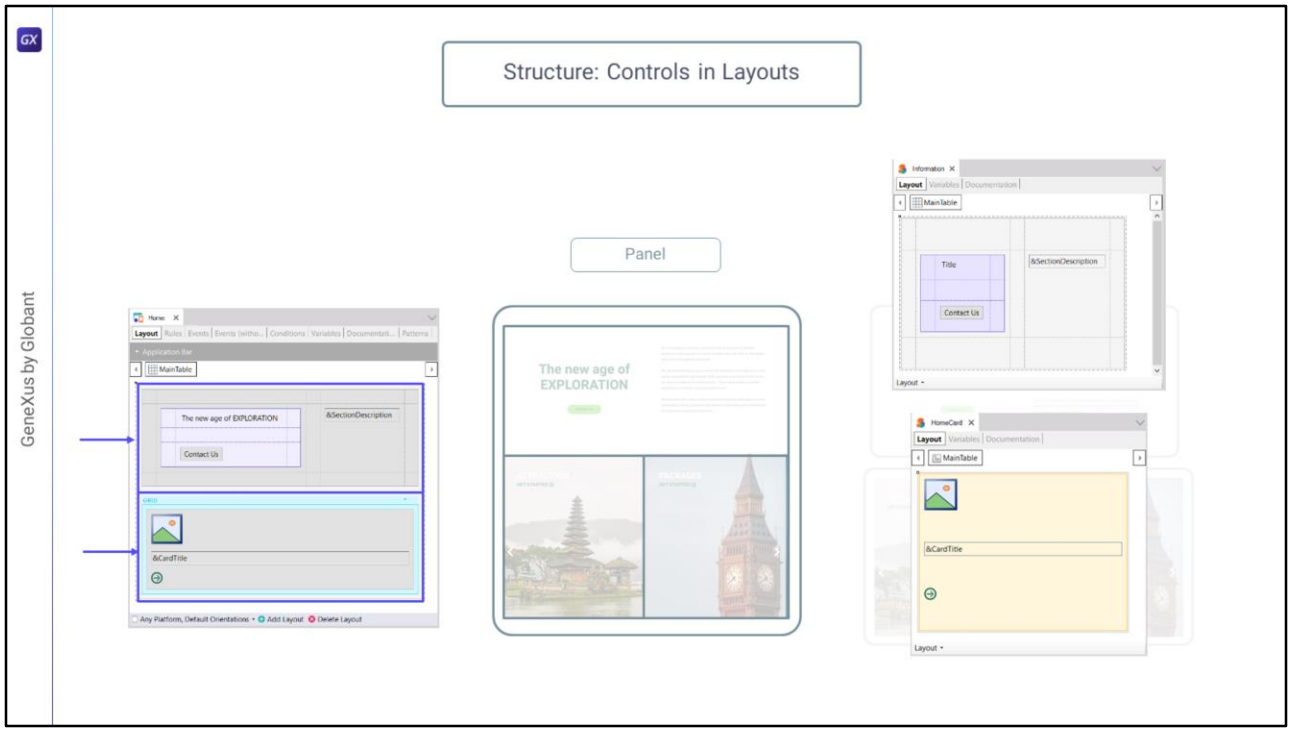


The question then is how to structure the layout of each object. Of the Master Panel, of the Panel, and of the Stencil in question.

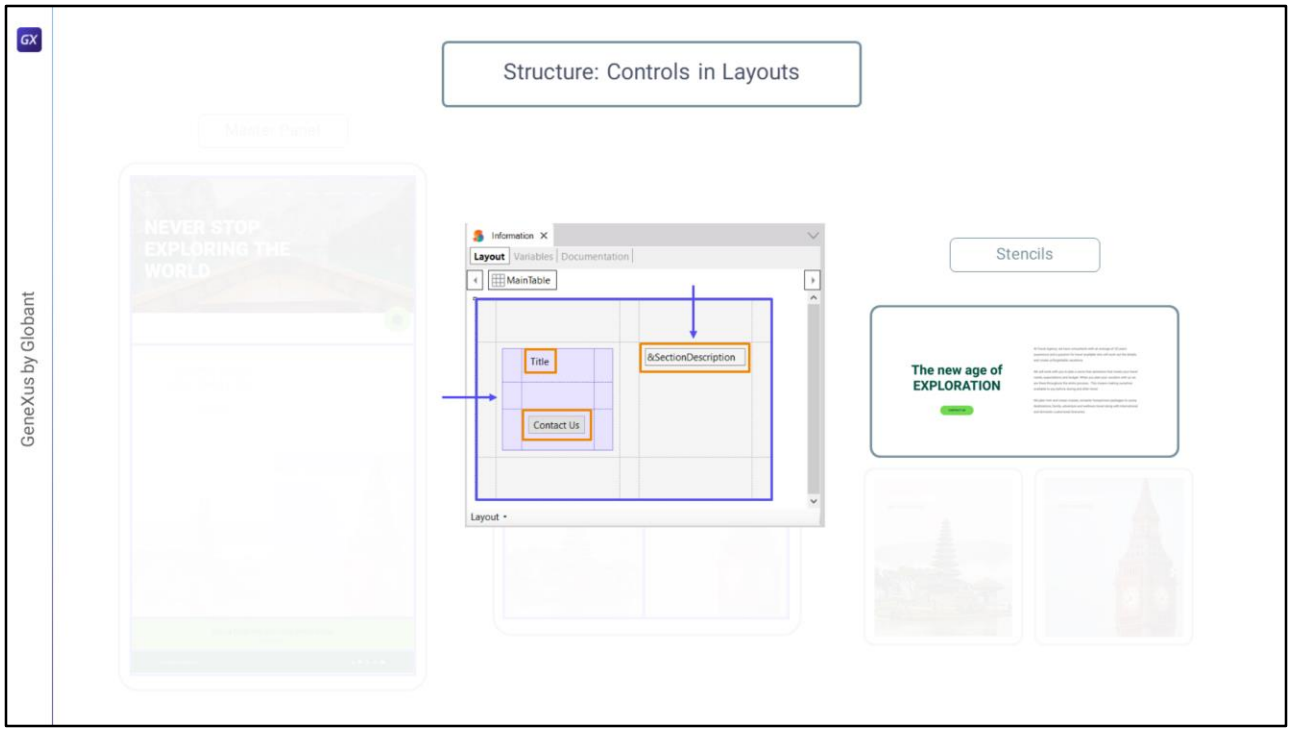
For example, if we analyze the Master Panel: here we are going to have a main table with 4 rows...

Inside each one there are controls that in turn can contain other controls... as in this first row where we have a special table, canvas (the canvas table is the one that allows us to superimpose controls), which in turn contains a control of image variable type, two tables, and a control of image type.
The second of these tables in turn contains another table and a variable control.

And so on... That is, we have a hierarchy of controls, where the parent of all is the main table.



Similarly, if we analyze the Home panel, it will also have a main table... this time with 2 rows... The first one will contain a stencil control and the second one a grid that will contain another stencil control.

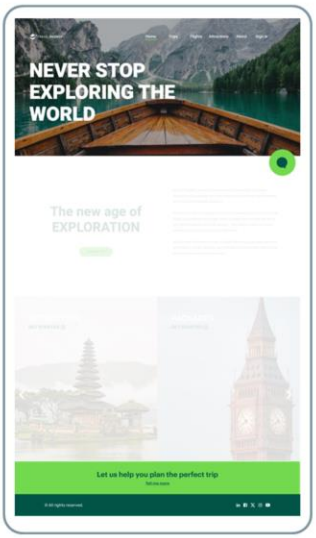


In addition, if we analyze, for example, the first of the stencils... again... we will have a main table with 3 rows and 1, 2, 3, 4, 5 columns. The cell in row 2, column 2 will have in turn another table, which will contain a text block control and a button inside. And the cell in row 2, column 4 will have a variable control.

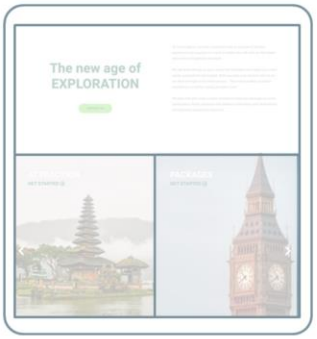
GeneXus by Globant

Structure: Controls in Layouts

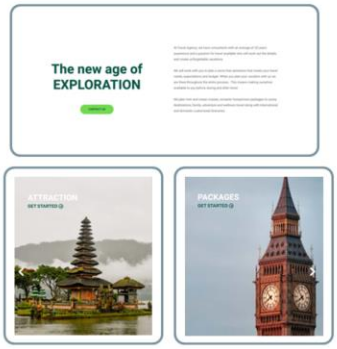
Master Panel

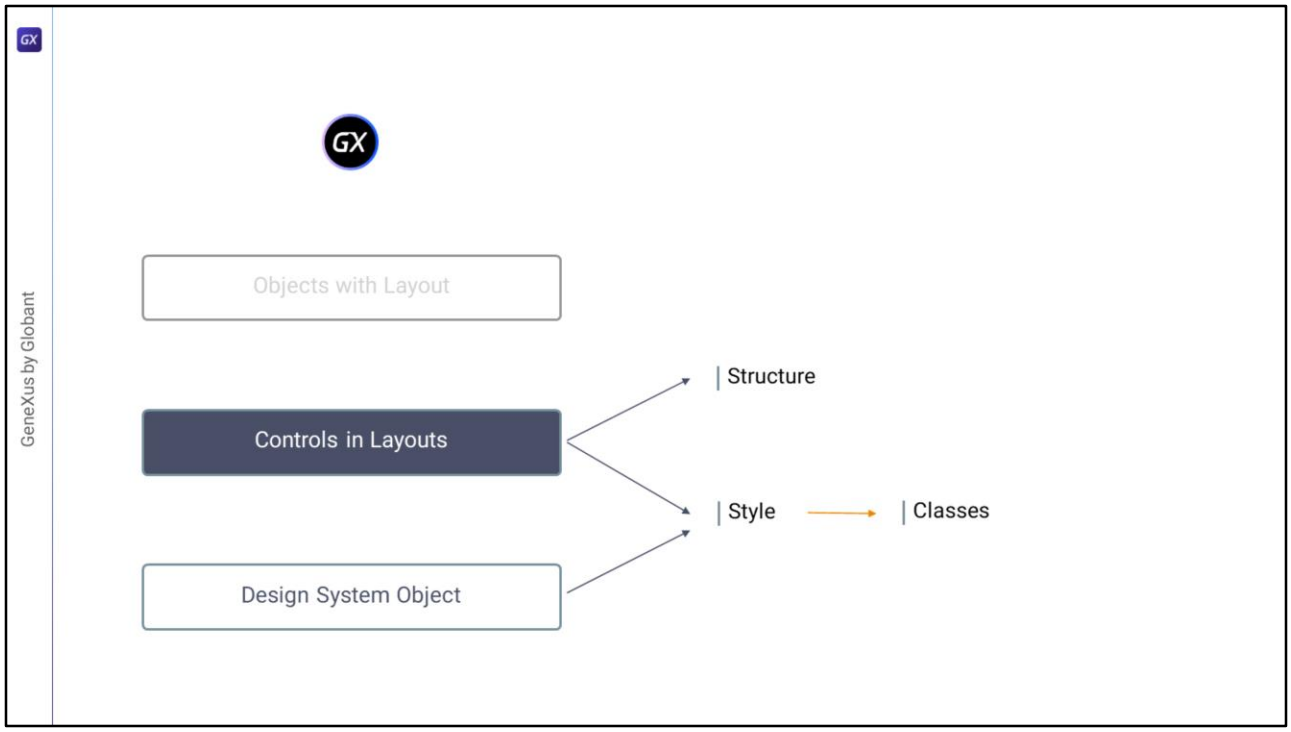


Panel

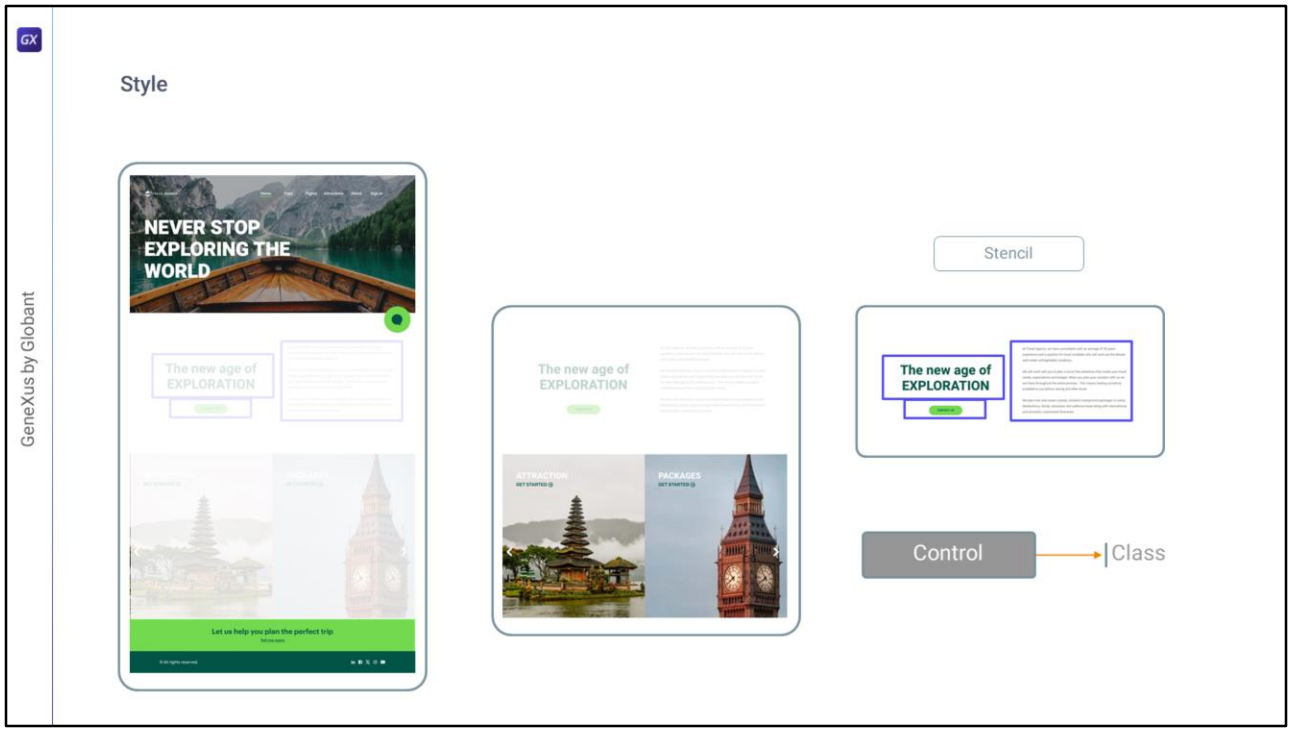


Stencils



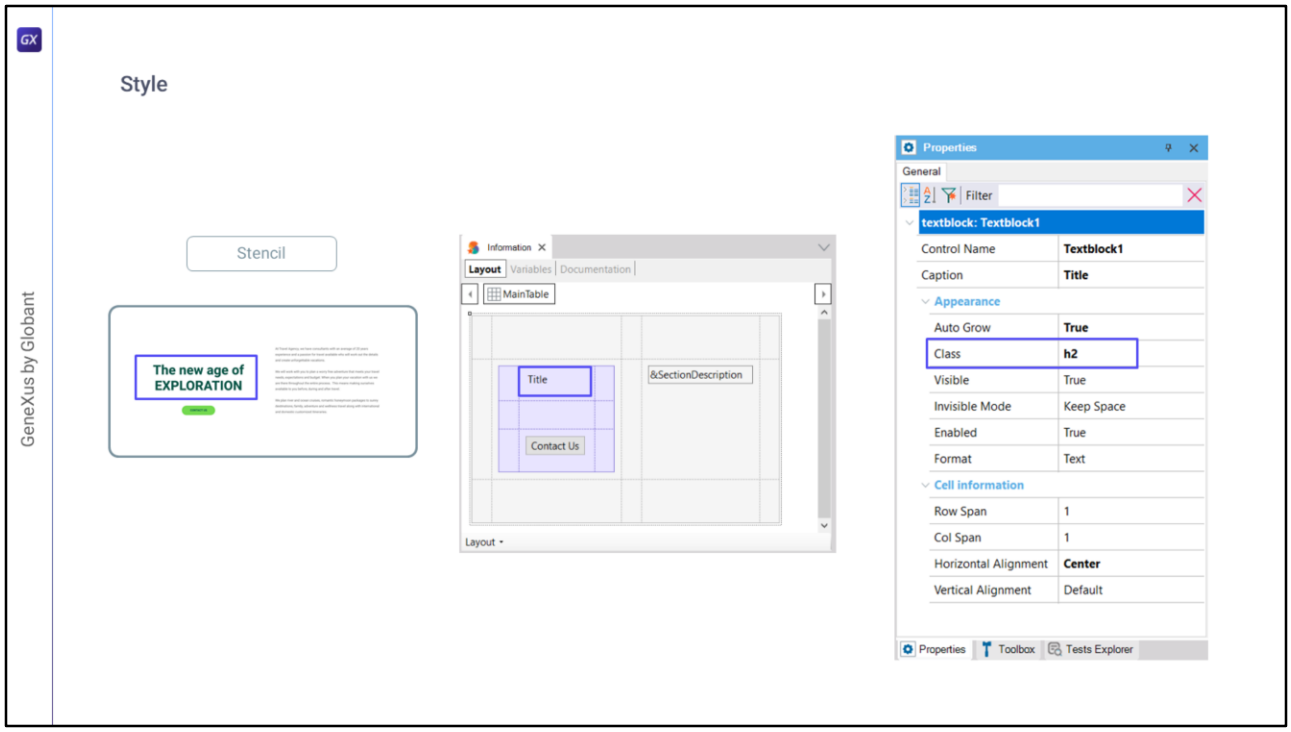


OK, regarding the structural aspects we have controls in their hierarchical order, and for the style –that is, for the elements to look a certain way– we have their properties and, mainly, those of their classes. And that is where the Design System object comes in.

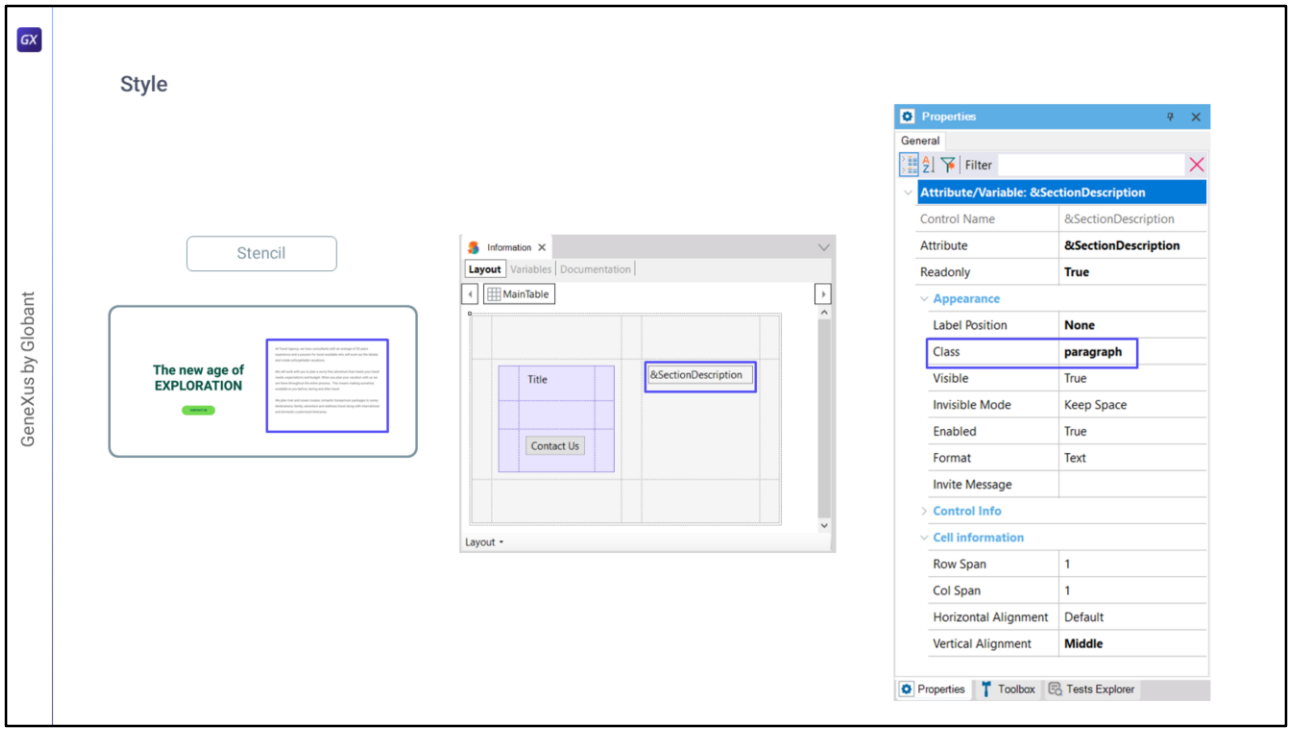


So, for example, to have this text displayed with this font and this green color, this paragraph displayed with this font and this color, and this spacing and so on, and for the button to be shown in this way...

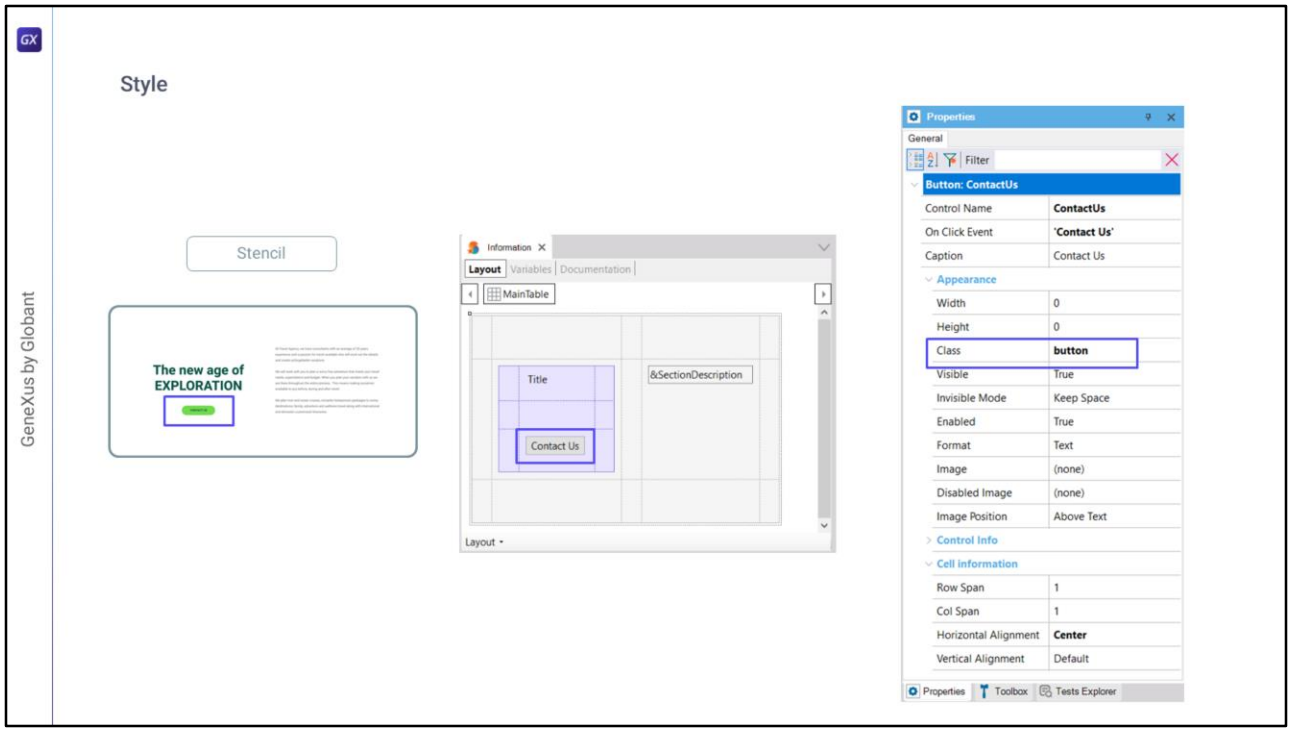
...in the stencil where this is implemented it is enough to associate each control with a class...



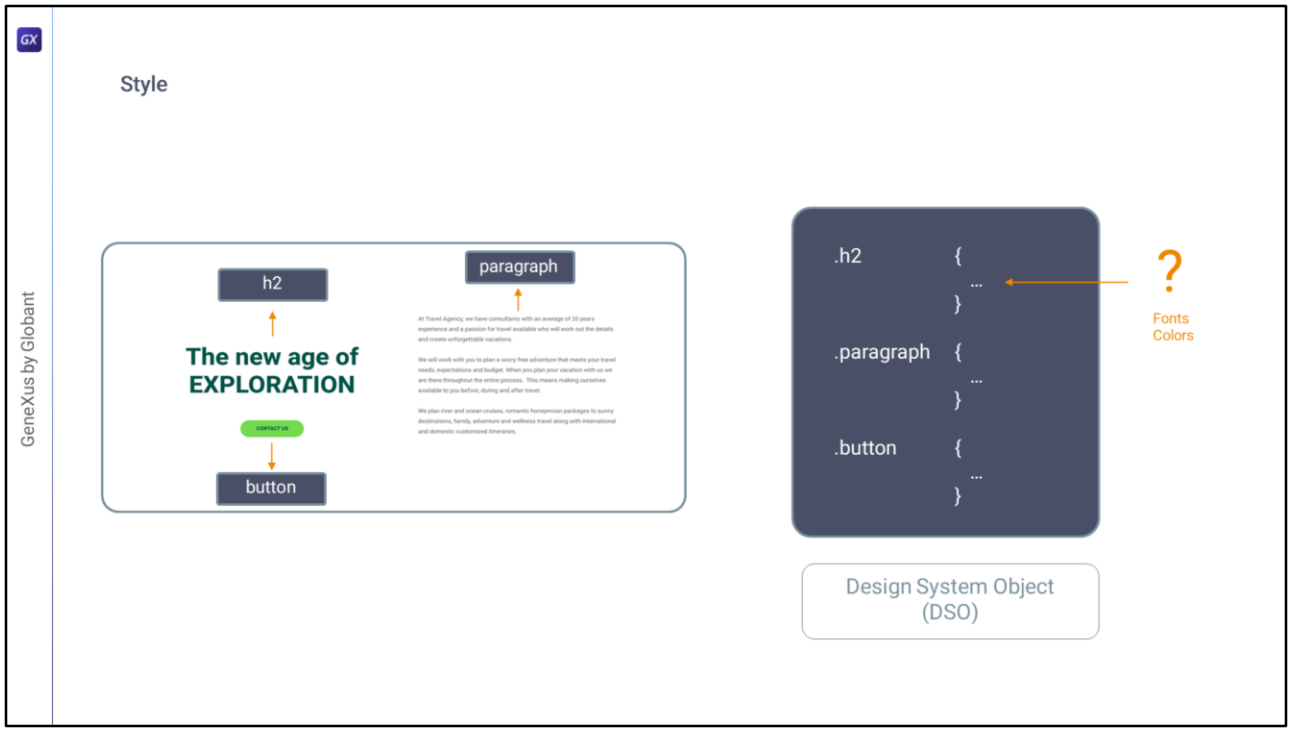
So, to the first control, to this text block, I will associate a class that I will call h2 (we'll see that I didn't invent it from nothing, because Chechu paved the way for me by defining an h2 style in Figma that will be reflected as a class in GeneXus)...



... for this variable we are going to define a paragraph class...

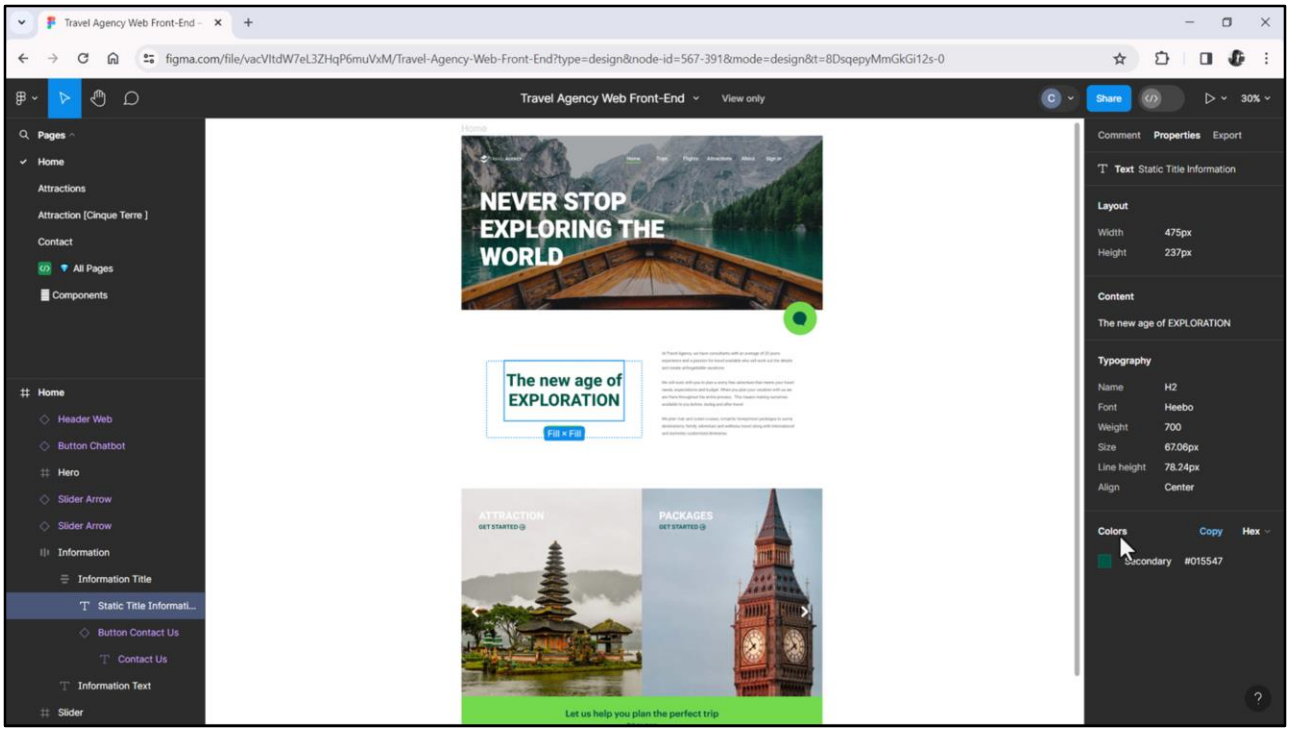


and to this button I associated the class of this name.



Once I've associated classes with each of my controls, it will be enough, in the Design System object that will be used to style all the objects, to define the characteristics of each of these classes (such as font, color, spacing, and so on).

As we know, in some cases these classes are identical to those of CSS. And as we can already imagine, we will define them from the Figma design file that Chechu shared with me and had already shown us.



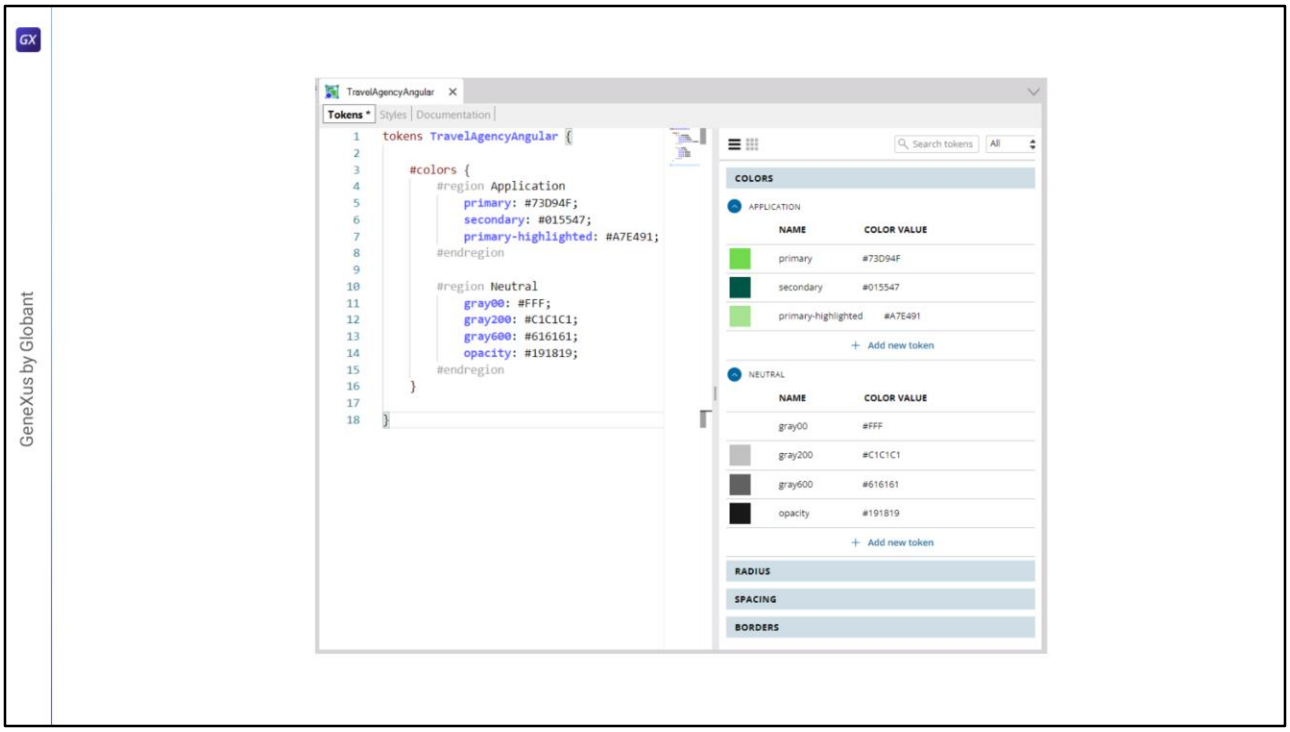
So, if we go to Figma in View mode... and we select this text, with the Control key, we see that the properties window is showing all these properties related to the typography of the text... properties to which Chechu gave a name –h2. This is going to be a style in Figma that Chechu created, which in our case will be transformed into the h2 class. Then we have this Secondary style, with this value, that is going to be a color style. And that is the color that the text is taking.

By right-clicking on the text / copy as / code / CSS, the CSS properties that correspond to that text are copied to the clipboard. I'm going to paste them so that we see them in this Notepad... and here is H2 with its CSS properties, that of color, Secondary and others that later on we will see whether to use them and for what purpose...

The same happens if we select the paragraph; we have the style that Chechu named Paragraph, with these characteristics, this color and so on.
Again, the same for the button; I can get the typography that she called this way, with these characteristics, and so on.

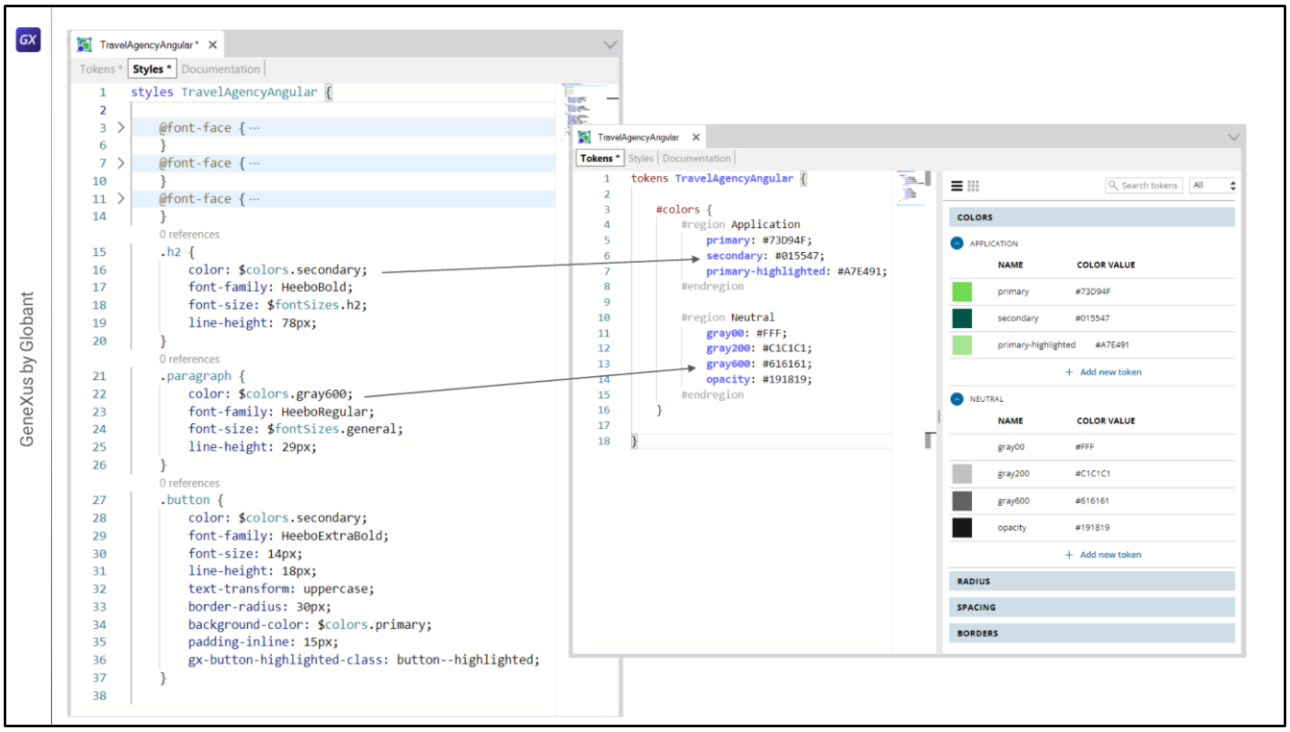
If we go to the All Pages page, for example, outside the frames, we can see that the properties tab lists all the text styles that Chechu created (here is H2 for example, here is Paragraph, and the rest), which are created for all these screens. Also the colors that are being used, with the names that Chechu created in Figma and to which she gave values.

Okay, so with this...

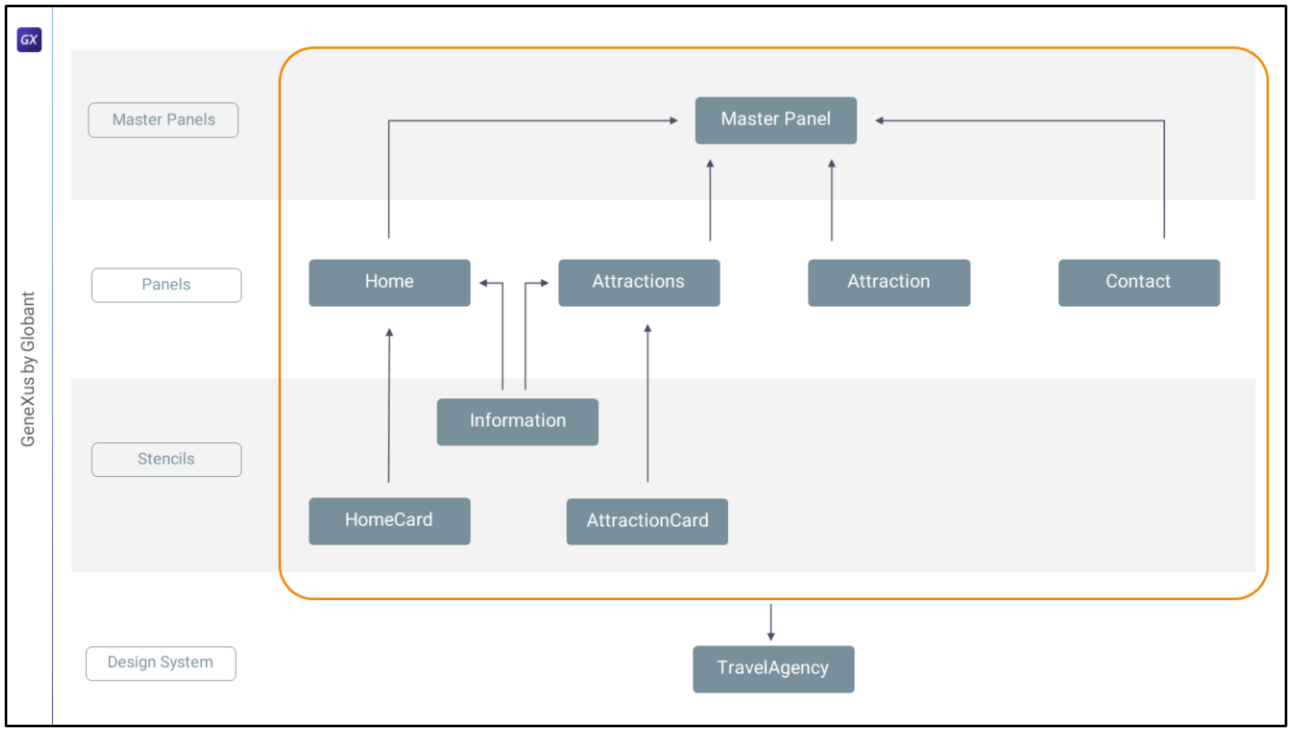


...I go to GeneXus and I can already start to build my Design System object.

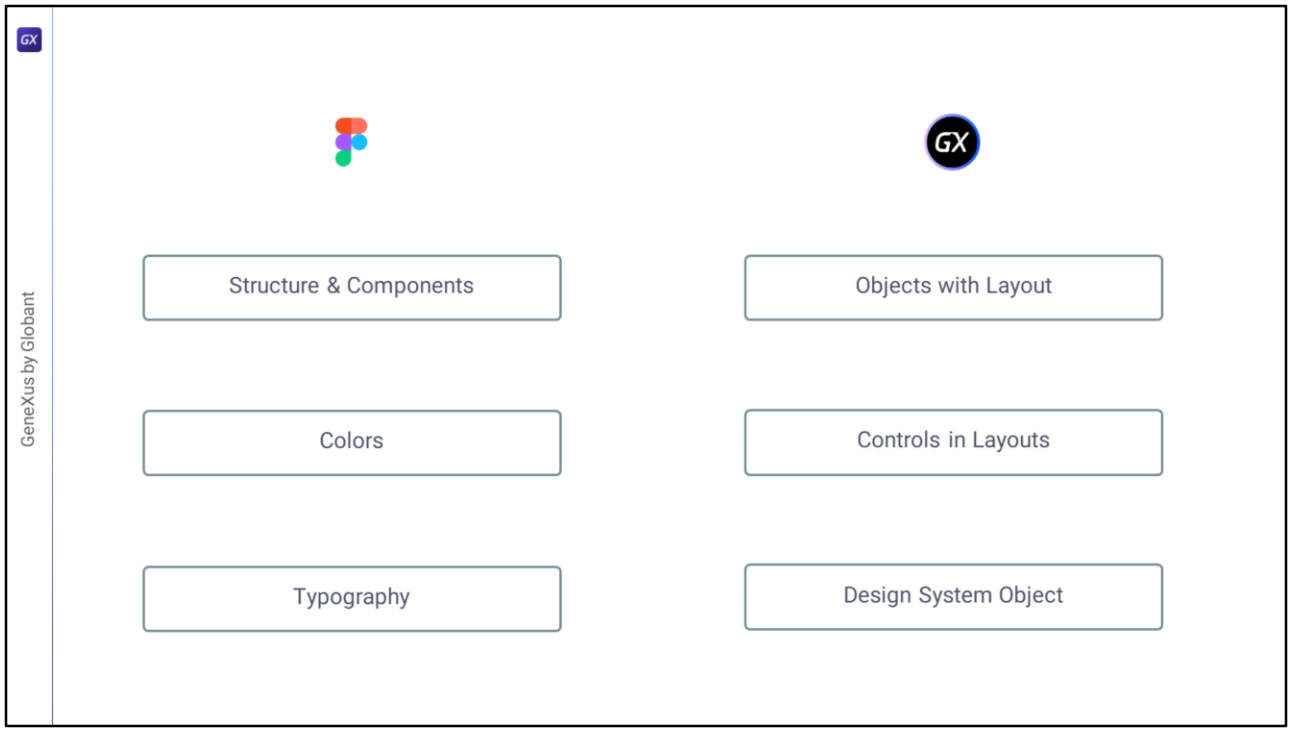
In the Tokens tab, which are the constants, I defined the color constants, those we saw just now, and that Chechu had identified and named in this way. Then these color constants are going to be used...



...inside the styles tab, inside the classes. For example, for the h2 class I'm defining the color property that's using the token shown there. And that's how we start building everything that has to do with the application's style, through the definition of the classes.



OK, here we see the scheme of what we've seen so far, dividing the whole application into objects: Master Panel, Panels, Stencils and the Design System determining the design and appearance of all the objects...



This is the end of a first review of all these players that will be involved in GeneXus to reflect all the decisions made by Chechu in her Design System. So all that's left now is to get down to work.


GeneXus by Globant
GeneXus™
by Globant
training.genexus.com