

In any software application, the correct use of the available screen controls is essential in order to provide the best user experience.

This video will show the most frequently used controls, as well as how we can organize them in the panel layout and customize their appearance.

Panel: ViewCustomers	Source* Rules Variables Help Documentation	Nama	Name
Varre Customeriameria Customeriameria Customeriameria Customeriameria Customeria Customeria Customeria Customeria Customeria Customeria Customeria Customeria Customeria Customeria Customeria Customeria Customeria Customeria Customeria Customeria Customeria Customeria Customeria	<pre>Sure* hus: Vursites; Hebp Documentation;</pre>	Name	MamMenu X Menu Kerns Variables Documentation Action (ViewAttractions)

We are going to create a panel to view the travel agency's customers data.

First of all, we import into our KB the Customers.xpz file that we found in the Materials section of the course page.

The Customer transaction was imported with attributes for the main data such as the identifier, name, address, phone, email, and photo. We can also see that the data provider Customer\_DataProvider was imported, including customer data and other objects.

To view customer data, we created the ViewCustomers panel, dragged a grid control with the attributes CustomerId, CustomerName, CustomerEmail, CustomerPhoto, and CountryName... and saved.

	Panel: ViewCustomers						
	🖾 Start Page 🗙 🗮 MainMenu X 🗸 🗸	Properties					
	Menu Events Variables Documentation	🗄 🛃 🌾 Filter					
	V 🔳 Menu	Action: Action (ViewCu	stomers)				
	✓ Items ↓ Action (ViewCountries)	Name	ViewCustomers	Start Page X 🔽 ViewCustomers X 🙍 Navigation View X		× ×	Toolbox
	Action (ViewCustomers)	Description	View Customers	Layout Rules Events Conditions Variables Document			Controls     ED Attribute/Variable
		Image	Customers		00 <u>8</u> 🖓		Button
		Unselected Image	(none)	← MainTable ←	View Customers		A Text Block
Ħ		Class	Menultem		57 Joh 💟 🤰		Containers
Globant				CustomerId CustomerName CustomerEmail	58 Ann 🗹 🧃		Canvas
0				S	59 Pete 🗹 👔		Component
o o					60 Chia 🔛 🚺		Grid
lsh					61 Man 🔽 🍶	Q	🕅 Group Stencil
eXı					62 Rogi		Tab
GeneXus by					63 Mei 🌄		Table Tabular Grid
0		Name					Miscellaneous
							Ads View
	Photo	Email					Animation View
							Query Viewer
		Country		🗌 Any Platform, Default Orientations - 🗘 Add Layout 🔇 De			Properties Toolbox
				E Output			
				Show: Build   Android Execution started			
	Addr	ress		Command: adb -s emulator-5554 install -r "C:\KBs Performing Streamed Install		\Android\MainMenu\buil	id\outputs\apk\debug\
				Performing Streamed Install Success Command: adb -s emulator-5554 shell am start -n			
				Starting: Intent { act=android.intent.action.MAI		<pre>xus.android.core.activ .genexus.mobilecoursev</pre>	
				Success: Android Execution Success: Run MainMenu			

Now we open the MainMenu object and drag the ViewCustomers panel to the Items node. In the Image property of the created action, we assign the Customers image.

Note: in order to see the Menu as it is shown here, create the Panel ViewAttractions, drag it to the Items node of the MainMenu object and assign in the Image property the image named 'Attractions'''. We will work on this Panel in a future video.

We recommend to always run the application with the emulator previously opened, to optimize prototyping times. In this way, when GeneXus tries to instantiate an emulator to run the mobile application, it will find this emulator running and use it. In this way, we will save the instantiation time of the virtual device and the execution will be much faster. Now we press F5.

The list of clients appears without any layout, and there are even some data that we don't get to see due to lack of space.

We will try to get to the following card layout, which seems to be the most appropriate. Each row of the customers grid will be displayed with this layout, with one customer per row.

Note that the ID is not included in this layout and that we want to show the customer's address which was not included in the original grid. Let's add the address by right-clicking on the grid and choosing "Add attribute or variable" and Customer Address.

nt	Panel: ViewCustomers          Photo       Image: Second			
GeneXus by Globant	ViewCustomers* X	$\sim$	O Properties	<b>9</b>
S G	Layout * Rules   Events   Conditions   Variables   Documentation		General	
us ł	<ul> <li>Application Bar</li> </ul>		Attribute/Variable: Custor	nerAddress
Xər	ImainTable     ImainTable     ImainTable     ImainTable	$\mathbf{F}$	Attribute	CustomerAddress
Ger	GRID		Label Caption	Customer Address
	O Customerid CustomerName CustomerEmail		Readonly	True
			~ Appearance	
			Label Position	Platform Default ~
			Class	Platform Default
	Customer Address		Visible	None
			Invisible Mode	Left
			Enabled	Тор
	oo		Format	Right
			Invite Message	Bottom
		_	✓ Link	Float

Let's start by moving the photo attribute to the left side of the grid and dragging the address field below the photo.

Note that CustomerAddress has a label on the left. If we go to the attribute properties, below the Appearance section, we see that the Label Position property has the Platform Default value. For Android it is the Top value, above the content, and for iOS it is to the left.

If we open the combo we see a series of values that change the default place where the label associated with the attribute will be displayed.

The value None means that the label will not be displayed. The values Left, Top, Right, and Bottom determine that the label will be displayed to the left, top, right, or bottom of the content field.

With the Float value, the label will be displayed on the position of the content attribute or variable, and when the user starts typing in the field, the label will move up the field as if floating. This only works for editable fields and data of numeric or character type; for all other types, the value Top is adopted.

<ul> <li>Application Bar</li> </ul>		Z Filter		
GRID	Grid1Table EB CustomerId		CustomerId True	
	Customerid	<ul> <li>Appearance</li> <li>Label Position</li> </ul>	None	
	Customerid	Class	Attribute	
	CustomerName	Visible Invisible Mode	False Collapse Space	
	CustomerEmail	Enabled	True	🖾 Start Page 🗙 🛅 ViewCustomers 🗙
CustomerAddress		Format Invite Message	Text Customer Id	Layout         Rules         Events         Conditions         Variables         Documentation           * Application Bar
Photo	Name Email Country SS			ImainTable       ImainTable       ImainTable       ImainTable       ImainTable         ImainTable       ImainTable       ImainTable

For CustomerAddress, we will choose the value None, so the label disappears.

We move the name, email, and country attributes below CustomerId. Since we don't want to show the CustomerId, we can remove it or set its Visible property to False. If we need to obtain other data from a selected customer, we must send by parameter the value of its identifier, so we should hide the attribute by setting the Visible property to False instead of deleting it.

Below the Visible property is the Invisible Mode property, which is set to Keep Space, the default value. This means that even if we hide the attribute as we did, it will still keep the space it takes when it is visible.

If we open the combo, we see that we also have the Collapse Space value, which means that the attribute will be hidden and the space will be collapsed, allowing the other controls to take the available space. However, this option only works to collapse row data (horizontal collapse), but not columns (vertical collapse). Therefore, since CustomerId is in the same row as CustomerPhoto, we must remove CustomerId from there, and if we move it to the last row, it will be collapsed.

We are going to rearrange the controls a bit to achieve the design we wanted. Since we need the address to take the space of the two columns, we go to the Cell Information section and in the Col Span property we enter the value 2. And press F5...

	avigation View X 🔯 ViewCustomers	×		- × -	nager Window Tools Test Help		
Application Bar     MainTable		501 ¢ 0 ≣ ← View Custon	₹⊿ ers	<b>●</b>	Stat Page X 🔯 Nevigation View X 🔽 ViewCustome		U
GRID		9	John Smith	*	Application Bar     MainTable     Grid1     Grid1Table	← View Customers	* <i>A</i>
~	CustomerName		jsmith@hotm 🎦	0		John Smith	*
	CustomerEmail	Sth. Avenue 254	Anna Brown		CustomerName	Sth. Avenue 254	jsmith@h
CustomerAddress		24	abrown@gma	Q	CustomerEmail	Anna Bro	Ø
0 0 0	•/2	7th.Avenue 123	9	< <p></p>	o CustomerAddress	7th Avenue 123	abrown@
o o	0	0	Peter Eliroy pe@gmail.com	0	Customerid	Peter Ellroy	0
		11th Avenue 789	9			11th Avenue 709	pe@gmail
		4	Chiara Pannini chiara@gmail 💟			Chiara Pa	***
Any Platform, Default O	rientations • 😋 Add Layout 🙁 De	Sth.Avenue 455	Ŷ			Sth.Avenue 455	chiara@g
Cutput Show: Build	• XQ =	AL.	Manuel García		Any Platform, Default Orientations • O Add Layout O Dek		0
Command: adb -s emul	xecution started ator-5554 install -r "C:\KBs	1st.Avenue 111	mgarcia@gm	\Android\M	Cotput Show: Build		mgarcia@
Performing Streamed Success Command: adb -s emul	Install ator-5554 shell am start -n	A	Rogie Duboi	xus.androi:	Command: adb -s emulator-5554 install -r "C:\K8s\ Performing Streamed Install	SXC Rogie Du	roid\/Main%
Starting: Intent { a Success: Android Exe Success: Run MainMen		_		.genexus.m	Success Command: adb -s emulator-5554 shell am start -n c Starting: Intent { act=android.intent.action.NAIN		android.com exus.mobile
					Success: Android Execution Success: Run MainMenu		

We tap on the View Customers icon and see that the list of customers is loaded.

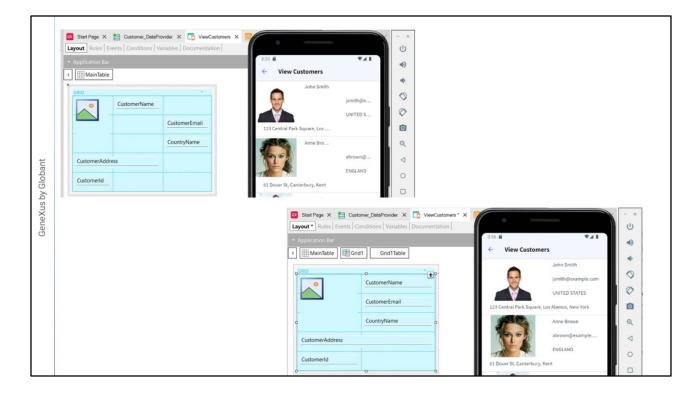
Although it may be similar to what we need, it doesn't look very good. We do notice that CustomerId is not displayed and that the place it would take up is collapsed.

Let's try to improve the size of the photo a little bit and have it take 3 rows, setting the Row Span property to 3.

Before we continue with other changes, let's use the Live Editing functionality provided by the GeneXus IDE.

To see the changes with Live Editing, in this combo box we change the Release value to Live Editing and run the application again.

We check that Live Editing is working by looking at the Live Editing tab of the Properties window, where the IDE is connected to the emulator we had running. We also see that the Live Inspector window has opened, where we can identify the screen controls and their properties.



Returning to our design, we see the client's name, email, and country in different columns. We place them in the same column. Thanks to Live Editing, the change was immediately reflected in the emulator without having to save or rerun.

This allows us to make changes to the design, such as changing the position or alignment, assigning another class, or changing the properties of a class and thus improving the appearance of the screen and seeing the changes instantly.

We are now going to delete the column that is empty, so we right-click and choose Delete Column. Once again, we see that the change is already visible in the emulator.

Layout * Rules Events	Conditions Variables Documer	tation			General		
					🗄 👌 🌾 Filter		
📢 🌐 MainTable 🚦	Grid1 Grid1Table			•	V Table: Grid1Table		
					Control Name	Grid1Table	
GRID	•	P		- 1	Layout Name		
	CustomerName			- 1	Appearance	50%:50%	
	CustomerEmail			_	Columns Style Rows Style		
-		Columns Style		×	Width	pd;pd;pd;pd;pd 100%	
0	CountryName	0			Height	pd	
CustomerAddress		Colu Width	Unit O Percentage		Auto Grow	True	
		2 50%	Device		Class	Table	
CustomerId			Independent     Pixel		Background	(none)	
0	0	0	Value 50	- 1	Visible	True	
			Value 00	- 11	Invisible Mode	Keep Space	
				- 1	Enabled	True	
			OK Cancel	11	V Scroll Behavior		
				-	Scroll Factor	1	
					Zoom Factor	0	
Any Platform, Default C	Drientations • 🗘 Add Layout 🔇 De	lete Layout			Properties Toolbox	Tests Explorer 🔂 Live Editing	
						9	41 W
	= Device Ind	lenendent	Pixel			Display	resolution = pixels per inc

To adjust the size of the columns and rows, we select the grid table, called Grid1Table. Below the Appearance section, we find the Column Style, Row Style, Width, Height, and AutoGrow properties that allow us to change the size and behavior of the controls.

In ColumnStyle, it says 50%;50% which means that the table has two columns and that each one is taking up 50% of the available width. We open this property and see the two columns identified and that each one has the Percentage unit selected with a value of 50.

The other unit of measurement is DIPs: Device Independent Pixels. The Device Independent Pixel corresponds to an abstraction of a pixel that then an application converts into physical pixels, which allows scaling to different screen sizes. The dip value has a different number of pixels for each platform.



The total width is fixed and is 384 dip, so with this design each column takes up 50% or 192 dip.

But if there is a column defined in dip, the percentage values are relative to the value that results from subtracting the fixed values (in dips) from the total width. For example, if we want to separate the photo column from the text column, we can add a column as a separator.

If we had three columns, the first with 30%, the second with a fixed value of 10 dip and the third with 70%, the values of the first and third columns would be obtained by applying these percentages to the value resulting from subtracting the sum of the fixed values (here only one, 10 dip of the separator column) from the table width.

For this reason, the space available for the first and third columns is: 384 - 10 = 374 dip; therefore, the first column will have 30% of 374 which is 112 dip, and the third one the remaining 70%, which is 262 dip.

🐼 Start Page 🗙 😫 Customer_DataProvider 🗙 🛅 ViewCu	ustomers * X 🧮 MainMenu X 🔂 I	ive Inspector X	V Properties	
Layout * Rules   Events   Conditions   Variables   Documenta	tion		General	
Grid1Table	nerAddress	Γ	Table: Grid1Table	
		L	Control Name	Grid1Table
GRID ℃			Layout Name	
CustomerName			✓ Appearance	
CustomerEmail			Columns Style	30%;10dip;70%
	Rows Style	×	Rows Style	pd;pd;pd;pd;pd
CountryName	Rows Style	~	Width	100%
	Now Height	Jnit	Height	pd
CustomerAddress	1 pd 2 pd	) Percentage	Auto Grow	True
CustomerId	3 pd (	Device Independent	Class	Table
	4 pd 5 pd	Pixel	Background	(none)
0		Platform Default	Visible	True
"Using the best value depe	v ding op	alue 0	Invisible Mode	Keep Space
the platform and context"			Enabled	True
the platform and context		OK Cancel	✓ Scroll Behavior	
			Scroll Factor	1
			Zoom Factor	0

Let's select the photo, right-click on it, and choose Insert column after. We will specify the Columns Style values as we had seen, entering 30% for the first column, 10 dip for the second one and 70% for the third, and see the result. Because we are using Live Editing, it has changed and now there is a space between the name and the image.

If we now look at the Row Style property, there are 5 rows with the value pd: Platform Default.

This value varies according to the platform and, for the same platform, it also depends on the content of the cell and whether the field has a label or not; if it does, if the label will be displayed at the top or at the left. The pd value corresponds to: "Using the best value depending on the platform and context."

For example, for Android with Label Position = Top, it corresponds to 64 dip, and in iOS to 53 dip.

Design Queters Objects OVMabileQourseDCO	> 🔁 Localization	Name		Any Android Device
Design System Object: GXMobileCourseDSO	Files	OS		Android
	🔝 Images	Version		
GXMobileCourseDSO X	V Platforms	Device Kind	d	Any Device Kind
Tokens Styles Documentation	Any Platform	Size		Any Size
1 styles Base {	Any Phone	Style		GXMobileCourseDSO
2 @import GeneXusUnanimo.UnanimoMobile;	Any Tablet 7"	/		
3	Any Tablet 10"		~ Platform: Any A	Apple Device
0 references	Any TV		Name	Any Apple Device
4 ImageRound{	Any Watch		OS	Apple
5 @include Image;	Android Device		Version	
<pre>6 border-radius: 80px; 7 gx-content-mode: fill-keeping-aspect;</pre>	Android Tablet 7"		Device Kind	Any Device Kind
8 }	Android Tablet 10"		Size	Any Size
9			Style	GXMobileCourseDS
	x X <> ♥ ☐ Proportion General E ♀   ♥   Filter	•	Style	GXMobileCourseDS
Image: Conditions         VewCustomers* X         Image: Analysis X         Nevigation View         X         Conditions         Live Inspector           Layout*         Rules         Events         Conditions         Variables         Documentation	x X ( ) 것 O Properties General 문화 21 월 Filter · Attribute/Variable: Custon		Style	GXMobileCourseDS
VewCustomers* X      MainMenu X     Impact Analysis X     Nevigation View X     C     Live Inspects     Layout*     Rules     Events     Conditions     Variables     Documentation     Application Bar	x x < > < C Properties General E 2 1 % Filter	CustomerPhoto	Style	GXMobileCourseDS
VewCustomens* X MantMeru X Impact Analysis X Marcalon View X Marcalon Vie	x x < C ⊂ C Proportion General		Style	GXMobileCourseDS
VewCustomers* X      MainMenu X      Impact Analysis X      Navigation View X      Lev Inspect     Layout* Rules Events Conditions Variables Documentation     Application Bar     MiniTable      Grid Grid CustomerPhoto     GRO	* X C C Proportion General 대한 가 Filter Attribute/Variable: Custon Attribute Readonly · Appearance	CustomerPhoto True	Style	GXMobileCourseDS
VewCustomens* X MantMeru X Impact Analysis X Marcalon View X Marcalon Vie	x x < C ⊂ Proportion General E S S Filter Attribute Viriable: Custon Attribute Readonly < Appearance Label Position	CustomerPhoto True	Style	GXMobileCourseDS
VewCustomen* X MainManu X Impact Avalysis X Mavigation View X Colline Inspect     Live Inspect     VewCustomen* Rules Events Conditions Variables Documentation     Application Bar     MainTable Impact Avalysis X MainTable     Grid1 Table     Grid1 T	* X C C Proportion General 대한 가 Filter Attribute/Variable: Custon Attribute Readonly · Appearance	CustomerPhoto True	Style	GXMobileCourseDS
VewCustomen* X MainManu X Impact Avalysis X Mavigation View X Colline Inspect     Live Inspect     VewCustomen* Rules Events Conditions Variables Documentation     Application Bar     MainTable Impact Avalysis X MainTable     Grid1 Table     Grid1 T	x X C ♥ Proportion Constrail	CustomerPhoto True None ImageRound	Style	GXMobileCourseDS
VewCustomers* X MainManu X Impact Avalysis X Mavigation View X Coll Live Inspect      VewCustomers* Rules Events Conditions Variables Documentation      Application Bar      MainTable Grid1 Grid1Table IBB CustomerPhoto      CustomerPhane      CustomerPhan	x x C C Properties General	CustomerPhoto True None ImageRound True	Style	GXMobileCourseDS
VewCustomers* X MaiMenu X Impact Analysia X Mavigation View X 62 Live Inspect     Layout* Rule: Events Conditions Variables Documentation     Application Bar     MainTable Emont GridTTable EMICustomerPhoto     CustomerEmail     CustomerEmail     CountryName	x x C C Proportion General	CustomerPhoto True None ImageRound True Keep Space	Style	GXMobileCourseDS
VewCustomers* X MainManu X Impact Avalysis X Mavigation View X Coll Live Inspect      VewCustomers* Rules Events Conditions Variables Documentation      Application Bar      MainTable Grid1 Grid1Table IBB CustomerPhoto      CustomerPhane      CustomerPhan	x x C Proportion General Constant Canada Canada Canada Canada Attribute/Variable: Custon Attribute/Variable: Custon Attribute/Variable: Custon Attribute/Variable: Custon Attribute/Variable: Custon Casa Visible Invisible Mode Enabled	CustomerPhoto True None ImageRound True Keep Space True	Style	GXMobileCourseDS

We check the execution and the rows look good, so we leave these values.

Now we are going to change the appearance of the photo so that it is round. To do so, we are going to assign to the photo control a class with properties that enable this.

We open the design system object GXMobileCourseDSO that was imported with the xpz, and in its Style tab we create a class named ImageRound. We assign to the border-radius property a value of 80 pixels that will make the image look round, and to the gx-content-mode property the value fill-keeping-aspect so that the image will automatically adjust to the available size without changing its aspect ratio; that is, the proportion between width and height.

For this Design System Object to be taken into account in mobile objects, we must assign it to the platforms in which we are going to generate. So, we open the Customization node and select AnyAndroidDevice; in its Style property, we assign the design system GXMobileCourseDSO.

We do the same for the AnyAppleDevice platform.

Now we select the photo and in its Class property we write ImageRound.

VewCustomes * X MainMenu X ↔ Live Inspector X [     Layout * Rules   Events   Conditions   Variables   Documentation       Application Bar		- × Properties ral Trial Trial Filter + tribute/Variable: Custor	
MainTable     Grid1     Grid1Table     Grid1Table     Grid1Table	• View Customers	Attribute/variable: Custor	
	John Smith	ttribute	CustomerPhoto
GRID	jsmith@example.com	eadonly	True
CustomerName		Appearance	
CustomerEmail	UNITED STATES	Label Position	None
	123 Central Park Square, Los Alamos, New York	Class	ImageRound
CountryName	Anne Brown	Visible	True
oo		Invisible Mode	Keep Space
CustomerAddress	abrown@example.com	Enabled	True
	ENGLAND	O Format	Text
CustomerId	61 Dover St, Canterbury, Kent	Invite Message	Customer Photo
	Peter Ellroy	Link	
		+++ Autolink	True
	pe@example.com	Accessibility	
	UNITED STATES	Accessible Name	Custom
	83 Fortune Dr, Irvine, California	Accessible Name Custom	Customer Photo
	Chiara Pannini	V Control Info	
Any Platform, Default Orientations - O Add Layout O Delete Layo		Properties Toolbox	Tests Explorer 😚 Live Editing
E Output	chiara@example.com		
Show: General	ITALY		
Generating Design System GXMobileCourseDSO	Via della Foce Micina 118, Fiumicino, Roma		

Note that as soon as we assign the class to the CustomerPhoto attribute, we can see the changes thanks to Live Editing. We could even change the values in the class and we would also see the change reflected automatically without even saving or compiling the object or running the application. This facilitates incremental development.

In this example, we are looking at the customer's photo, which is an image stored in the database. However, we often need to use a fixed image, such as a logo or a background image.

In this case, we must take into account that when adding the image to the KB, we must actually add several images with different resolutions, since the devices running the application may have more or less screen density and resolution.

	Working with images	🔚 Ima	ges X												
	<b>,</b>	Name:	1		Modul	r 💮r	Root Module			× …	More >>				
		4	Nome	Module	Description	Modified Dat	Last User	Imps 1	files						
		-	GXMobile	Root Mod	GX Mobile	9/10/2023	ARTECHIr.	27/6	da anno						
		-	GXMobile_	Root Mod	GXMobile	11/10/202	ARTECHIr_		AGENCY		AGENCY				
		-	Img3b14f8	Root Mod	Img3b14f8	21/7/2021	ARTECHIr		Logo.png		.ogo@1x.png				
		L.	italy	Root Mod	italy	20/9/2023_	ARTECHIr.	27/8	rogopiig		ogole ix pilg				
		-	John	Root Mod	John	17/10/202	ARTECHIr_	÷							
		-	Li	Root Mod_	Li	11/10/202	ARTECHr.	-	AGENCY		ACENCY				
		2.	Logo	Root Mod	Logo	6/10/2020	ARTECHIr.	and the second se	ALA.		autorit				
		-	LondonTo_	Root Mod	London To	26/8/2020	ARTECHIr.		Logo@2x.pn	, L	.ogo@3x.png				
		1	LongBridg_	Root Mod	Long Bridg	14/3/2019	ARTECHIr.								
			Louvre_Ex.		Louvre_Ex.	15/3/2019	ARTECHI/								
			Louvre M.	Root Mod	Louvre_M.	15/3/2019	ARTECHI		AGENCY						
			Manuel	Root Mod	Manuel	17/10/202	ARTECHI	2.7.5	Logo@4x.pn						
			indiada.		and to be				Logo@4x.pnj	9					
							🔛 images 🗙	🔛 Logo 🗙						V O Properties	
							Images Doca	mentation						2 7 Filter	
							Image Image		Style	Langua	" Density	Laver	Options	<ul> <li>Image: Logo.png</li> <li>Name</li> </ul>	Logo.png
							A THE LODOR	ing				None		Description	Logo.png
							RAE Logoup KENCY Size:15	ит) ібх44 рх	Any	Алу	Unknown	reone		Is External	False
							A BE Logod		Any	Any	100% (mdpi Android, 1x iOS, 1x Web)	None		Style	Any
							HENCI Size 23	34x66 px (234x66	i dip)	Any	Tuths (magi Anatola, 1x ius, 1x web)	none		Language	Any
							A THE Logod	92x.png	Any	Any	200% (xhdpi Android, 2x iOS, 2x Web)	None		Density	Unknown
							EEC Size 31	2x88 px (156x44	(dip)	(and	CONTROLOGY PROPERTY OF THE A	reurre		Layer Rendering Mode	None Automatic
							A RAE Logot	P3x.png	Any	Any	300% (xxhdpi Android, 3x iOS, 3x We	None		Flips for Right to Left	False
								58x132 px (156x4	H dip)					Source	C:\Modets\Sor
							RAL Lopod	Hx.png	Any	Any	400% (xxxhdpi Android, 4x Web)	None			
							Preview								
										-	TRAVEL				
										2	TRAVEL AGENCY				
							Strech (with a	aspect ratio)							
í														Real and the second sec	

We select the image named Logo and see that there are actually several associated images. If we open the image, we can see that each image has a different resolution; when running the application, the appropriate one will be automatically selected depending on the screen resolution of the device.

We can also see that for each image we can specify which style it will use; that is to say, the associated Design System. This allows us to upload different images associated with different Design System Objects. In our example, the value Any means that the same image will be used for all the styles defined in the application.

We can also associate a specific language to it; for example, if the image includes a text in its graphic and this text varies by language, so that we can upload several associated images, one for each language defined in our application.

We can upload images with different density. For example, the value 100% corresponds to medium density (approximately 160 dpi) which is taken as a baseline for Android, as well as for Web and iOS; in the latter case, the value corresponds to screens that are not Retina Display. Remember that dip (Device Independent Pixels) are then converted to actual pixels according to the resolution of each device.

The value 200% is for Extra-high-density (about 320 dpi) corresponding to 2x Retina images for iOS and 2x for Android. A value of 300% corresponds to Extra-extra-high-density with about 480 dpi, which is for Retina 3x for iOS and 3x for Android, and 400% corresponds to devices with 640 dpi. As devices with higher resolution values become available, this value will increase.

Layout  Rules Events Conditions Variables Documentation	11:00	U	2 Filter	
Application Bar	← View Customers	•	Attribute/Variable: Countryl	Name
ImainTable     ImainTable     ImainTable     ImainTable       ImainTable     ImainTable     ImainTable     ImainTable	-		Attribute	CountryName
GRID ·	John Smith	0	Readonly	True
CustomerName	jsmith@example.com		~ Appearance	
	UNITED STATES	$\Diamond$	Label Position	None
CustomerEmail	123 Central Park Square, Los Alamos, New York	D	Class	AttributeBold
CountryName	Anne Brown		Visible	True
		Q	Invisible Mode	Keep Space
CustomerAddress	abrown@example.com	⊲	Enabled	True
	ENGLAND	0	Format	Text
CustomerId	61 Dover St, Canterbury, Kent		Invite Message	Country Name
	Peter Ellroy		~ Link	
	pe@example.com		Autolink	True
			~ Accessibility	
	UNITED STATES		Accessible Name	Custom
	83 Fortune Dr, Irvine, California		Accessible Name Custom	Country Name
	Chiara Pannini		v Control Info	
Any Platform, Default Orientations • 😌 Add Layout 🔇 Delete Layo	chiara@example.com	•	Properties Toolbox 🗟 Te	sts Explorer 🔂 Liv
Cutput	ITALY			
Show: General · X   Q ⇒				
Generating Design System GXMobileCourseDSO	Via della Foce Micina 118, Fiumicino, Roma			

We return to developing our application.

Now we are going to show the name of the country in bold, so we create an AttributeBold class that inherits from the Attribute class, with the following properties: font-weight in bold and the color property set to the value: \$colors.AttributeBolded that we had defined as a token. Now we assign the class to the name of the country.

Also, we adjust all the data of the second column with Horizontal Alignment set to Left and Vertical Alignment set to Middle. The direction is also aligned horizontally to the left and vertically to top.

	Adding a combobox	to filler by country	Values Editor			×	Invite Message	country ia
							Link	
			Name Name1	Value 0	-	Add	Autolink	True
				0		Remove	~ Accessibility	
						Edit	Accessible Name	Custom
						Move Up	Accessible Name Custom	Country Id
	- & Variables		_				~ Control Info	
	Standard Variables					Move Down	Control Type	👶 Combo Box
	CountryId	Attribute:CountryId					Auto Grow	False
							A Values Empty Item	True
		we have been as	_			ок	Empty Item Text	GX_EmptyItemText
	Layout * Rules   Events   Cor	nditions Variables Documentat				Cancel	~ Cell information	
	0	0		V Control Info	_			
		0		✓ Control Info				
2	Country Id &CountryId ~	· •		Control Info Control Type	👶 Dynamic Combo Box	x	~	
6000000	Country Id &CountryId ~				Bynamic Combo Box	x	~	
	oCountry Id &CountryId ~			Control Type	-	x	×	
	Country Id &CountryId ~			Control Type Auto Grow	False	x	~	
	Country Id &CountryId ~			Control Type Auto Grow Data Source From	False Attributes	x	×	
	Country Id &CountryId ~	CustomerName		Control Type Auto Grow Data Source From Item Values	False Attributes Countryld	x	~	
	Country Id &CountryId ~	CustomerName		Control Type Auto Grow Data Source From Item Values Item Descriptions	False Attributes Countryld CountryName	x		
	GRID	CustomerName CustomerEmail		Control Type Auto Grow Data Source From Item Values Item Descriptions Sort Descriptions	False Attributes Countryld CountryName	x		
	Country Id &CountryId ~	CustomerName CustomerEmail		Control Type Auto Grow Data Source From Item Values Item Descriptions Sort Descriptions Conditions	False Attributes Countryld CountryName	x		
	GRID	CustomerName CustomerEmail		Control Type Auto Grow Data Source From Item Values Item Descriptions Sort Descriptions Conditions Instantiated Attributes	False Attributes Countryld CountryName True	x		
	GRID	CustomerName CustomerEmail		Control Type Auto Grow Data Source From Item Values Item Descriptions Sort Descriptions Conditions Instantiated Attributes Empty Item	False       Attributes       CountryId       CountryName       True	x		

OK. To complete this screen's functionality, we will add a combo box that allows us to select the country and filter the customers by the selected country. To do so, first we create the &Countryld variable which, due to its name, is automatically based on the Countryld attribute, and add it to the panel form at the top of the grid.

If we go to the properties of &Countryld, we see that in ControlType it says Edit, so by default this variable will be seen as an edit field. To change its appearance and behavior, we open the combo of this property and see that we could assign it the ComboBox type, but in this case we should enter the values manually in the Value property to fill the combo.

We already have the countries' data in the database, so instead of this type of combo we choose a Dynamic Combo Box and see that the Items Values property is automatically assigned the Countryld attribute because this is the name of the variable, which will be the value returned by the combo when selecting a country.

The properties box says that we must assign the Items Descriptions property that will be the field whose values will appear in the combo when we open it, so we assign the attribute CountryName to this property. Lastly, as we want the combo box to start without any country by default, we set the EmptyItem property to True, leaving a default value for the Empty Item Text.

	~ Data		Editing Conditions <pre>Editing CountryId_SCountryId when not &amp;CountryId.IsEmpty();</pre>
	Orders (0 orders)		countryig accountryid when not accountryid.isempty(),
	Search	(0 filters)	
	Conditions	CountryId=&CountryId when not &CountryId/	OK Cancel
	Base Trn		
	Unique		
			420 B 9.4 B 430 B 9.4 B
			← View Customers ← View Customers
÷			Country M Country M
ban	Layout Rules Events Conditions Variables Documentatio		United States
Glo	&Countryld.ControlValueChanged		John Smith Rogie Dubol
GeneXus by Globant	<pre>1 Event &amp;CountryId.ControlValueChanged 2 Refresh 3 Endevent</pre>		junith@example.com United States
Xus			123 Central Park Square, Los Alamos, New York 22 Rue Myrha, Paris, Paris
sue)	3 - Endev	Vent	Peter Elizy
Ge			pe@example.com
			United States
			83 Fortune Dr. Irvine, California
		es Events Conditions	
	1	Parm(&CountryId);	

To make the country selected in the combo box act as a filter in the customer grid, we go to the Grid properties and in Conditions we add: CountryId=&CountryId when not &CountryId.IsEmpty();

In this way, only the customers whose country identifier matches the country identifier selected in the combo box will be displayed. If no identifier is selected, there will be no filter and all the customers will be displayed.

Now we go to the events tab and insert an event of the &Countryld variable – ControlValueChanged– and write the Refresh command. This will refresh the contents of the grid when we finish selecting the combo box and load the data from the database again, filtered by the selected country. We will see more about events later.

Finally, we go to the rules and add a Parm rule with the &Countryld variable. This is necessary in this mobile architecture because to minimize server queries we prioritize the use of cached data. Also, for the server to understand that we want to bring new data, we add a Parm rule with the variables we use in the filters, so that when their value changes, the page is updated.

Because we add filters and events that involve the server, to see the changes we have to compile the application again, so we press F5.

We open View Customers and see the combo box at the top that says (None) to indicate that we have not selected a country yet. We choose United States and see that the filter works and only customers from that country are shown. We select another country and check that everything works as expected.



In this video, we saw how to work with basic screen controls, and how to improve their appearance and distribution on the screen. In the following videos, we will continue to see other screen controls with interesting functionalities.