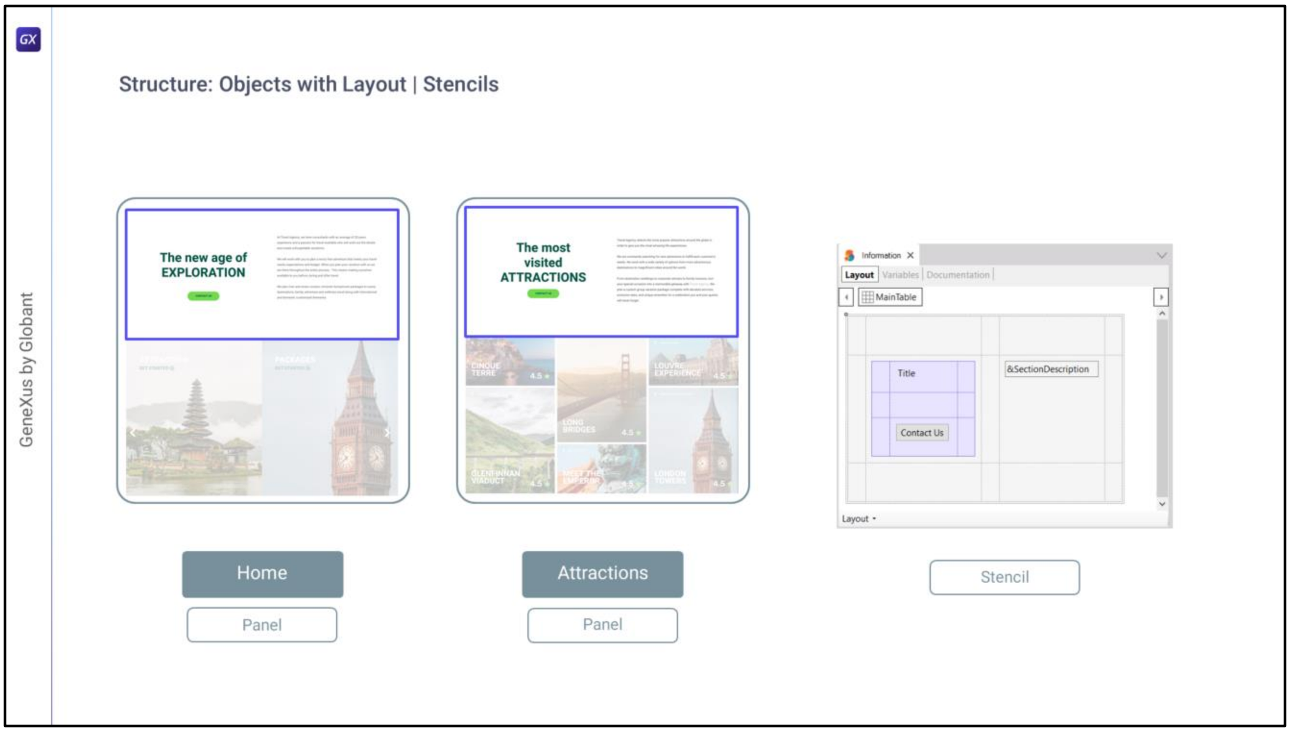
GeneXus by Globant

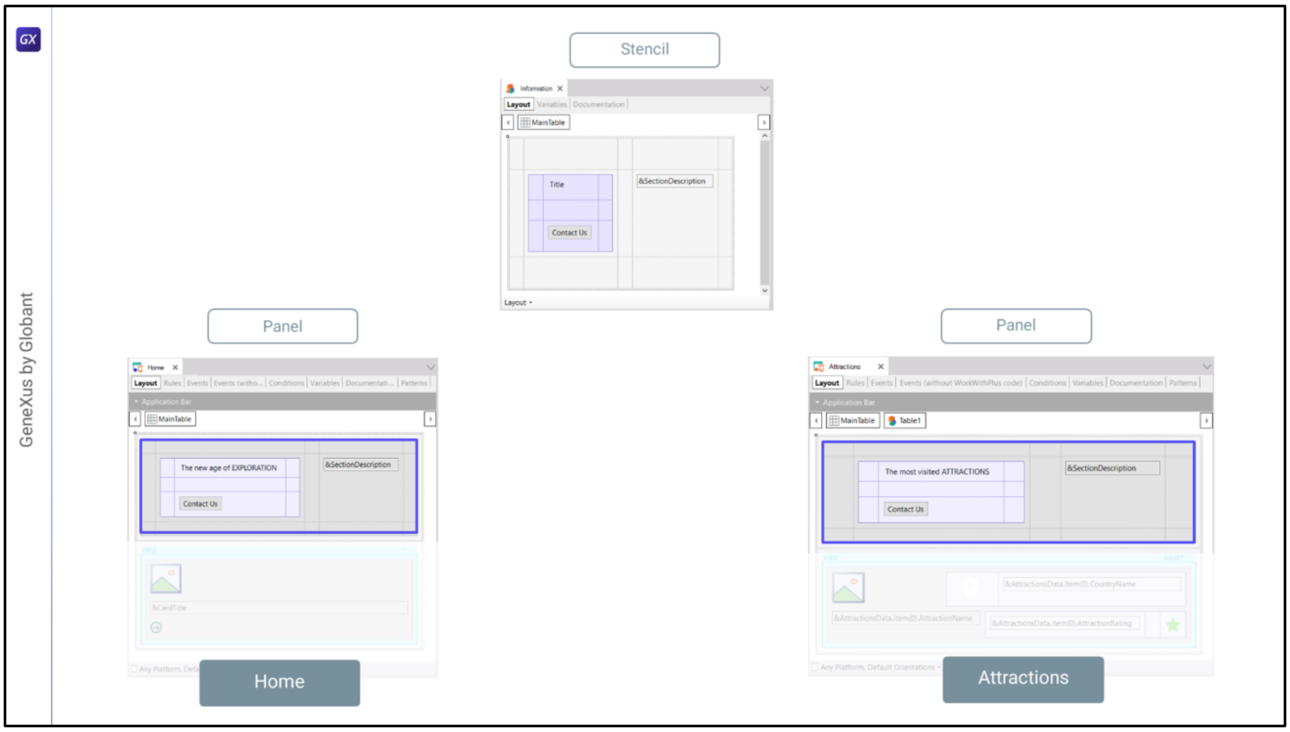
First Layout in GeneXus. Structure

Cecilia Fernandez

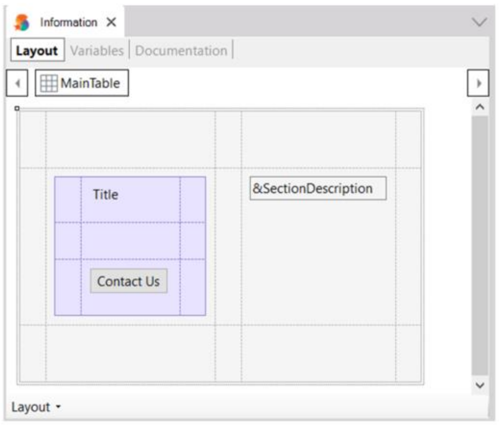


Structure: Objects with Layout | Stencils

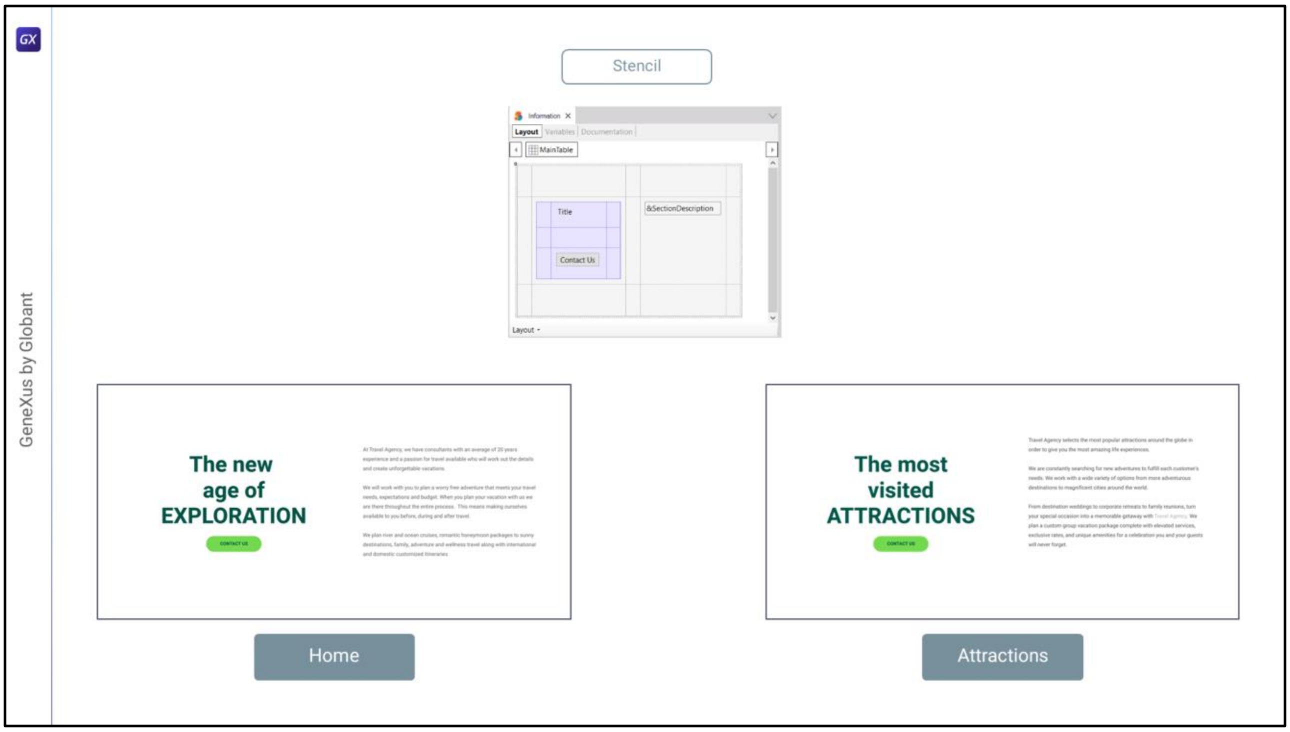
In the previous module we had seen that these two parts of the Home and Attractions screens had the same structure so we could choose to model them inside a Stencil object, precisely to reuse them, and then....



...insert it as a control in each of these panels.

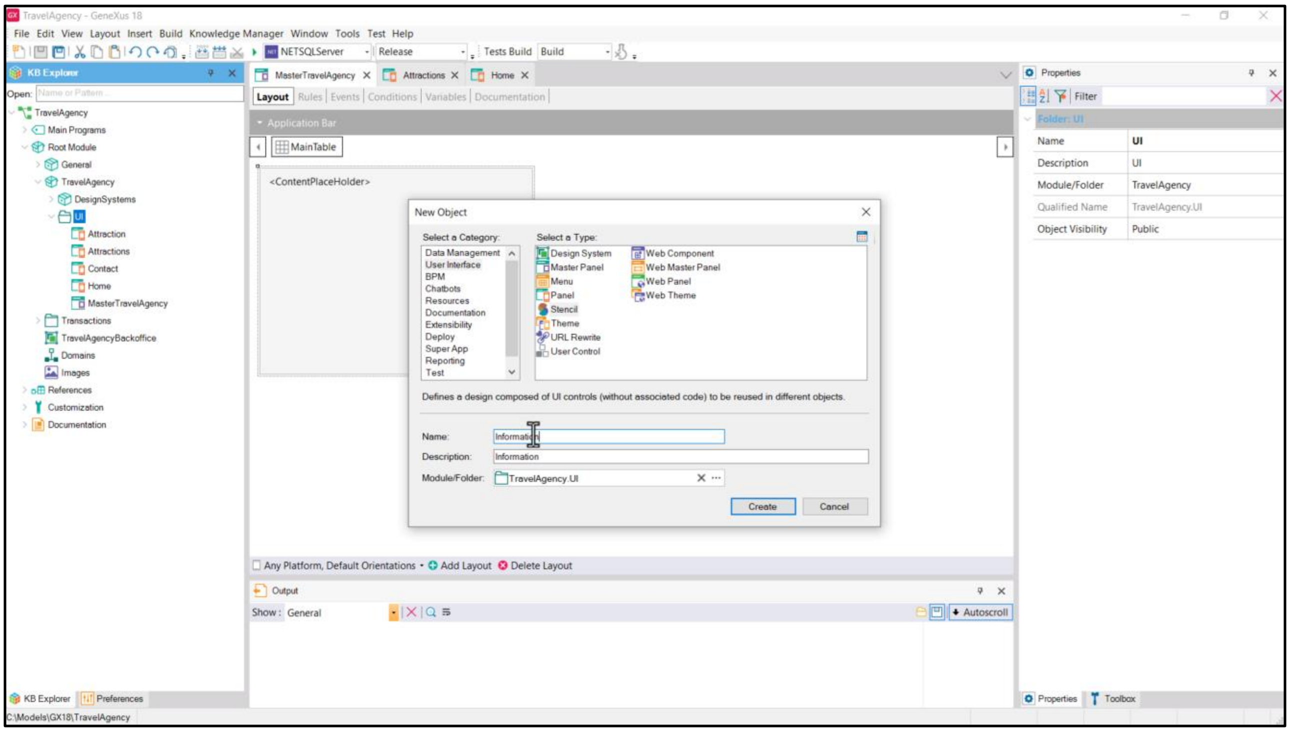


In this class, we are going to analyze which control hierarchy should be used to implement the structure of this stencil...



In the next classes we will see how to make that structure look as defined by our designer in the Figma file.

So let's start by creating the Stencil in GeneXus.

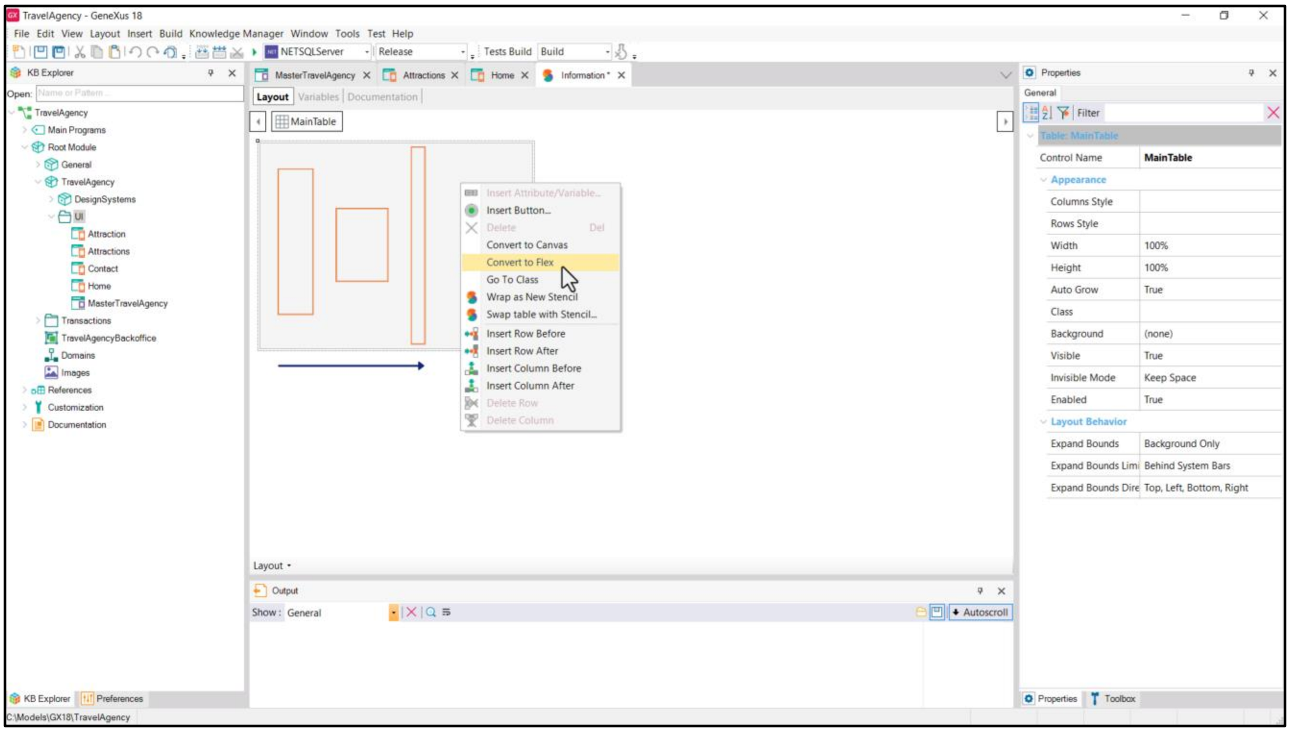


Defines a design composed of UI controls (without associated code) to be reused in different objects.

Name: Information
Description: Information
Module/Folder: TravelAgency.UI

Name	UI
Description	UI
Module/Folder	TravelAgency
Qualified Name	TravelAgency.UI
Object Visibility	Public

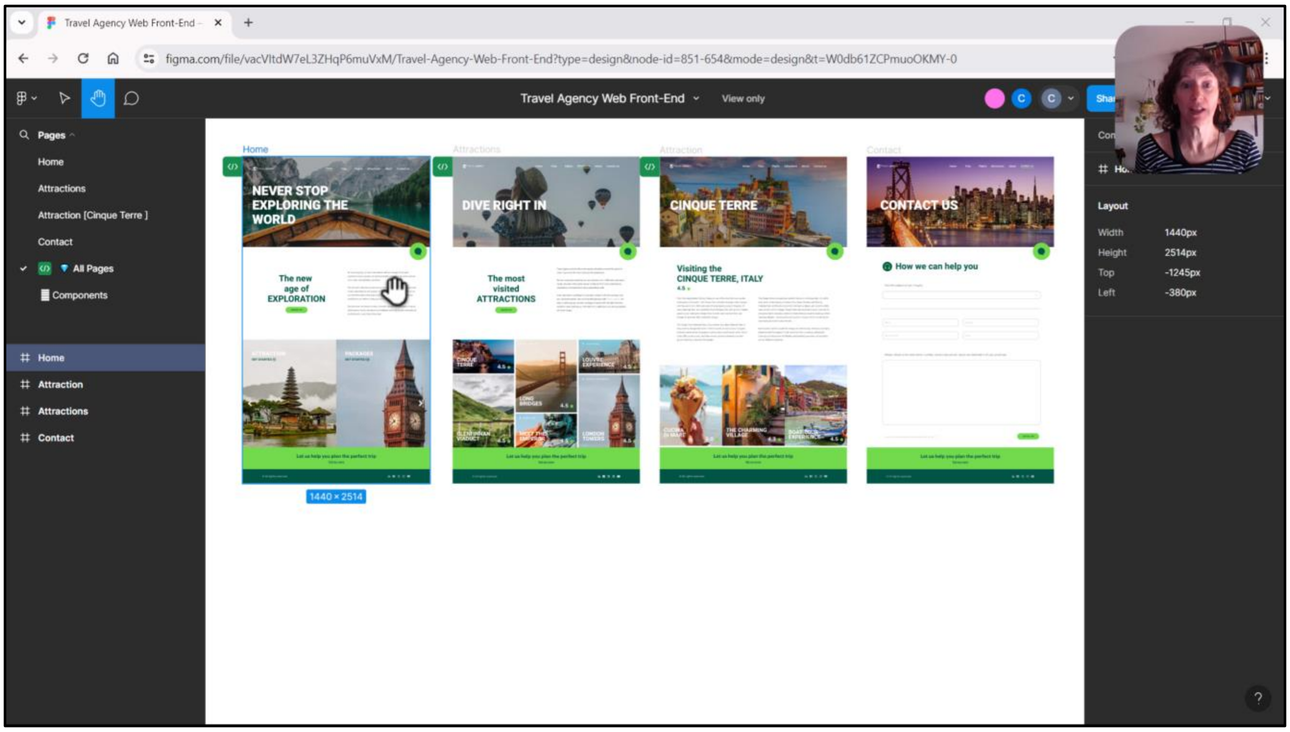
To do this, from this folder, which is where we want to create the Stencil object, we choose the stencil object type and name it Information.



Control Name	MainTable
Appearance	
Columns Style	
Rows Style	
Width	100%
Height	100%
Auto Grow	True
Class	
Background	(none)
Visible	True
Invisible Mode	Keep Space
Enabled	True
Layout Behavior	
Expand Bounds	Background Only
Expand Bounds Limi	Behind System Bars
Expand Bounds Dire	Top, Left, Bottom, Right

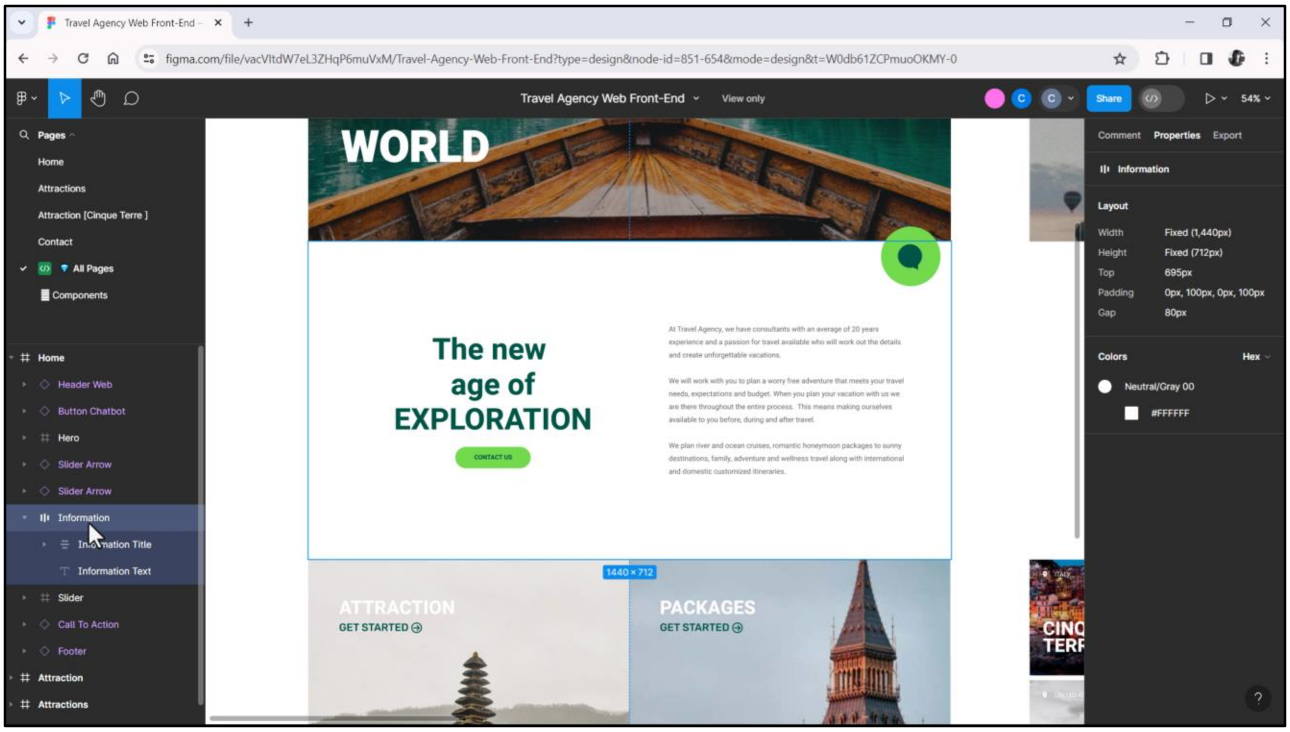
The layout is initialized with a table called Main Table. As long as we don't place any controls inside, it will remain empty.

If we right-click, it offers us to convert this table into two types of containers that have specific uses: Canvas, which we will use later to implement the Header of the Master Panel, and that will allow us to superimpose controls; and Flex that will allow us to place the elements of the table in one direction, either horizontally –as a row– or vertically –as a column.


Travel Agency Web Front-End
figma.com/file/vacVltdW7eL3ZHqP6muVxM/Travel-Agency-Web-Front-End?type=design&node-id=851-654&mode=design&t=W0db61ZCPmuoOKMY-0
Travel Agency Web Front-End
View only
Pages
Home
Attractions
Attraction [Cinque Terre]
Contact
All Pages
Components
Home
Attraction
Attractions
Contact
NEVER STOP EXPLORING THE WORLD
The new age of EXPLORATION
DIVE RIGHT IN
The most visited ATTRACTIONS
CINQUE TERRE
Visiting the CINQUE TERRE, ITALY
CONTACT US
How we can help you
1440 × 2514
Layout
Width 1440px
Height 2514px
Top -1245px
Left -380px

Let's analyze this layout, which is the one we want to implement, in Figma.

First, note that here we are positioned in the Home frame, which is the one that designs the Home page. It has a width of 1440 pixels and a height of 2514 pixels. And they will all be more or less the same. What does this mean? That if we run our GeneXus objects, our panels, in a browser with these dimensions, the screens should look exactly like this.
But, as Chechu told us, this design was devised so that the width can be extended or reduced a bit and the design adapts to the new width. As she said, this design corresponds to a certain width and small variations of that width, because if we shrink it enough, we will have to reassemble these screens, design them again to arrange the information in a different way, according to that screen width. This is what is known as breakpoints, where the design must be modified. It will also be used for Tablet, Phone and small variations of those sizes.



At Travel Agency, we have consultants with an average of 20 years experience and a passion for travel available who will work out the details and create unforgettable vacations.

We will work with you to plan a worry free adventure that meets your travel needs, expectations and budget. When you plan your vacation with us we are there throughout the entire process. This means making ourselves available to you before, during and after travel.

We plan river and ocean cruises, romantic honeymoon packages to sunny destinations, family, adventure and wellness travel along with international and domestic customized itineraries.

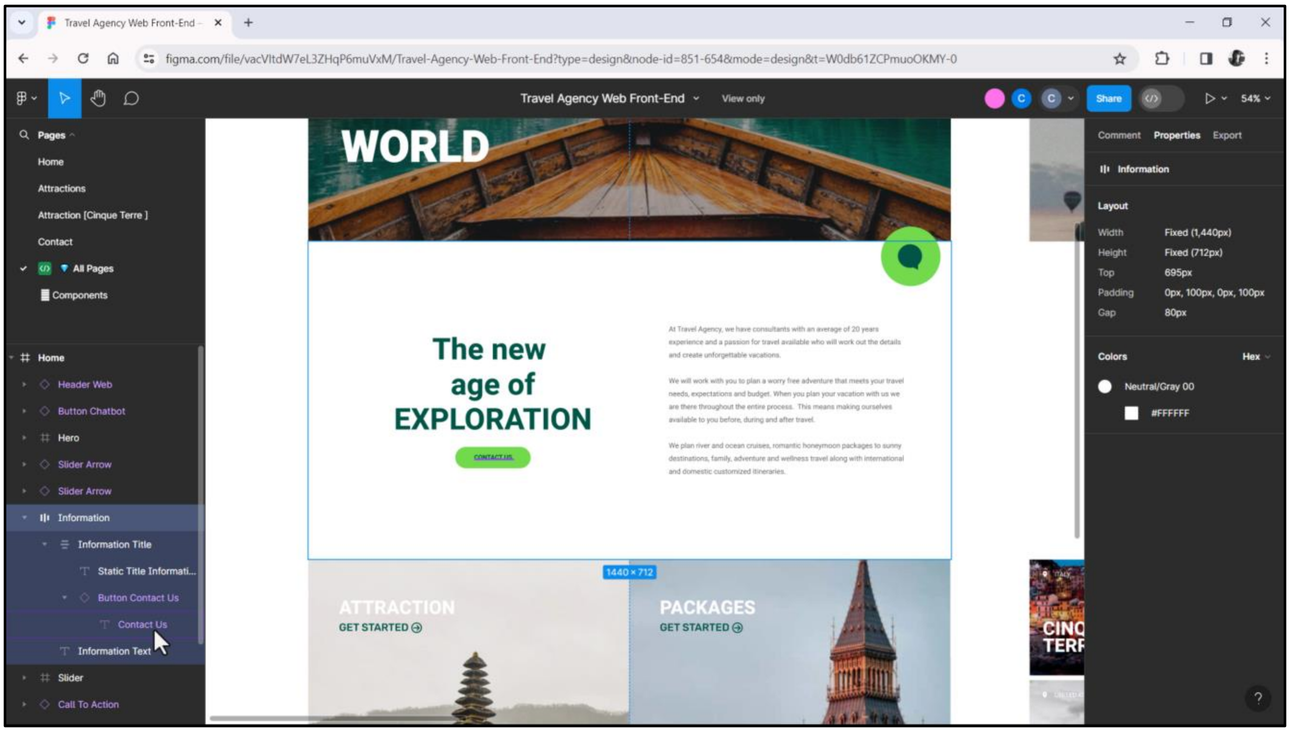
Layout

Width	Fixed (1,440px)
Height	Fixed (712px)
Top	695px
Padding	0px, 100px, 0px, 100px
Gap	80px

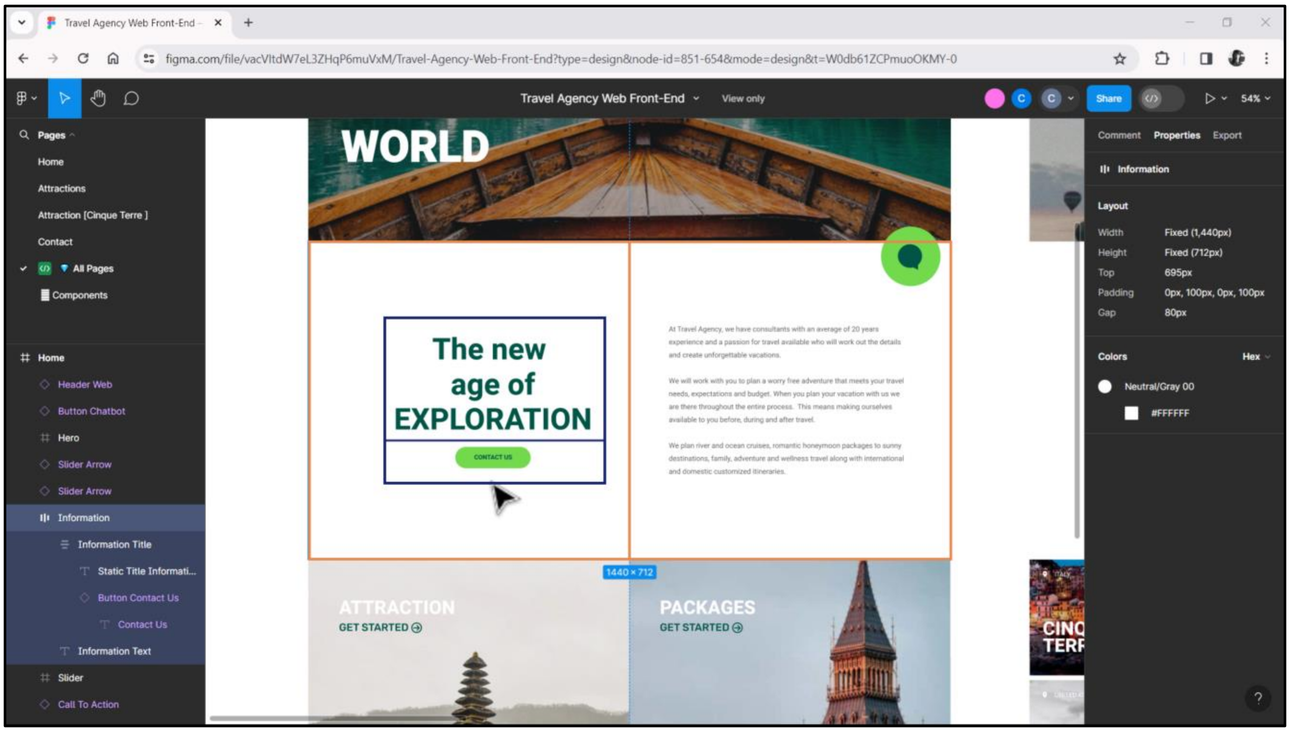
Colors — Hex

●	Neutral/Gray 00
⬜	#FFFFFF

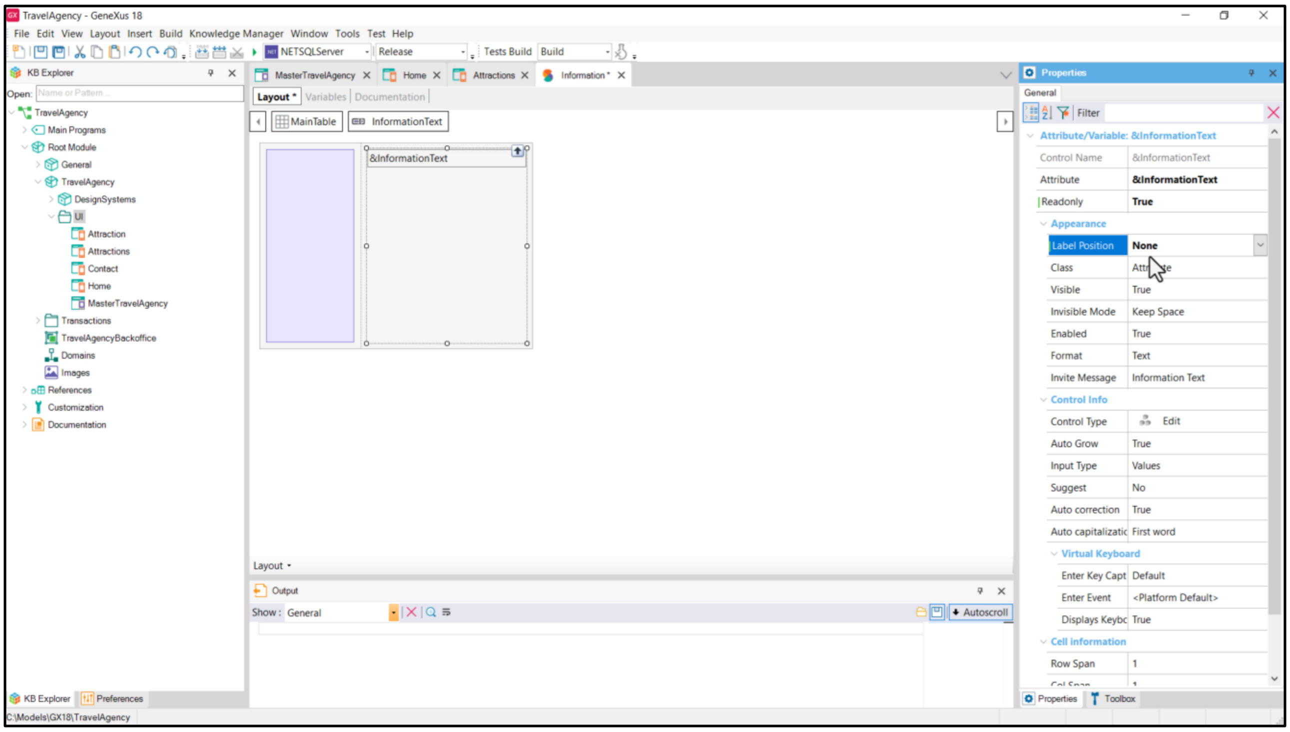
Now let's analyze the layout we are interested in, the one we are trying to implement with the Stencil object. It corresponds to this container here, the one named Information, which contains two elements: another container, the one named Information Title that we see on the left side of the screen, and a text to the right.



The container on the left, in turn, contains another text, this time on top, and a button below, which as we can see, is an instance of a component; this component will have two layers, one for the background and one for the text of the button, and we will analyze it later.



So, structurally we could organize this screen like this: a table (our main table) with a single row and two columns, where in the first one we have another table, and in the second one the text control, which in GeneXus could be either a TextBlock or a text variable of type edit, readonly. And the internal table would have a single column and two rows, where in the first one we would place the other text control (TextBlock or variable, we will have to choose) and in the second one the button.



Attribute/Variable: &InformationText

Control Name	&InformationText
Attribute	&InformationText
Readonly	True
Appearance	
Label Position	None
Class	Attr[illegible]e
Visible	True
Invisible Mode	Keep Space
Enabled	True
Format	Text
Invite Message	Information Text
Control Info	
Control Type	Edit
Auto Grow	True
Input Type	Values
Suggest	No
Auto correction	True
Auto capitalizatic	First word
Virtual Keyboard	
Enter Key Capt	Default
Enter Event	<Platform Default>
Displays Keybc	True
Cell information	
Row Span	1

If we do it in GeneXus...

We already have the table; it is the Main table. So, let's insert the first element, which is the other table. Let's rename it to match the name that our designer gave it, it will be easier to find it. Okay, now let's insert the element on the right, which was the text.
We can choose to implement it with a variable type control or with a textblock.

I'm going to go with the variable. I'm going to drag it and create it with the same name that our designer gave it, Information Text. I'm going to set as data type any of those that represent characters: it could be Character, VarChar or LongVarChar, the one I'm going to choose. As we will see, and I will explain in a little while, choosing this data type will cause the Auto Grow property to be automatically set to True.

One of the drawbacks of variables is that they are write-enabled by default. So I'm going to change the Readonly property so that it is read-only.
The other drawback is that they have a label by default, so I'm going to set the position of the label to None, that it is nowhere, hidden.

As I was saying, I could have chosen, instead of the variable, to represent this with a Text Block. I choose the variable because it is possible that this information, the content, comes from the Backoffice, that is to say, it is extracted from the database, and it is easier to assign the content to a variable, simply with the direct assignment by means of the equal sign, than to do it for the Text Block, for which I will have to use the Caption property. Textblock, period, caption, equal... and then assign a value. This is precisely one of the border cases, isn't it? This text is not going to vary, therefore it could well be a text block. It is clearly its use... we are inclined to use a variable when the content changes.

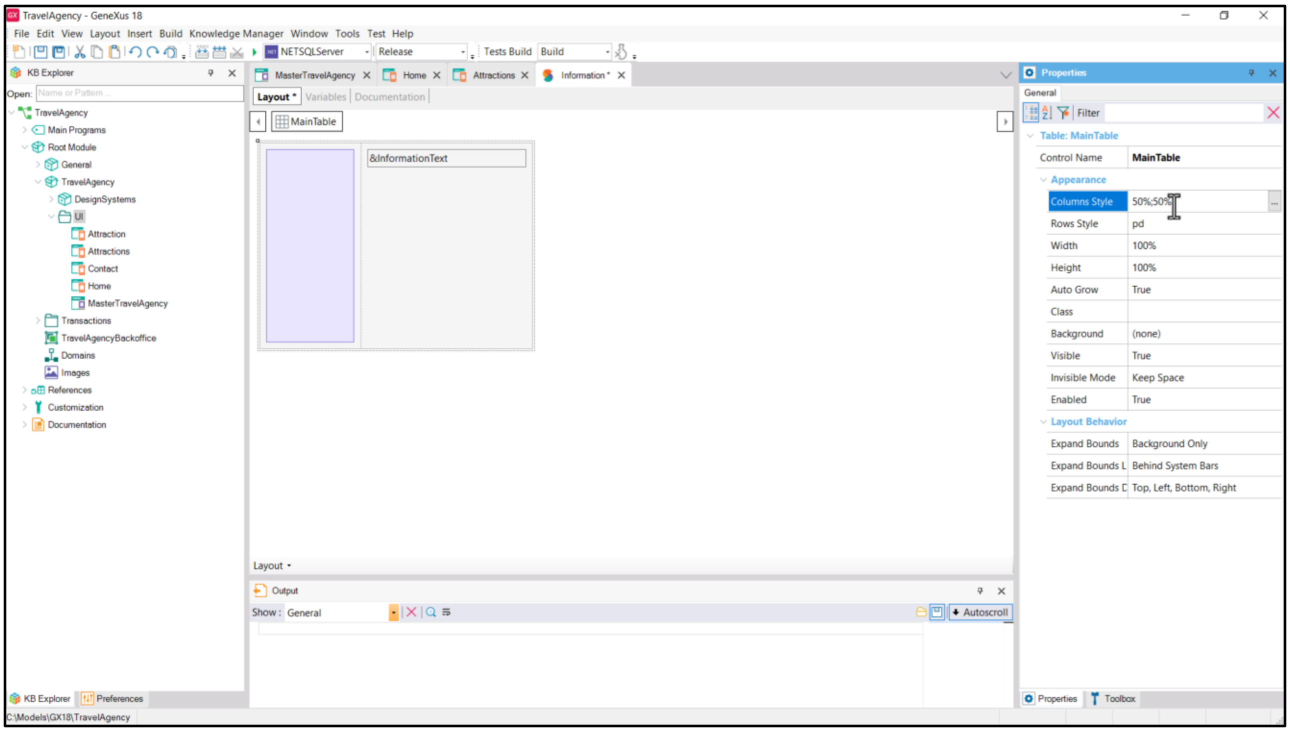
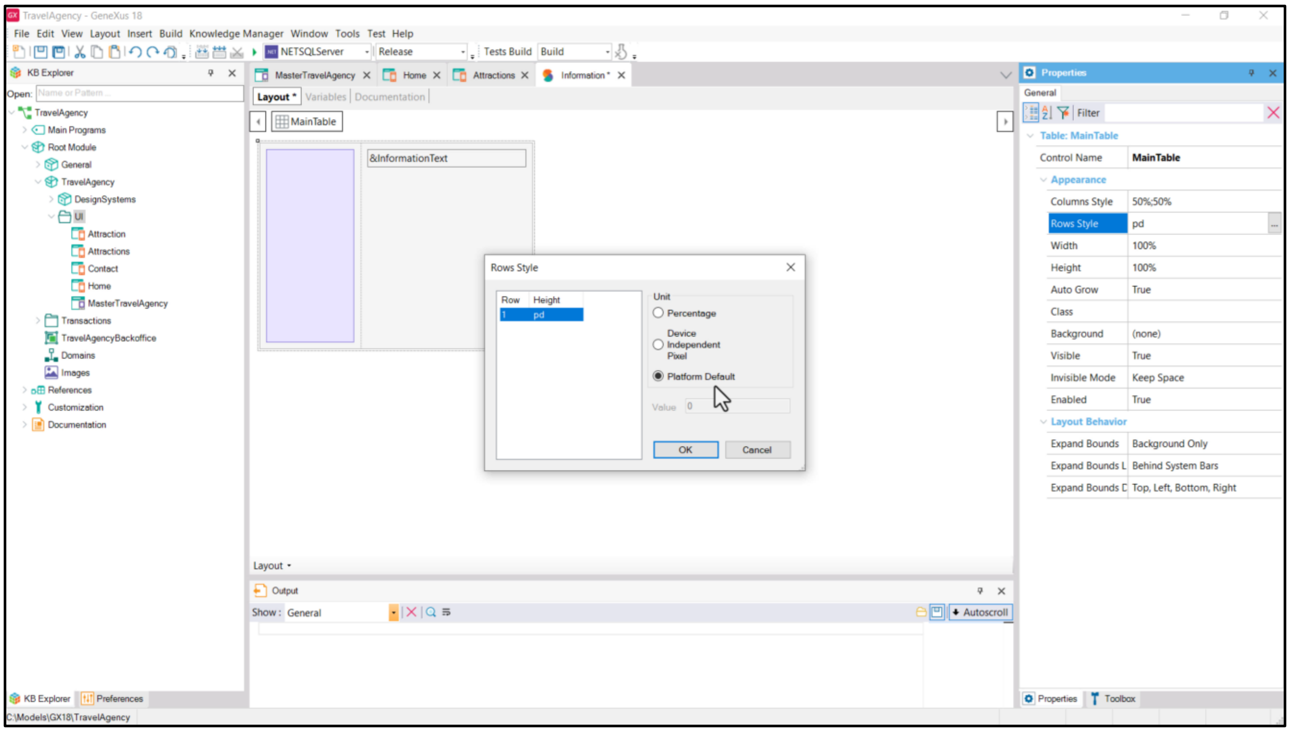


Table: MainTable

Control Name	MainTable
Appearance	
Columns Style	50%;50%
Rows Style	pd
Width	100%
Height	100%
Auto Grow	True
Class	
Background	(none)
Visible	True
Invisible Mode	Keep Space
Enabled	True
Layout Behavior	
Expand Bounds	Background Only
Expand Bounds L	Behind System Bars
Expand Bounds D	Top, Left, Bottom, Right

Now, before adding the controls for this table, let's look at the properties of the main table, in particular these four.
The first one specifies the width of each of the two columns. And the second one specifies the height of the only row that the table has at the moment.
What do these values of 50% mean for each column? That its width will be 50% of the total width of the table. And what is the total width of the table? The one given by the Width property, which in this case is also a percentage, indicating that the width of the table will correspond to 100% of the width of its container.



Rows Style

Row	Height
1	pd

Unit: ( ) Percentage ( ) Device Independent Pixel [✓ Platform Default]

Value: 0

OK Cancel

Table: MainTable

Control Name	MainTable
Appearance	
Columns Style	50%;50%
Rows Style	pd
Width	100%
Height	100%
Auto Grow	True
Class	
Background	(none)
Visible	True
Invisible Mode	Keep Space
Enabled	True
Layout Behavior	
Expand Bounds	Background Only
Expand Bounds L	Behind System Bars
Expand Bounds D	Top, Left, Bottom, Right

And what does the pd value mean in the property that determines the height of the rows? Here it indicates it clearly, it is an abbreviation of Platform Default. This default depends on each platform, but it is a fixed value. We could specify a percentage, in which case it is relative to the height of the table —here we also see that it is 100%— or we could specify a value in dips, which is the same as pixels.

The dip is an abstraction of the pixel unit that was first used in GeneXus when we started implementing native applications. The equivalence is one to one.

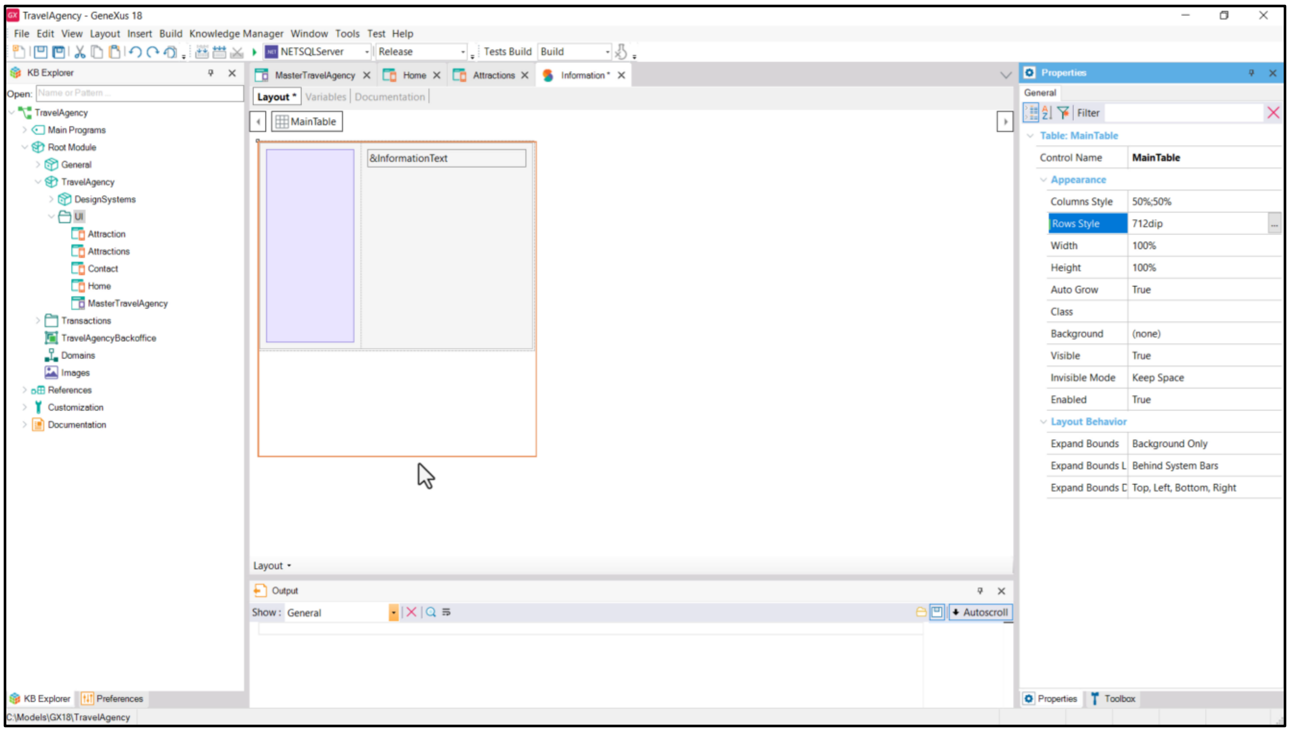


Table: MainTable

Control Name	MainTable
Appearance	
Columns Style	50%;50%
Rows Style	712dip
Width	100%
Height	100%
Auto Grow	True
Class	
Background	(none)
Visible	True
Invisible Mode	Keep Space
Enabled	True
Layout Behavior	
Expand Bounds	Background Only
Expand Bounds L	Behind System Bars
Expand Bounds D	Top, Left, Bottom, Right

Let's place here the 712 dips or pixels that we obtained by inspecting the frame in Figma. Does it seem odd that the height of the row is this size, but the height of the table (which only has that row) is 100%? Here's a word of caution: although the editor is not distinguishing the table border from the row border, they are not the same thing. The table border will extend to 100%, and the row border will correspond to these 712 dips.

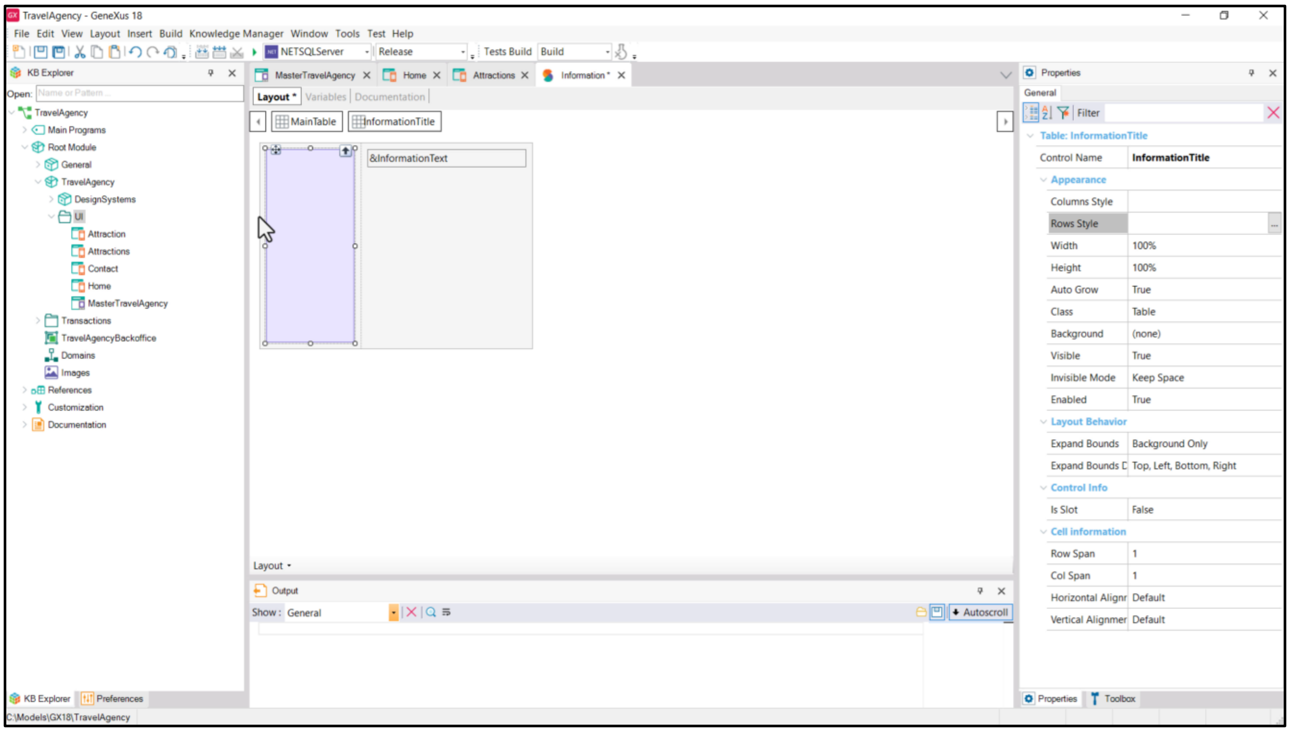
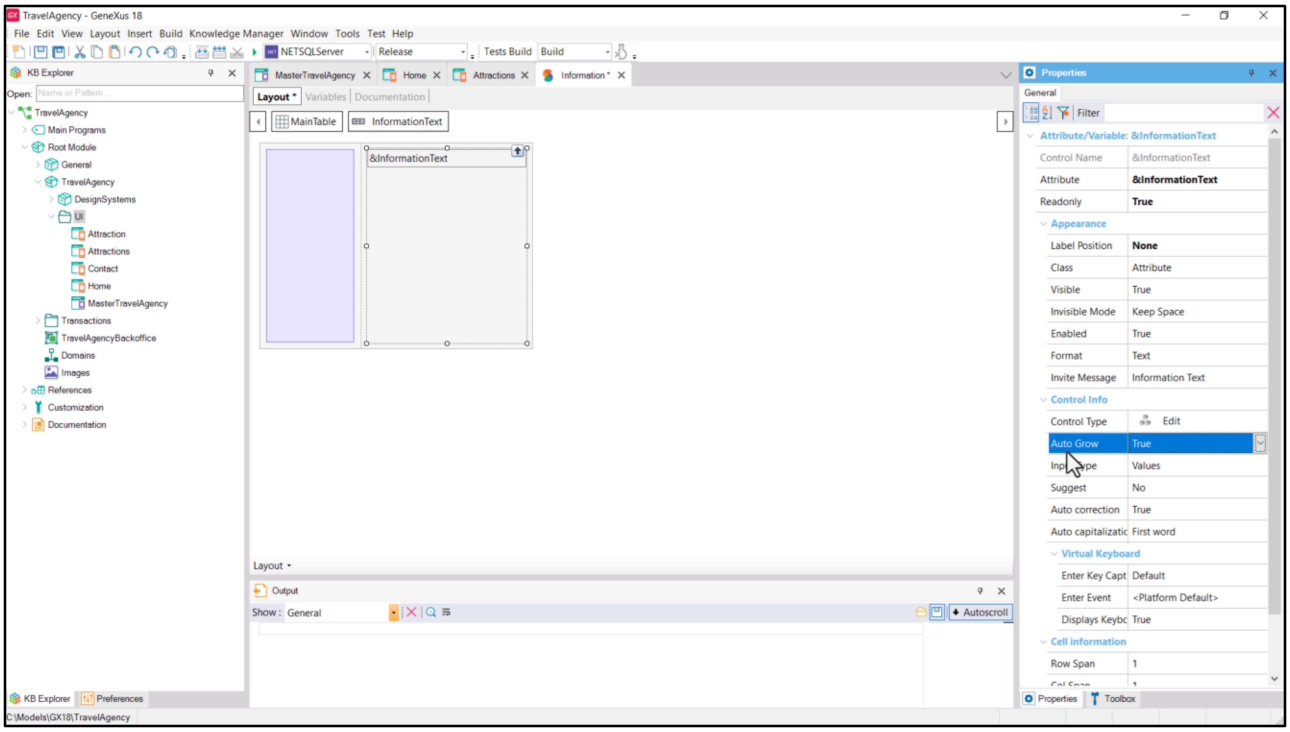


Table: InformationTitle

Control Name	InformationTitle
Appearance	
Columns Style	
Rows Style	
Width	100%
Height	100%
Auto Grow	True
Class	Table
Background	(none)
Visible	True
Invisible Mode	Keep Space
Enabled	True
Layout Behavior	
Expand Bounds	Background Only
Expand Bounds D	Top, Left, Bottom, Right
Control Info	
Is Slot	False
Cell information	
Row Span	1
Col Span	1
Horizontal Alignr	Default
Vertical Alignmer	Default

Controls, by default, take up the entire space of the cell in which they are placed. Thus, if we look at the properties of this table, which is empty for now, its width and height are set to 100%, i.e., they will take up the entire cell in which they are placed.



Attribute/Variable: &InformationText

Control Name	&InformationText
Attribute	&InformationText
Readonly	True
Appearance	
Label Position	None
Class	Attribute
Visible	True
Invisible Mode	Keep Space
Enabled	True
Format	Text
Invite Message	Information Text
Control Info	
Control Type	Edit
Auto Grow	True
Inp ype	Values
Suggest	No
Auto correction	True
Auto capitalizatic	First word
Virtual Keyboard	
Enter Key Capt	Default
Enter Event	<Platform Default>
Displays Keybc	True
Cell information	
Row Span	1

The same will happen with the variable, and we need to know this because there is no property that evidences it.
What will happen if the content of the variable is so large that it doesn't fit in the 712 pixel height of the cell? Since the Auto Grow property is set to True, as is the same property at the table level, then what will happen is that the cell will expand downwards so as not to leave content outside. If the property were set to false, then the content would be truncated at 712 pixels or dips.

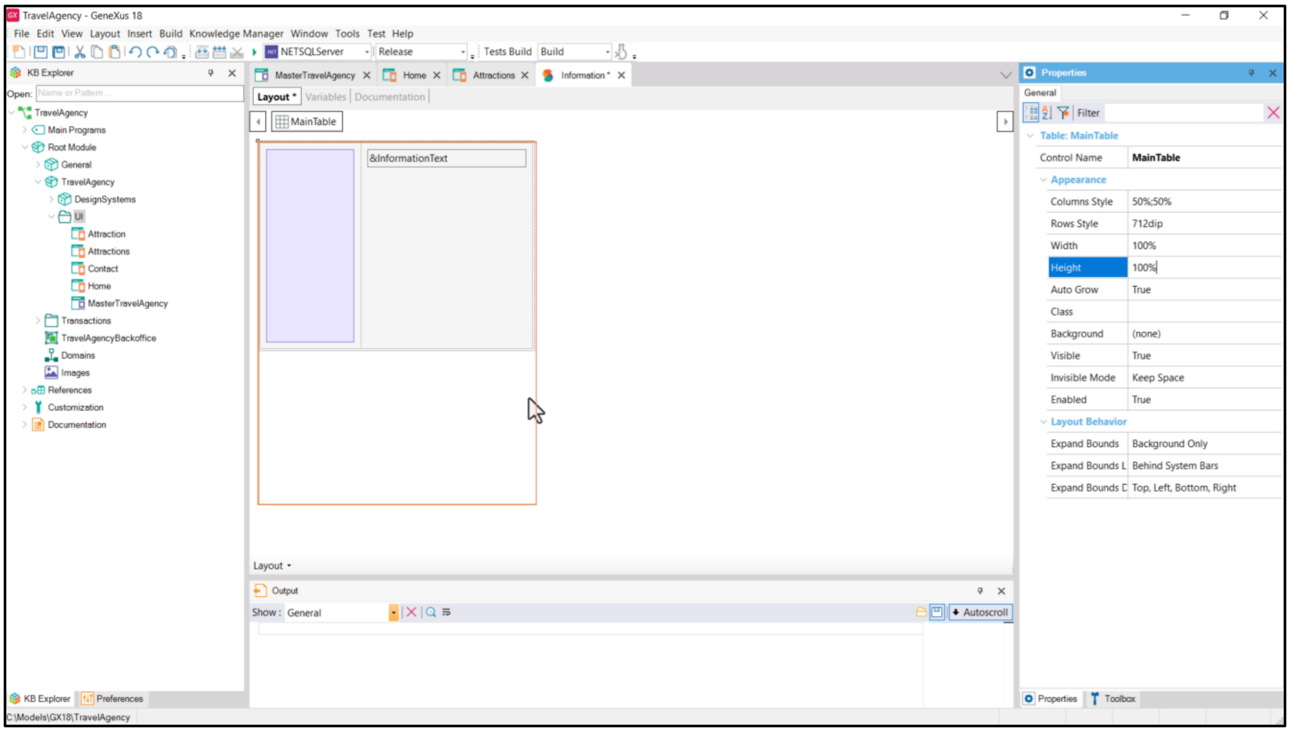


Table: MainTable

Control Name	MainTable
Appearance	
Columns Style	50%;50%
Rows Style	712dip
Width	100%
Height	100%
Auto Grow	True
Class	
Background	(none)
Visible	True
Invisible Mode	Keep Space
Enabled	True
Layout Behavior	
Expand Bounds	Background Only
Expand Bounds L	Behind System Bars
Expand Bounds D	Top, Left, Bottom, Right

If we leave it like this, although the row will be 712 pixels or dips high, as the height of the table was set to 100%, if the container of this stencil has a height greater than 712 pixels, then there could be an empty space.

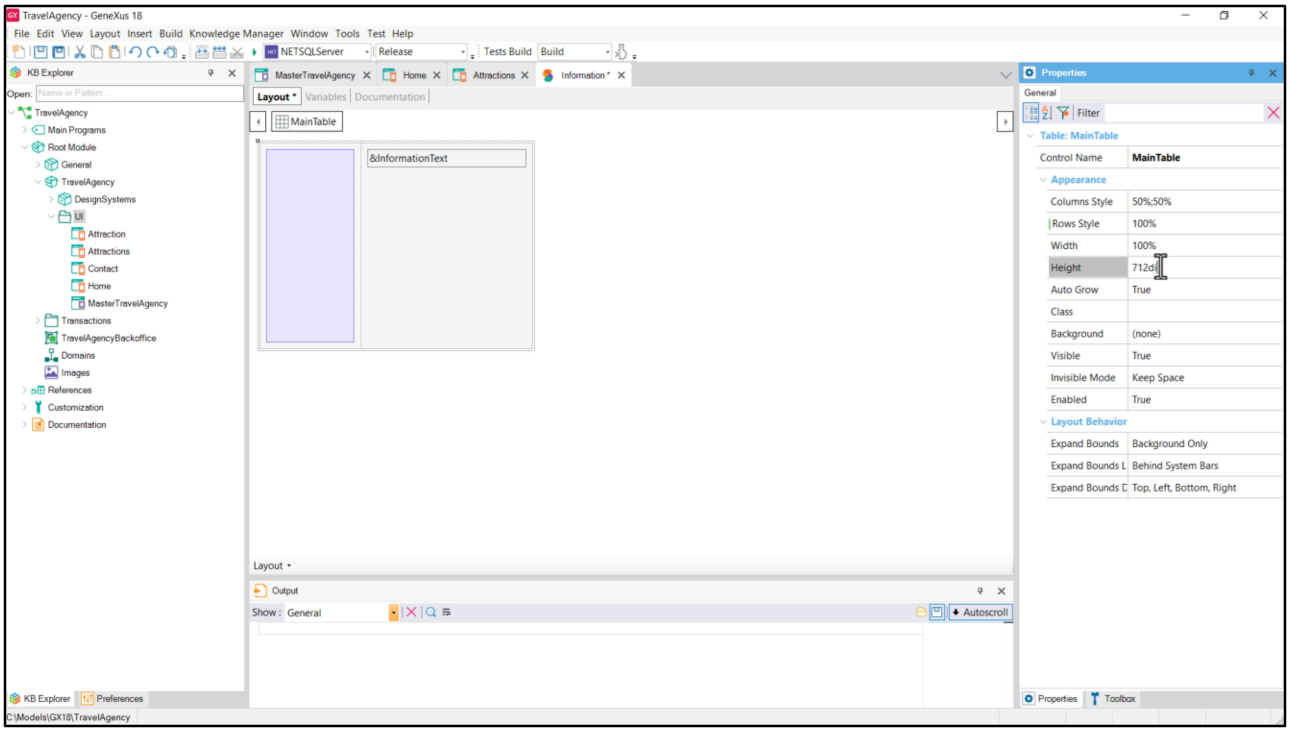


Table: MainTable

Control Name	MainTable
Appearance	
Columns Style	50%;50%
Rows Style	100%
Width	100%
Height	712dip
Auto Grow	True
Class	
Background	(none)
Visible	True
Invisible Mode	Keep Space
Enabled	True
Layout Behavior	
Expand Bounds	Background Only
Expand Bounds L	Behind System Bars
Expand Bounds D	Top, Left, Bottom, Right

The other alternative would be to set the row height to 100%, i.e., to take up 100% of the table height, and define in the table height those 712 dips or pixels.

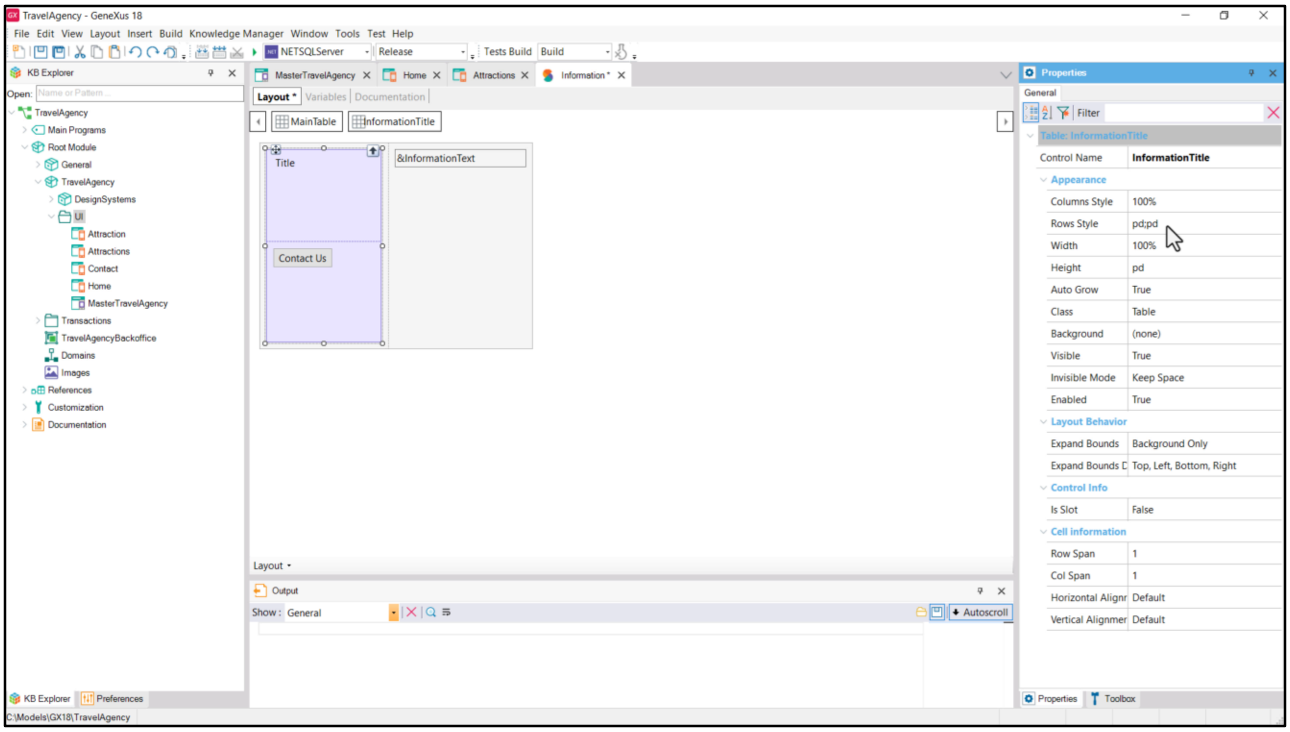


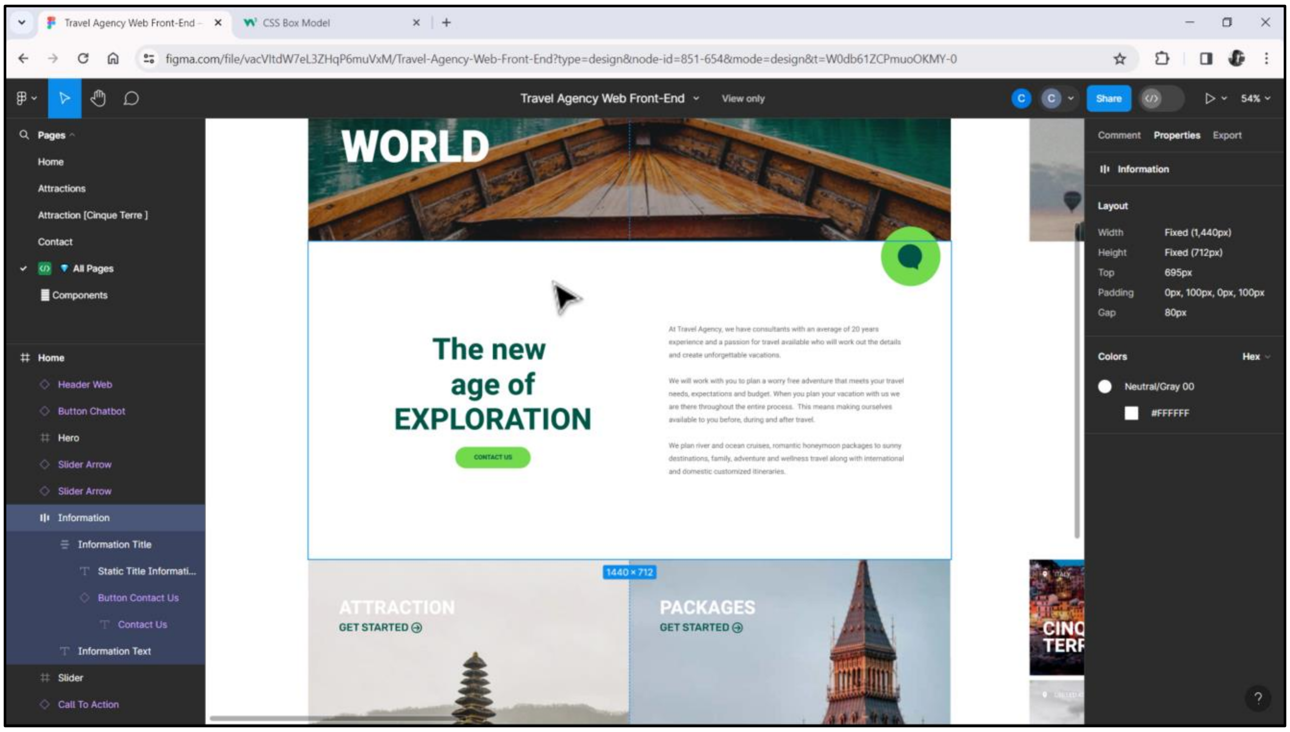
Table: InformationTitle

Control Name	InformationTitle
Appearance	
Columns Style	100%
Rows Style	pd;pd
Width	100%
Height	pd
Auto Grow	True
Class	Table
Background	(none)
Visible	True
Invisible Mode	Keep Space
Enabled	True
Layout Behavior	
Expand Bounds	Background Only
Expand Bounds D	Top, Left, Bottom, Right
Control Info	
Is Slot	False
Cell information	
Row Span	1
Col Span	1
Horizontal Alignr	Default
Vertical Alignmer	Default

Now let's insert the text as TextBlock in this table (we know that it is a short text, which probably will not change), let's call it the same as Chechu did in Figma. And to the Caption for now I'm going to leave Title, because later on it will be updated in the Home and Attractions panels with the value that it will have to take in each one.

And in another row we insert the button.

If we look at the table properties, we see that the column is now a single column, 100% of the width, and each row will take the default platform height. Clearly we will have to change this, so let's go to Figma to see what values we need for the widths and heights.



But before analyzing these sizes, let's analyze the case of the external container.

What I want to focus on now is the spacing between the elements and the border of the container. We will continue in the next video.

GeneXus by Globant

training.genexus.com