GeneXus for Mobile course

Practical Exercises

GeneXus™ 17

November 2021

CONTENTS
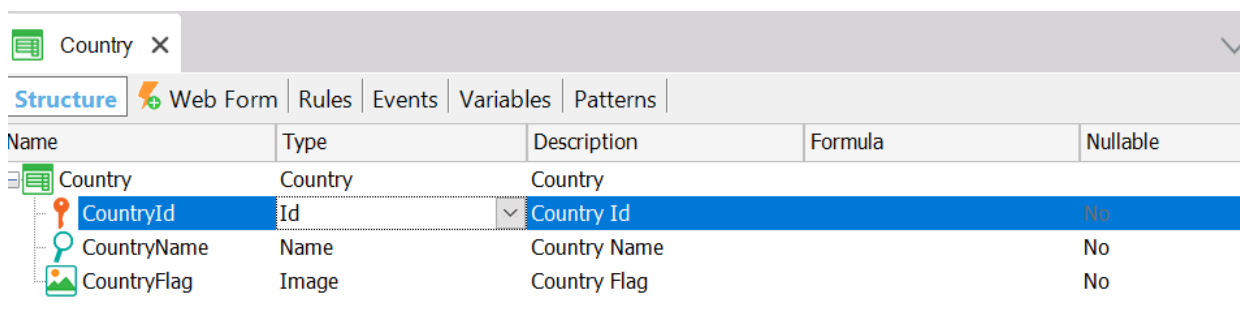
1. GETTING STARTED

Throughout these practice exercises, an application will be developed for an Event (EventDay).

Prototyping will be done in Android because an emulator is provided. Also, if you have a device (running on Android or Apple), you will be able to test the application in it.

- Create a new **Knowledge Base** (if the Trial version is used, it will be created with .NET Environment and Default C# Web generator).

- Note the properties of the Default generator (C# Web —using the Trial version you will find the Generators node in the KB Explorer itself), in particular those of the **Execution** group.

- Use the **Deploy to Cloud** option (Generator property: **Deploy to Cloud = Yes**) —when the Trial version is used, it is the default option and the only possible option.

- Note the properties of the **Default DataStore** that indicates where the application will be created —in the Trial version this information is hidden.

- Note the References node and its contents. The domains as well.

2. WEB APPLICATION AND SEMANTIC DOMAINS

- Create the Country Transaction according to the figure below:

Create the following Domains:

- Id - Numeric(8.0)
- Name - Varchar(100)

In the CountryId attribute, configure the Autonumber property (in the attribute, not in the domain).

- Create the Company Transaction according to the figure below:

Create the following Enumerated Domain:

- CompanyType: Character(1) – Enum. Values: Company (C) – StartUp (S)

- Create the **Speaker** Transaction according to the figure below:

- Configure the Autonumber property in the SpeakerId attribute.

- SpeakerFullName is a formula attribute = format("%1 %2", SpeakerSurname.trim(), SpeakerName.trim())

> Look at predefined domains in GeneXus. Several of them correspond to semantic domains. What does it mean?

- Add these rules:

```
Error( 'The Speaker Name must not be empty') if SpeakerName.IsEmpty();

Error( 'The Speaker Surname must not be empty' ) if SpeakerSurname.IsEmpty();

Msg( 'Curriculum Vitae should not be empty') if SpeakerCVMini.IsEmpty();
```

- Change the tables of the three transactions and create Unique indexes as shown in the figures below:

Company

Structure	Indexes

Attribute	Order	Description
Company Indexes		Company
ICompany	Primary Key	Automatic Index
UCompany	Unique	User Index
CompanyName	Ascending	Company Name

Country

Structure	Indexes

Attribute	Order	Description
Country Indexes		Country
ICountry	Primary Key	Automatic Index
UCountry	Unique	User Index
CountryName	Ascending	Country Name

Attribute	Order	Description
Speaker Indexes		Speaker
ISpeaker	Primary Key	Automatic Index
SpeakerId	Ascending	Speaker Id
ISpeaker1	Foreign Key	Automatic Index
CountryId	Ascending	Country Id
ISpeaker2	Foreign Key	Automatic Index
CompanyId	Ascending	Company Id
USpeaker	Unique	User Index
SpeakerName	Ascending	Speaker Name
SpeakerSurname	Ascending	Speaker Surname

- Apply the **Work With for Web** Pattern to the three transactions.

- Import the file **1-PopulateCountries_and_Speakers.xpz** into the KB. Here, basically three procedures to initialize the tables with data will be imported (including images of country flags, company logos, and Speakers photos).
  After importing the objects, in the Start Event of the **Home** Web Panel (it is located in the GeneXus/Web folder) add the invocation to the following procedures which were imported with the previous xpz file:

```
Event Start
        InitializeCountry()
        InitializeCompany()
        InitializeSpeaker()
Endevent
```

  Important Note: these procedures were developed to create the data so as not to alter the transactions you created. In GeneXus 15, the natural way to initialize with data the tables of both transactions is to use the "Data Provider" property of the transaction, together with Used to "Populate Data." (To learn more about this topic, you can watch the video http://training.genexus.com/genexus/curso-genexus-15?es#poblar-con-datos-desde-la-propia-transaccion)

- Run the application by pressing F5 (you will have to accept the creation of the database tables). Because you're prototyping in the cloud, if you hadn't already done it in the current session, you will be prompted to enter your GeneXus username:

- Run the **Home** Web Panel to start it and view the loaded data. Open **Work With Countries, Work With Companies** and **Work With Speakers** to check that they have actually been loaded.

## 3. APPLYING THE WORK WITH PATTERN FOR SMART DEVICES

- Apply the **Work With for Smart Devices** Pattern to the Country and Speaker transactions.

- Look at the **List**, **Detail** and **Section(General)** nodes of the Work With for Smart Devices object, and the screens that appear in each case.

- Save and note what happens with the following elements:

  o Transaction properties:

    - **Business Component** = True

    - **Expose as Web Service** = True

    - **Web Services Protocol** = REST Protocol

  o Note how the Work With for Smart Devices object is displayed in the KB Explorer, below the transaction.

  o The generator for **Android and Apple** was added to the Front end in the Preferences tab.

- Create the **EventDay** Menu (keep the name for subsequent object imports), according to the figure below:
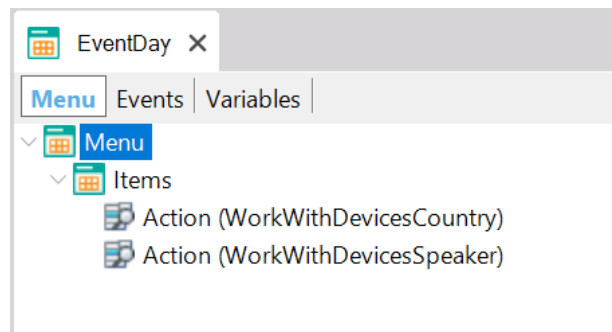
Look at the code of the event associated with each option. In the first item, Work With countries is being invoked. What part of it? Pay attention to the invocation syntax.

- Associate an image with each option (they are provided with this material):
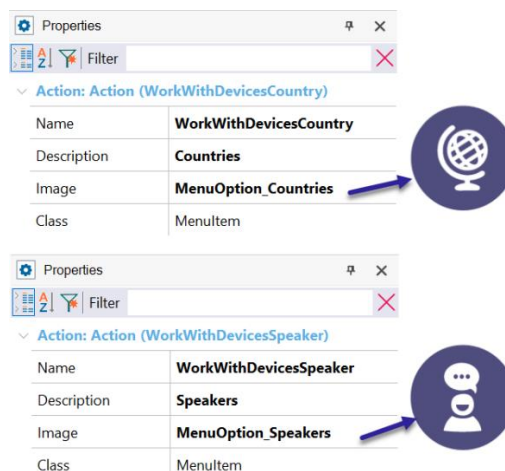
- Also, edit the properties of the EventDay menu, enter a title for the application, and define the app and launch images.

Why are they defined in this object?

- Save and note that the **Main Program** property is set to "True."

- Press **F5** to generate and run the app for the Web and for Android.

Consequences:

- Given that no "Startup Object" has been indicated, GeneXus will open the web Developer Menu in your computer's default browser.

Note: if you have an iOS device and want to test it there, install the KBN on it (in the Developer Menu, look for the QR code to download the KBN), run it, and from the menu choose the "Add Knowledge Base" option and then "+." Scan the QR code displayed in the Developer Menu for the KBN (the only one enabled). Save and test.

- In the **Developer Menu**, in addition to the links to run the Web application objects, QR codes will be displayed. Remember that in the theory lesson you learned what they are all used for.

- The created file will not always be compiled. It will depend on whether it has a configured "Startup object" and on the execution mode used (F5, Run or "Run with this only" over a main object, etc.). In this case you still **don't have the compiled file**, so the QR Code for Android is disabled.
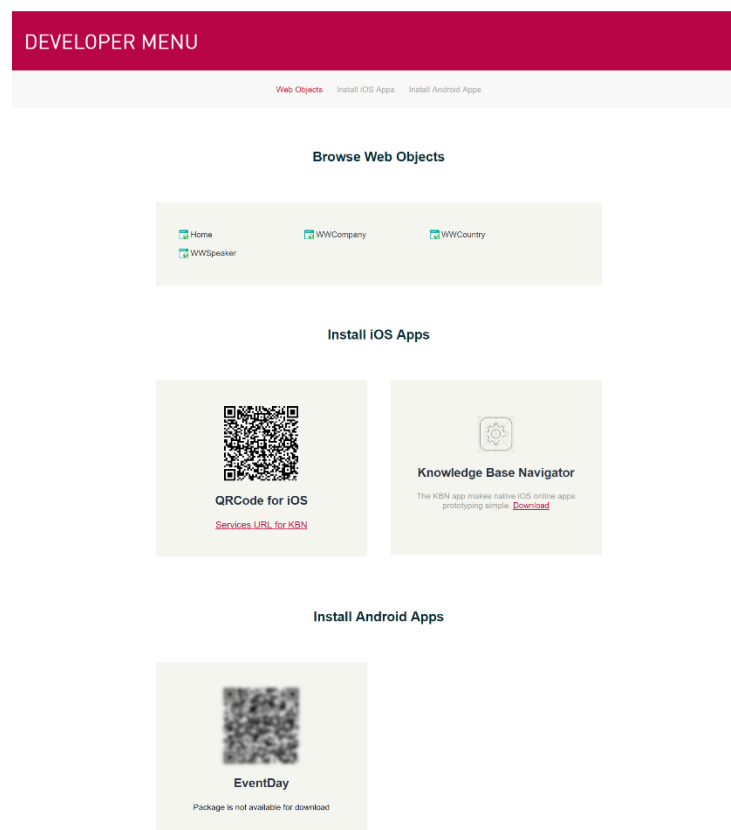
- Since you are running the app for Android, which **doesn't allow** running it with the KBN, an error will be displayed:

- Right-click on the **EventDay** menu object and select "Set as Startup object."

> Verify that the CarmineSD Theme checkbox is selected, and if it isn't, do so before running the app.

- Press F5 to run this object and all its dependencies. Here it will be compiled, and if you refresh the web Developer Menu, you will see that the QR code is now enabled. Also, you'll see that GeneXus will automatically open the Android emulator, in which the application **menu** will be run.
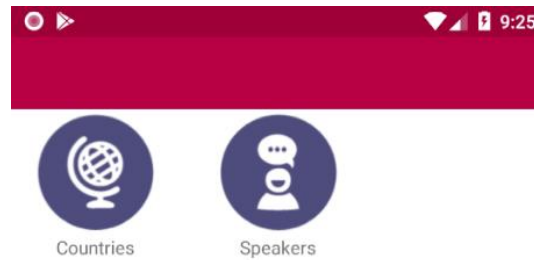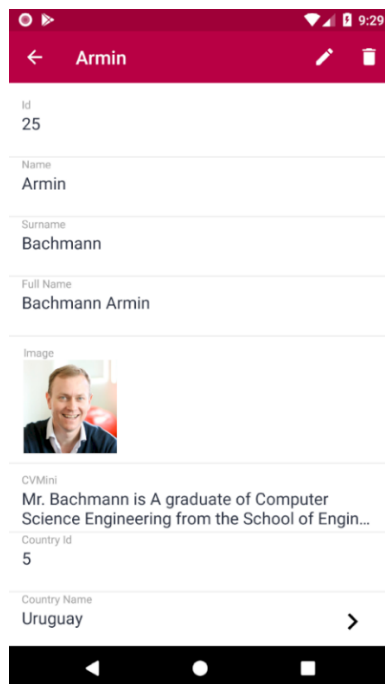

In the emulator:

- Tap on "**Speakers**".

  o Note that the same speakers that were entered from the Web application are displayed.

  o Take a look at the attributes being displayed for each speaker. Go to GeneXus and in the Pattern look for the Layout of this **List**, to see how they match.
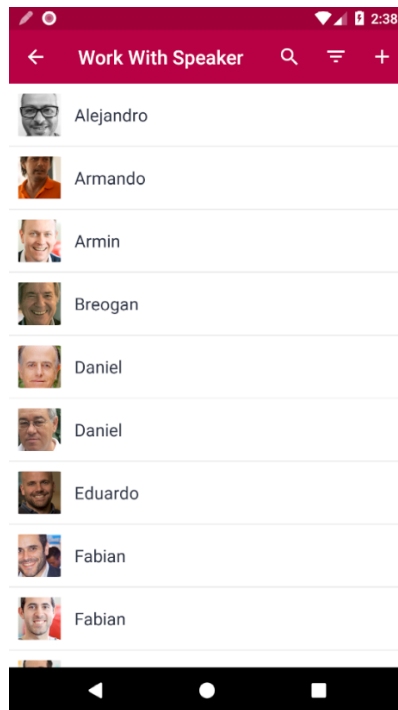
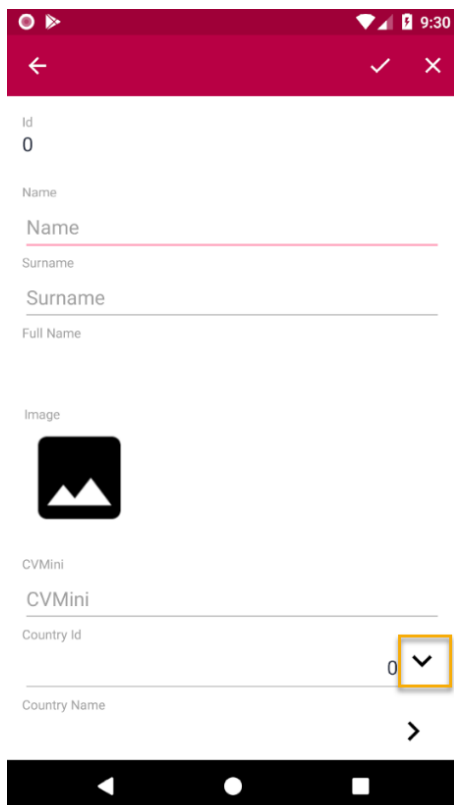- Tap on any speaker and note how their details are displayed (is this **Layout** in **View** or **Edit** mode?):

- **Edit** the speaker being displayed. What layout is being displayed for this **General Section** of the details? Is it of the **View** or **Edit** mode?

- **Insert** a new Speaker. To do so:
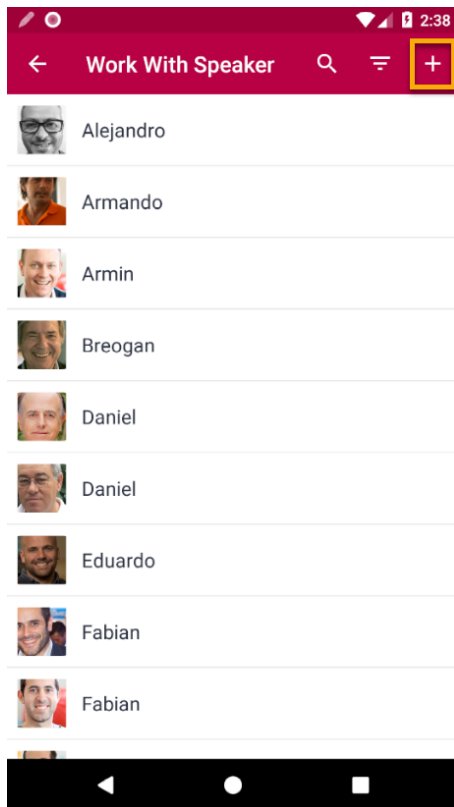
- Try to leave the name or last name blank. Who is invoked by clicking on the image to the right of CountryId?

Save. Why can't you perform the operation? Enter the name or last name and try again. Note how the new speaker is displayed on the list.

- Try to search for speakers by name. Then try by last name. What other information can be searched for? Open the Work With component to see where these searches are configured. Delete any of them if you don't think they are convenient and try again.

- Try to filter by country:
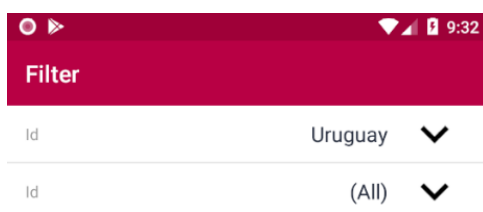
Running the compiled app

As we have said, in Android it is not possible to interpret the application through the KBN. Several options exist to obtain and run the compiled app:

- Right-click on the **Menu** tab and select the option "**Run With this Only**" or "**Run.**"

- Configure the **Menu** as a **Startup Object** (right-click on the object tab/Set As Startup Object, or Preferences/.Net Environment, "Startup Object" property, if you're not using the Trial version). The next time you press F5, only the application that depends on this object will be compiled; in this case, only the application for Smart Devices whose entry point is the Menu. The Web Developer Menu will not be opened, only the emulator with the compiled Menu running (if the device is connected to the computer, it will be opened on the device).

There are more options available that can be found at wiki.genexus.com or in the video "Starting to develop the first Objects for Smart Devices" of the course on Smart Device development.

- Remember that, while compiling, the corresponding **QR Code** will appear in the Web Developer Menu. You can access it from **View / Show QR Codes**:
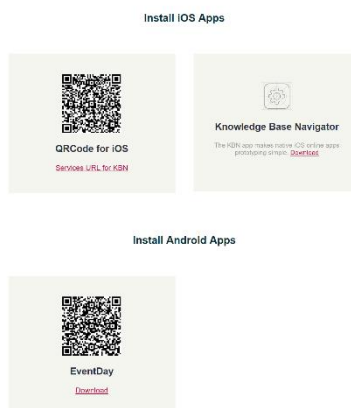

If you have an Android device, try to download and install the compiled application through the QR Code reader. However, the best option will be to connect via cable the device to the computer that is running GeneXus.

4. DEVELOPING THE APPLICATION

Design - Themes and Images

- Look at the themes included in your KB, and what theme is used for each platform (to do so, access the Customization folder, Themes node and SD Platforms → Any Android, Any iOS, etc). Note that by default they have the Theme property set to CarmineSD, CarmineAndroid, or CarmineIOS.

- Import the file XPZ 2-ThemeGX28.xpz containing the Theme GX28Android4 of the Event. Next, set this Theme for the AnyAndroid platform (if you want to, you can configure the GX28IOS Theme for the AnyIOS platform).
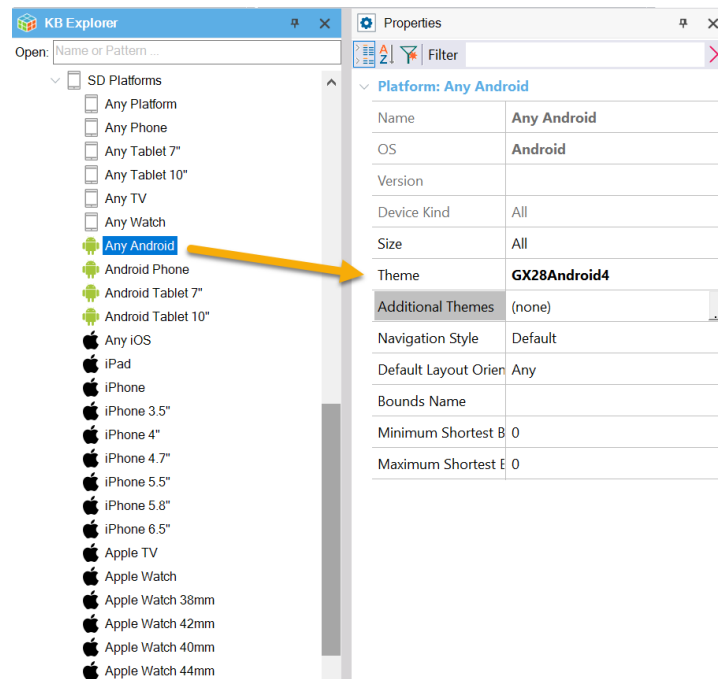
- In the object EventDay Menu, set the **Base Color Scheme** property to Light.
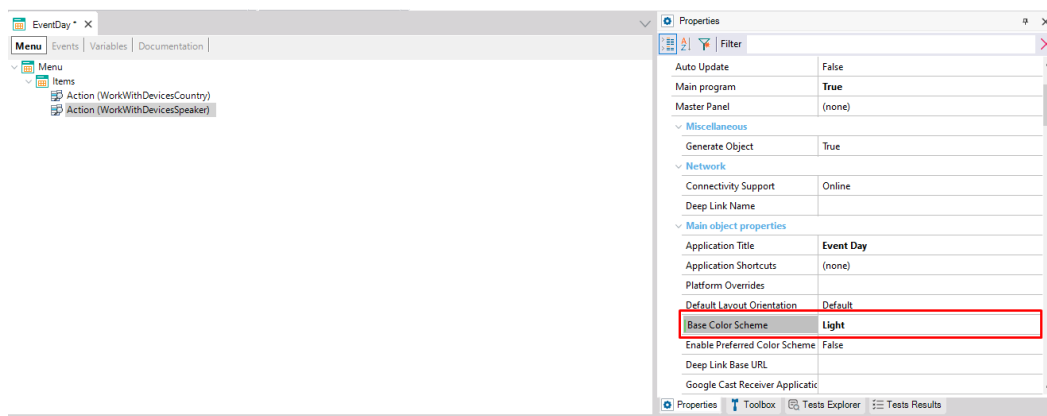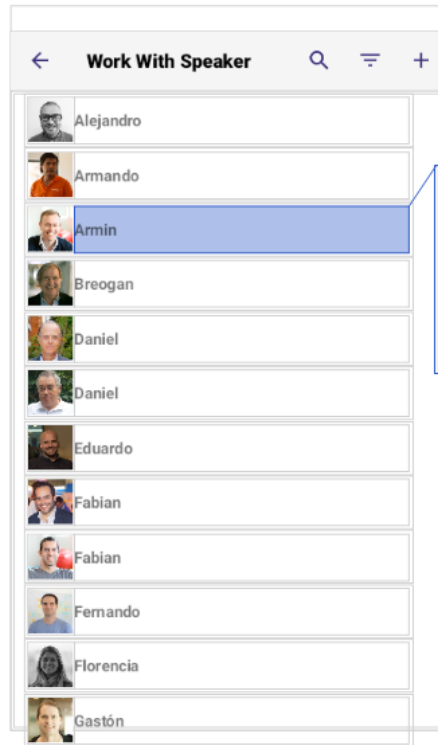

- Enable Live Editing (through the combo in the upper bar in GeneXus). Look for the window with that name and Run again the Menu object, so that Live Editing starts "listening" and runs the application.

- Note how the entry is displayed in the Live Editing window, and the Live Inspector window is opened to show the app running on the emulator or device:
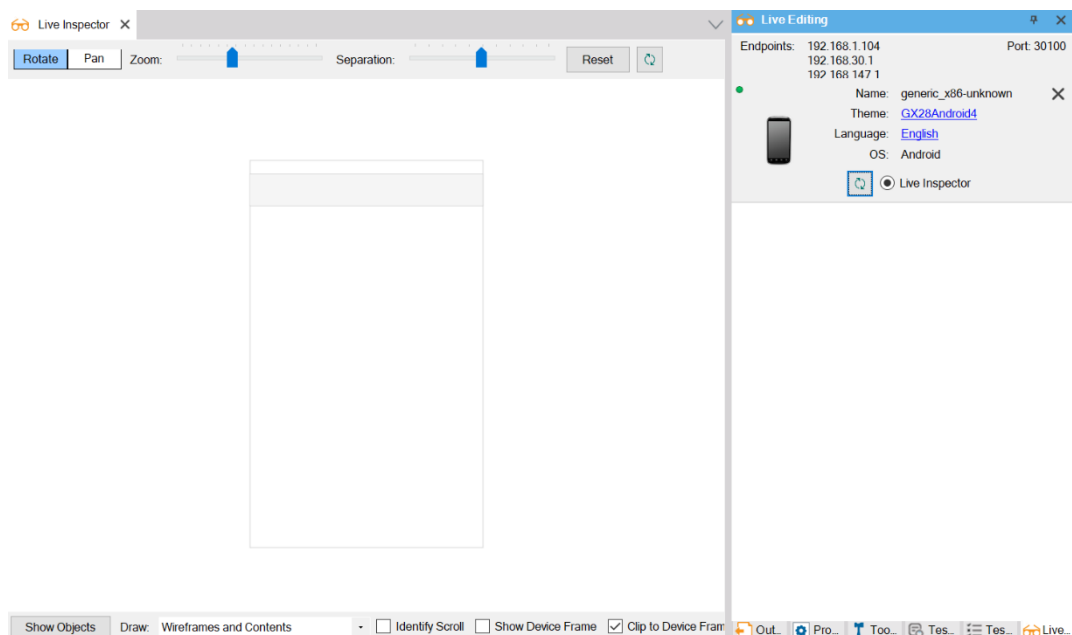
Note: the Menu cannot be viewed using Live Editing; only the panels generated with Work With for Smart Devices and Panels are displayed. Try to access the Speakers List and now you will see the Live Inspector screen:
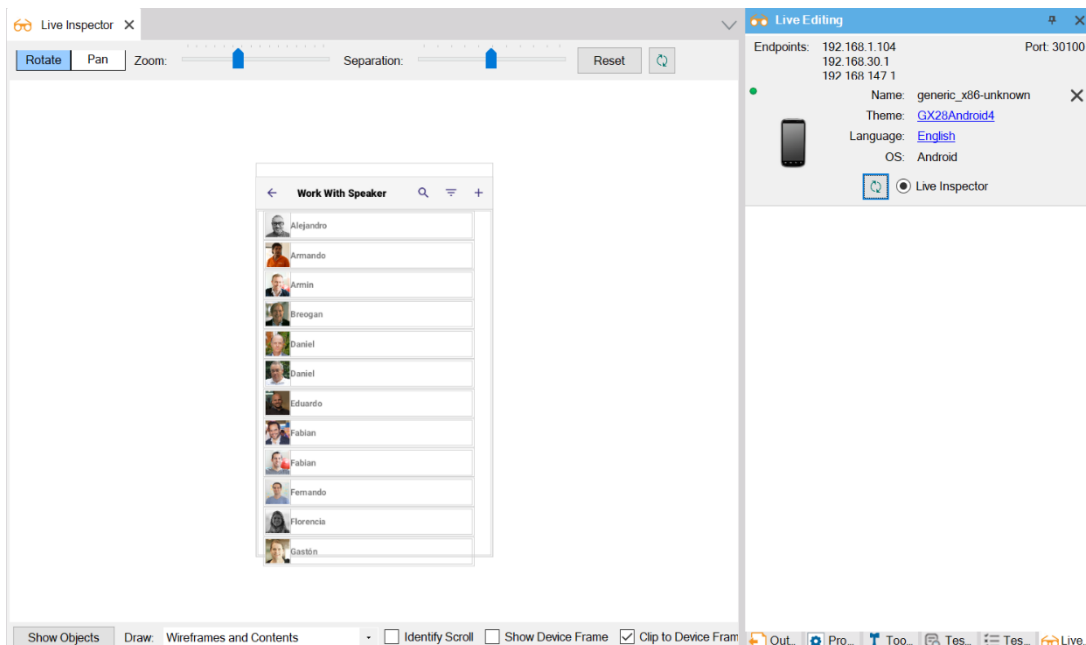

Note that if you click on the name of a speaker in the grid you will see the control in question (the SpeakerName attribute), as well as its class, size and position:

Work With Speaker

Alejandro
Armando
Armin
Breogan
Daniel
Daniel
Eduardo
Fabian
Fabian
Fernando
Florencia
Gastón

Object: WorkWithDevicesSpeaker
Control: Speakername
Class: AttributeTitle
   └─Label Class: TextBlockTitle2
Size: 315 x 44 dip
Position: x:60 y:189 dip

- Rotate the screen with the mouse to see the layers:
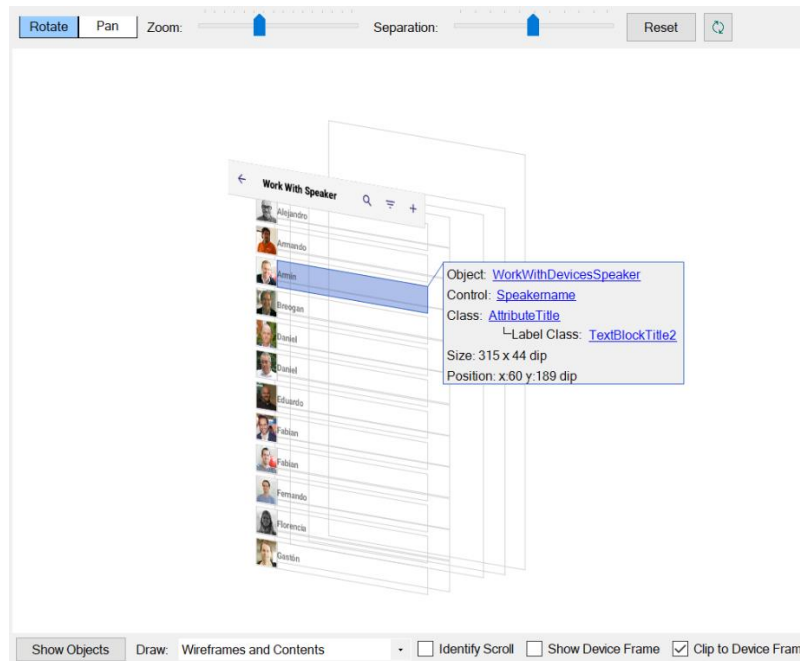
Customizing Objects

- Check that you're inspecting your app with Live Editing. To the **Grid** that shows the list of Speakers, add the speaker's company name –CompanyName (with no tag) below his/her name. Note that your device or emulator will show a blank space; why?
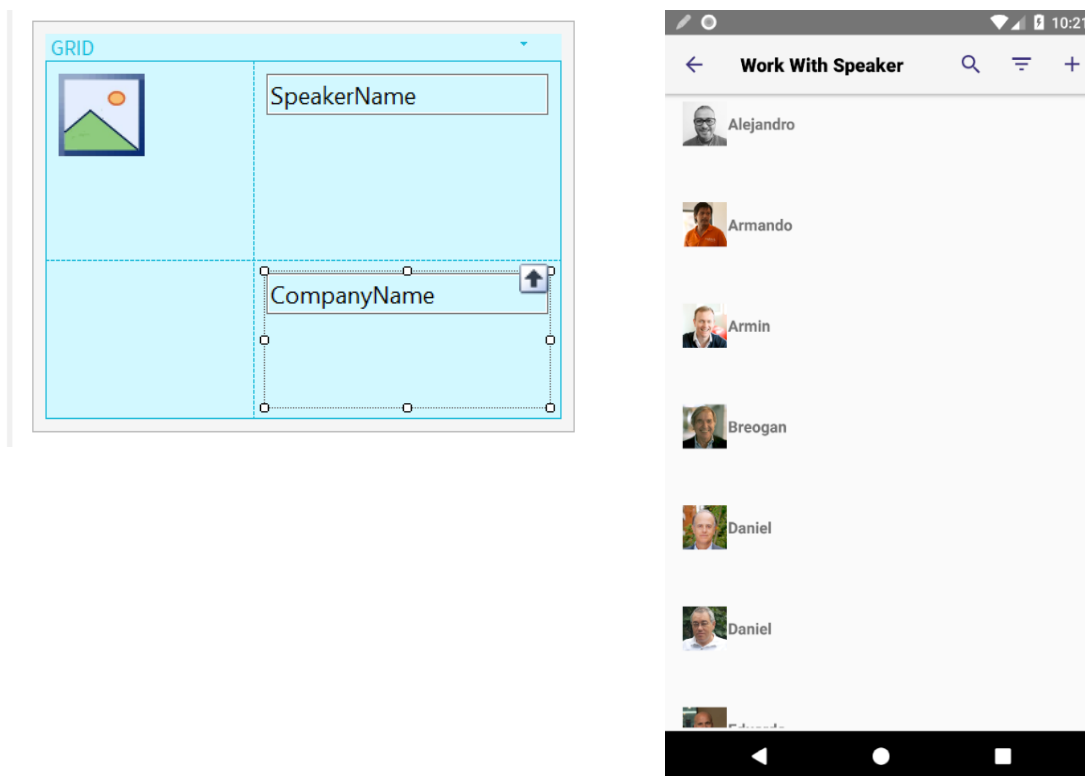
Answer: Live Editing uses metadata to show the results on the device and in the Live Inspector window without compiling, instantly. When adding an attribute control, its content must be searched in the database through the corresponding rest service (data provider); therefore, Live Editing cannot be updated and a message will be displayed that the connection was interrupted. To see the changes and synchronize Live Editing again, the Main object will have to be executed again with (F5 – Run).

- Look at the class associated with the control Att/Var SpeakerName. Delete the SpeakerName attribute and replace it with the SpeakerFullName attribute, also with no tag. To the right, add the SpeakerId (with no tag, hidden). Note two things: 1) The class associated with the new attribute will be Attribute, not AttributeTitle; and 2) the changes made will not be seen now in the Live Inspector window, or on the device. To see them at runtime, select Run on EventDay or F5 if it is a Startup Object.
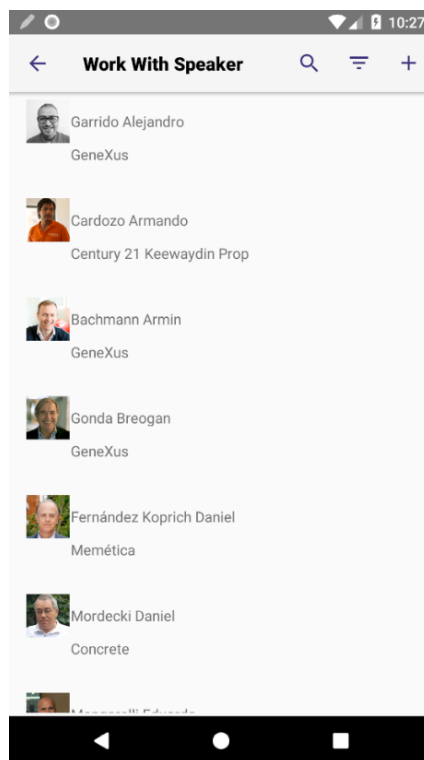
- Several issues should be corrected in this screen: one of them is how controls are arranged. We don't want the photo to be displayed too close to the speaker's name and country. To achieve the desired design we will insert 2 tables into the grid.

    In the outermost table, we will place in the first column –80 DIPs width– the SpeakerImage attribute; in the second column, we'll add another table that will take the remaining space.

    For the second Table, in the first column that takes 100% of the width we will add the controls SpeakerFullName and CompanyName below it. In the second column of 15 DIPs, we'll add SpeakerId (hidden). Also, in the second table we'll need 2 additional rows of 5 DIPs to create a space. A diagram is shown below.
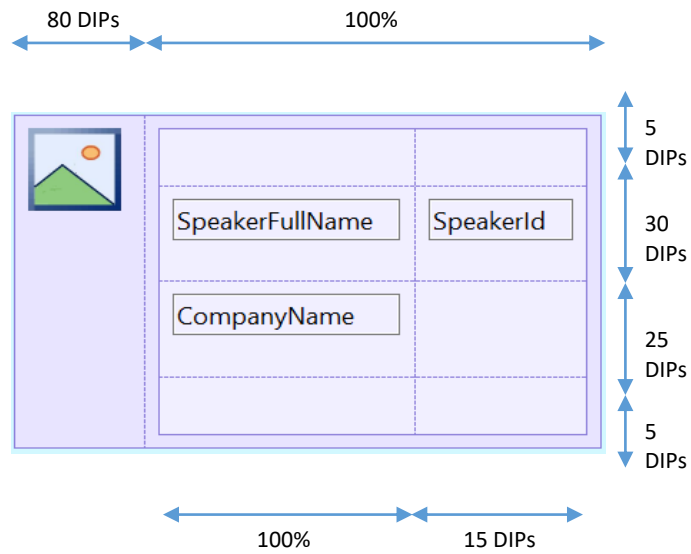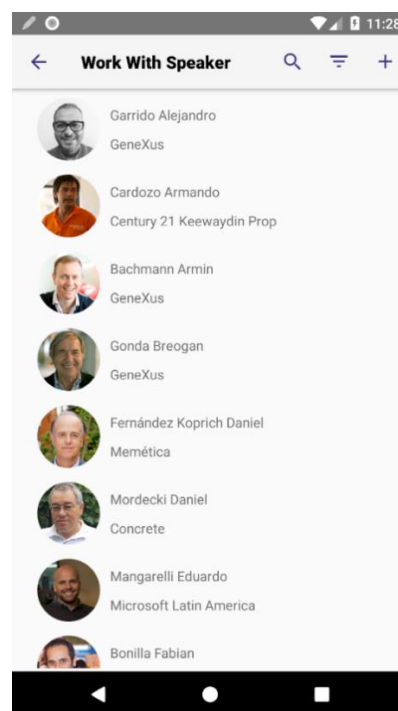
Note: Make sure that the Columns Style property has a value of 100% for the grid table.
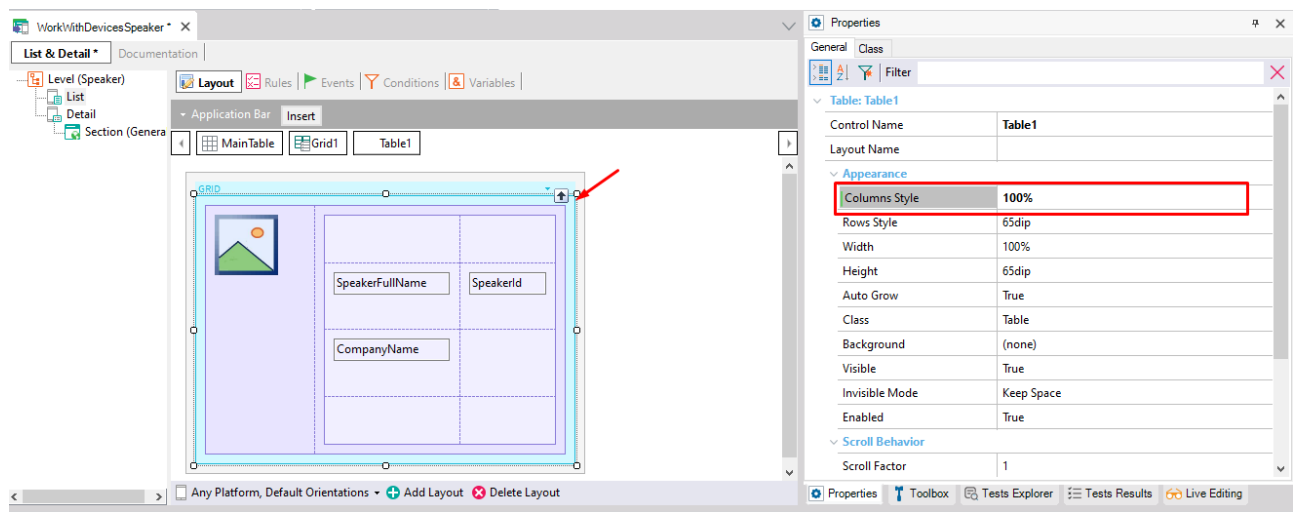

With these values, we will achieve the following result:
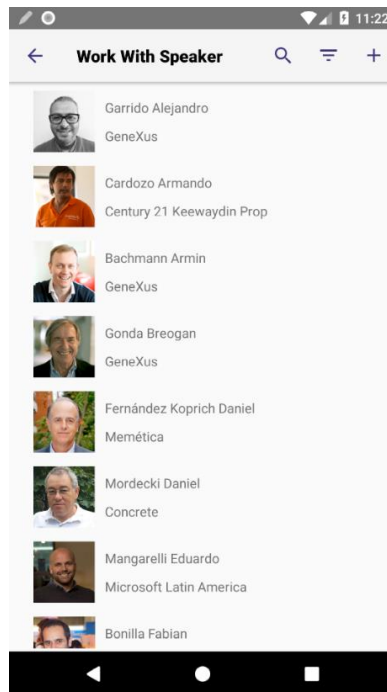
- The only thing missing from our design is to give the image a rounded appearance, and for this we will use the ImageSpeakerList class.

- We want the speaker's full name to be highlighted, and to do so we'll change the SpeakerFullName class and use AttributeTitlegx16; also, in CompanyName we'll use the class AttributeSessionTitle01:
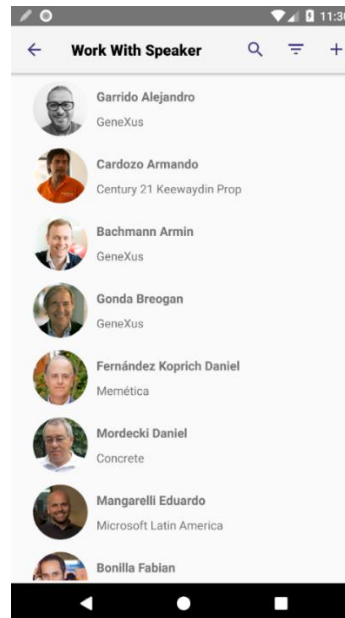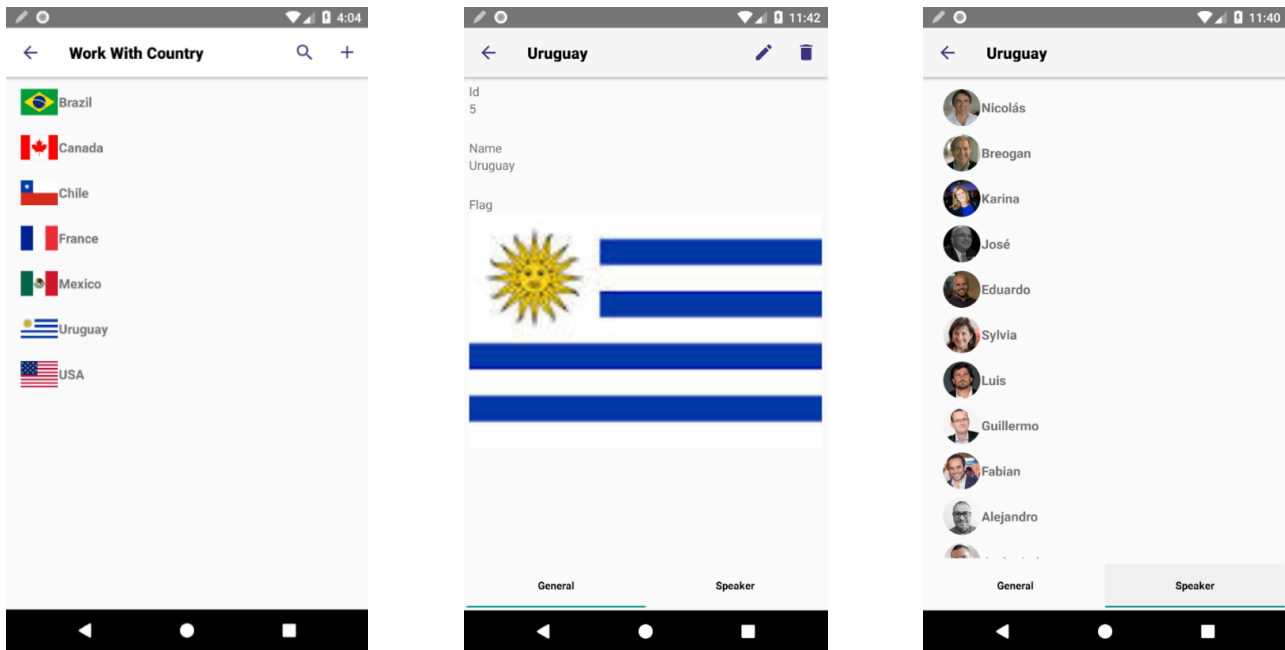

- Note that the List shows the caption "Work With Speaker" at the top. Change it to Speakers (from the List node properties):
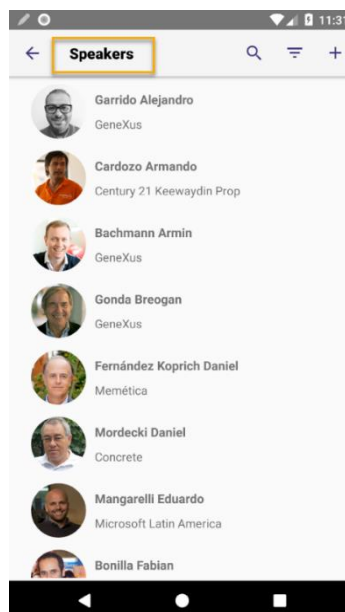
Using Stencils

In the application, access the Countries, select Uruguay and click on the Speakers tab.


Here, it would be preferable to show the same information and with the same design previously achieved.

- Create a Stencil called "SpeakerInfo" from the table contained in the grid of the Work With Devices Speaker List. To do so, right-click on the table and select the option *Wrap As New Stencil*. Save the Stencil and the changes made to the object WorkWithDevicesSpeaker.
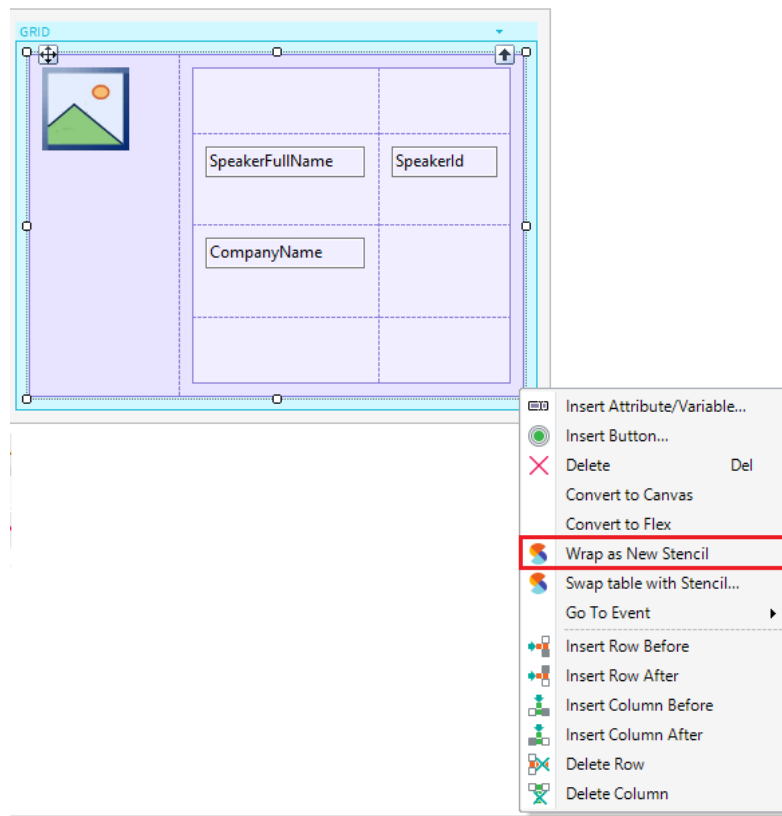
- Review the Stencil you've just created; why are variables used instead of attributes? Why doesn't it have any events?
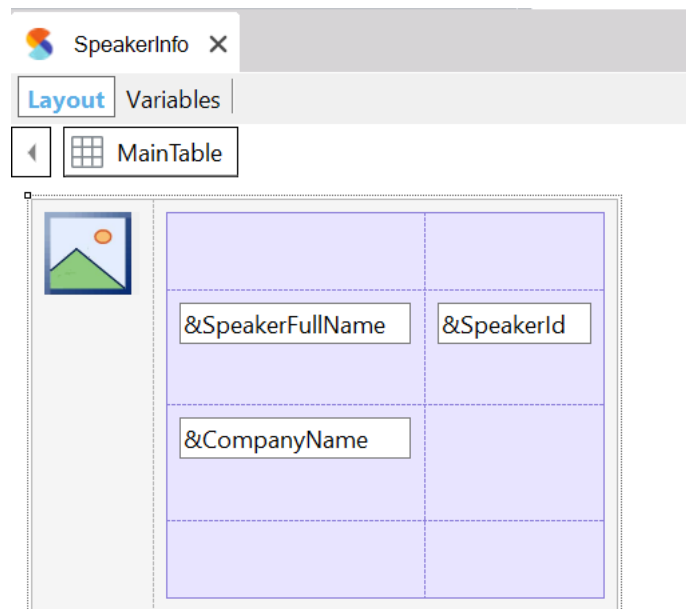
Answer: Stencils allow designing control layouts that can be reused in different objects; in this type of object, you can only change what's related to the controls' design, and for this reason they don't have any events. In addition, there are control properties that apply to their behavior and are not available here either. Since the goal is to reuse designs and not refer to any particular behavior, variables are used instead of attributes.

- Apply the Stencil to the object *WorkWithDevicesCountry*, in the *Section Speaker* of the *Detail* node. To do so, you can use the option *Swap table with Stencil* by right-clicking on a table or inserting the Stencil from the menu options Insert -> Stencil. You can also open the Stencils panel from the View menu -> Other Tool Windows -> Stencils.

  Note: Remember that you must have **a table other than the grid** to apply the Stencil when using the option Swap table with Stencil. In addition, make sure that the Columns Style property of the grid table has a value of 100%.
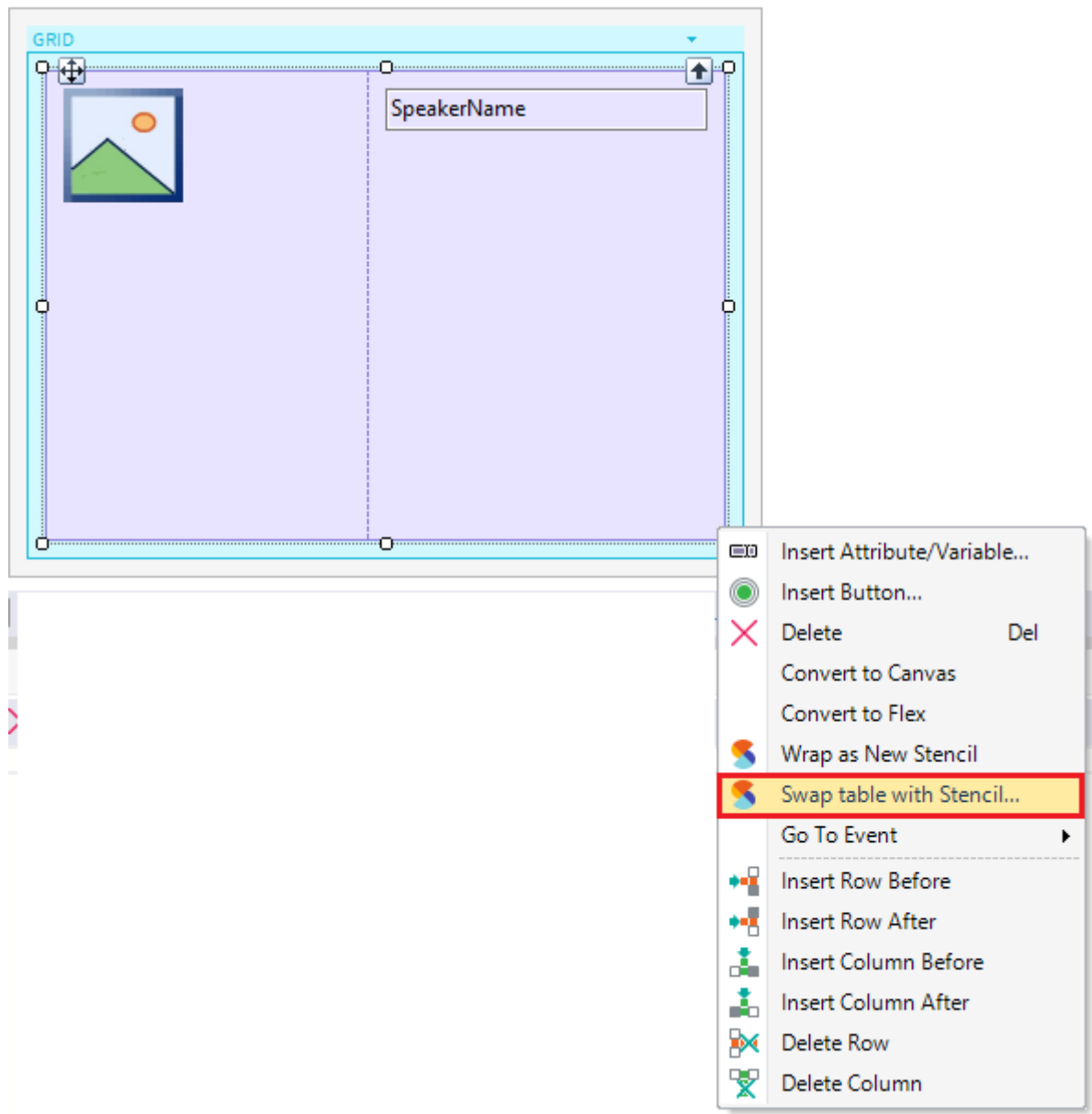
Try the *Swap table with Stencil* option and then try the Insert option. Notice the differences between using one method or the other.

After applying the Stencil, make sure that attributes are used in the grid, and then run the application by pressing F5.
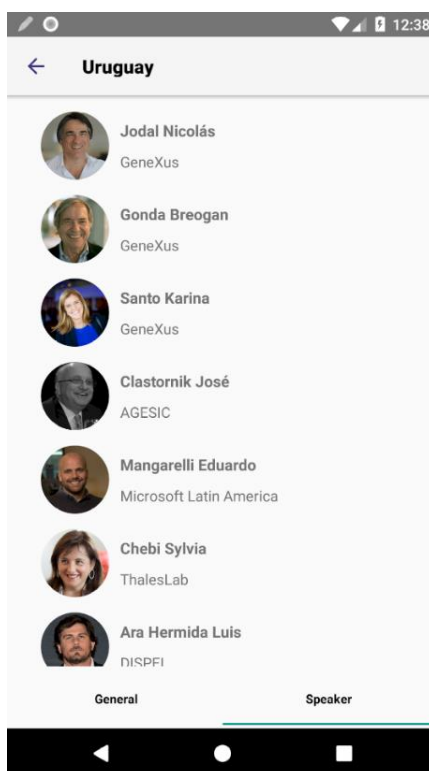
Adding new Transactions (Sessions, Tracks, Rooms, Restaurants)

- Import **3-MoreTransactions.xpz**.

    This xpz file contains, among other things:

    - TRNs: Track, Room, Session, Restaurant, SessionFavorite, User, Device

    - Data Providers (Autopopulate Data in transactions):

        o Track_DataProvider

        o Room_DataProvider

        o Session_DataProvider

        o Restaurant_DataProvider

    - Domains: Active, Status, TrackColor, Position, Floor, etc.

    - WorkWith for Web: of the imported transactions

    - WorkWith for Smart devices

        o WorkWithDevicesRestaurant

        o WorkWithDevicesSession

   - Stencils

        o SessionInfoLists

- o   TrackView

- o   SessionGeneralView

- Procedures (used in formulas)

  - o   getSessionSpeakers

  - o   ArrangeName

- Menu: EventDay

- Select Run on the **Home** web panel (you will be prompted to reorganize so that the tables corresponding to the new transactions are created). At the end of the Build process, the tables will be filled through the data providers defined for this purpose. Make sure that they have been run.
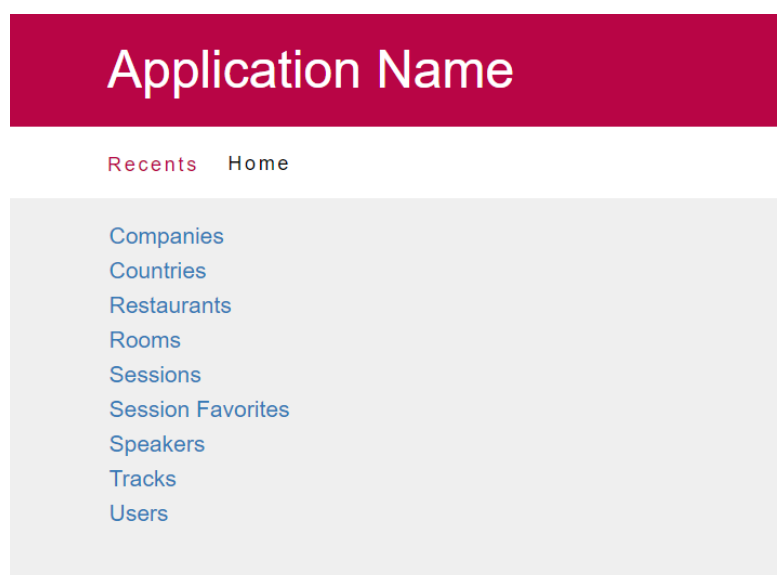
- Take some time to familiarize yourself with the structure of the **Session** transaction that stores the information about the event's presentations.

- Run the Menu (F5 if it is configured as a Startup Object), select the **Sessions** option (the Work with List will be displayed).

> Note that in each Session there is a list of Tracks and each one has a certain color.
> You can find out how this was implemented by going to the Events tab in the Tracks node.

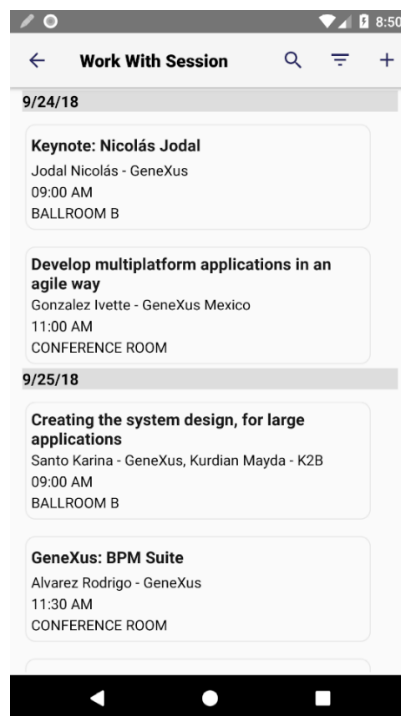- Sessions are **grouped by day**. How was it achieved?
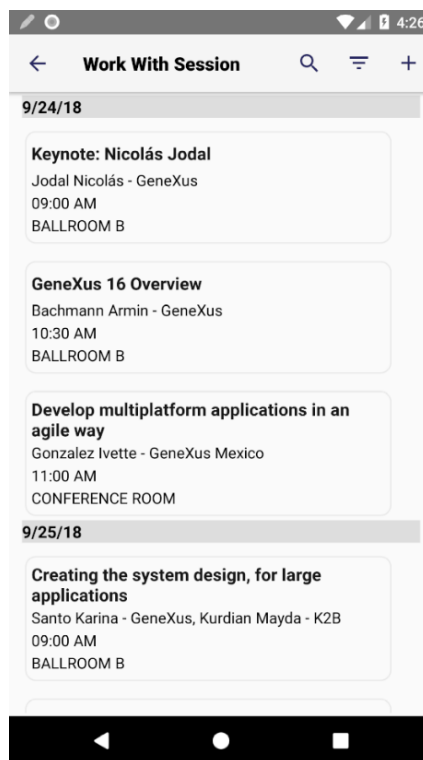
- Only active sessions should be displayed, so it is necessary to filter those talks in which the attribute SessionActive = Active. How is it implemented? To try this, mark a session as Canceled using the Web Backend and run again the Sessions List screen on the device; the selected session should not be visible.

Remember the Conditions property that is associated with grids. You may also establish Conditions at the object level, so try both options.
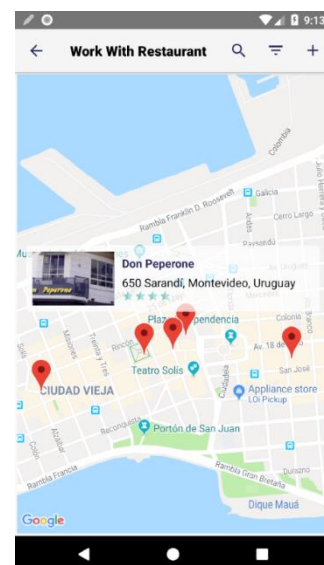
Restaurant Transaction

- Look at the **Restaurant** Transaction. The list of Restaurants should not be displayed in the standard manner; instead, the restaurants should be seen as **points on a map**, because they have a geolocation attribute. When you tap on a point, you should see its information, and when you tap back on it, you should see its detail.
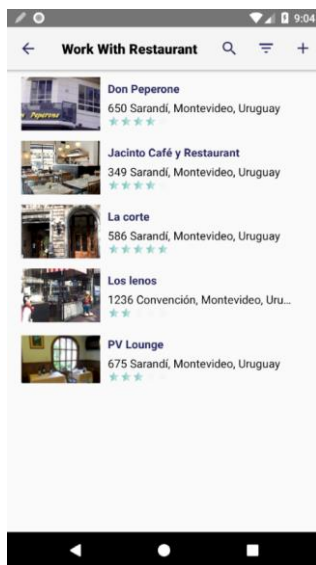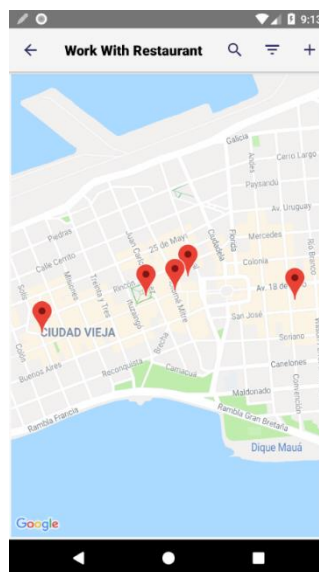
Current View   Map View

Note: For the map app to work, you must have the Google maps emulator installed on your device. In any case, you will have to configure the Android Maps API Key at the main object level (in this case, the EventDay menu). Ask the instructor for this key to try it:

- Also, note that the restaurant rating is shown with stars. How is it done?

- When editing a restaurant's information, the user will also be able to also enter its rating using stars. Implement it (with Live Editing).
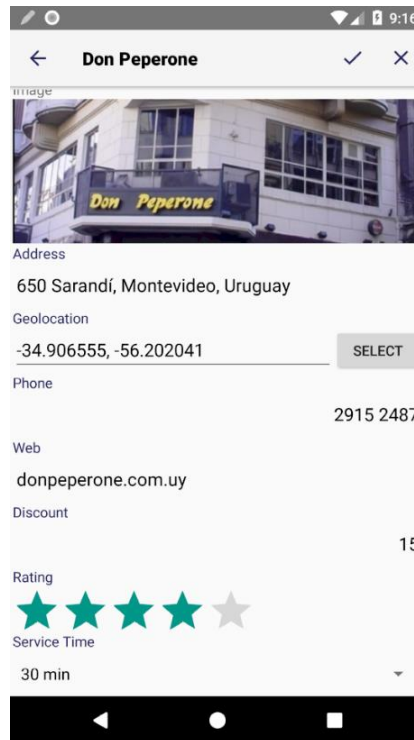
- The grid of Speakers associated with a presentation should display the speakers one by one with horizontal scroll. How do you implement it?
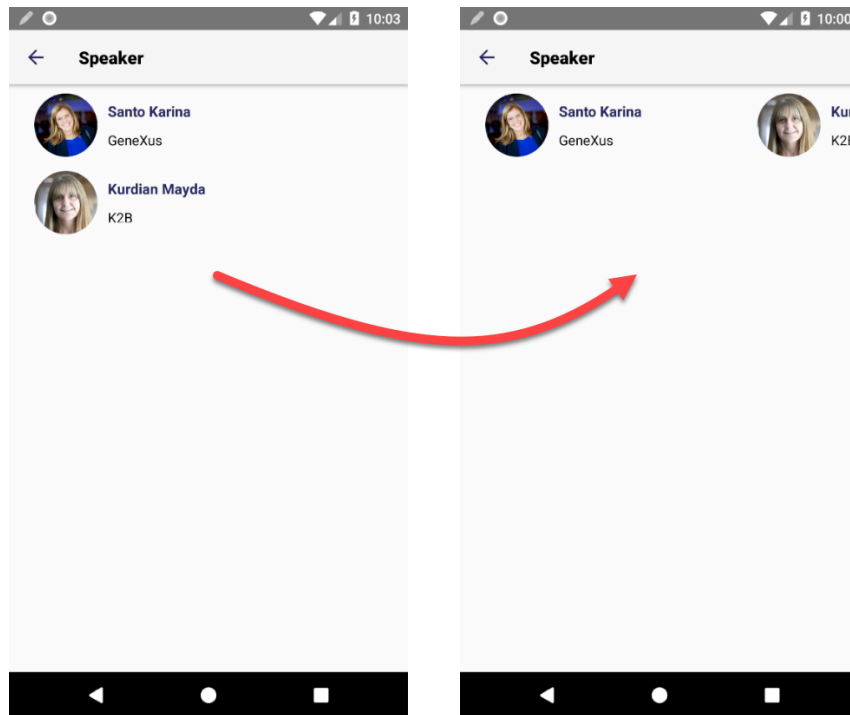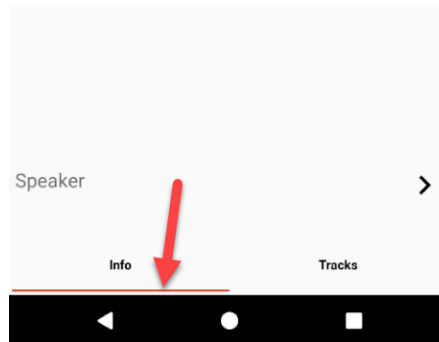
Customizing the Detail

- If, instead of showing the **Detail** info as **Tabs** (image on the left), you want to show **Inline** the first section and as **Link** the second one within the same tab page, and then in another tab page the track information, how do you do it? (Use Live Editing)
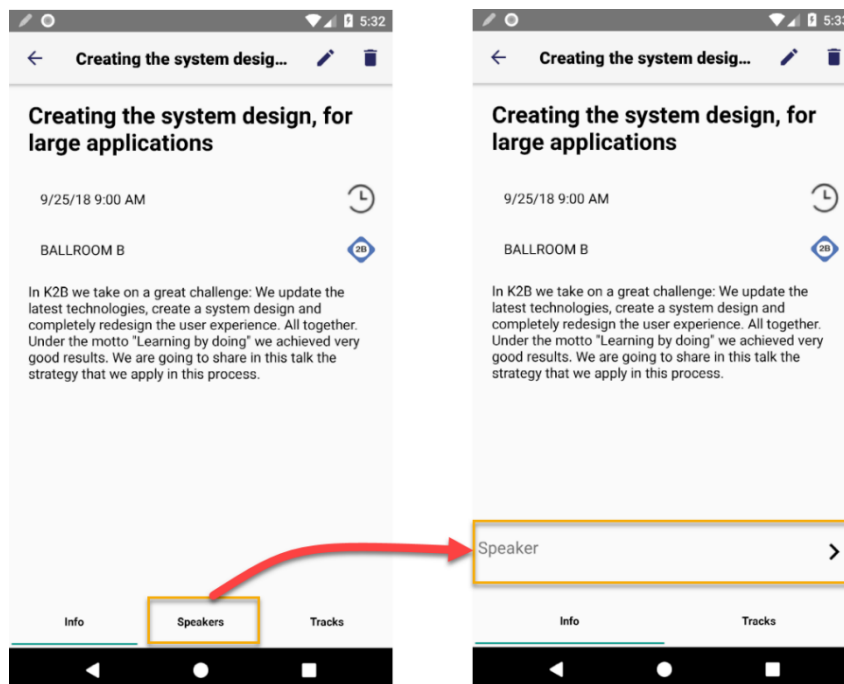
> Remember that if you delete the control <All Sections Content> from the Detail, it will reappear in the Toolbox, together with one for each individual <Section>. If you add the two individual sections, remember to keep the Rows Styles property of the table where they are inserted, and make sure to add the "pd" (Platform Default) value for the second.

Note: You can change the underscore indicating that a tab is active by selecting the Tab class of the tab control and changing its "Tab Strip Indicator Color" property (do it from the Live Inspector to try different colors until you find the desired one):

Make sure to save the changes so that they are applied the next time you compile the application (because Live Editing allows you to test how the changes will look like, but they are not applied unless you save).

Customizing the View and Edit screens

- Customize the **View** screen of the Speaker's **Detail** so that it looks at runtime as shown in the figure below (use Live Editing to see the changes in the emulator or device as you change the View):
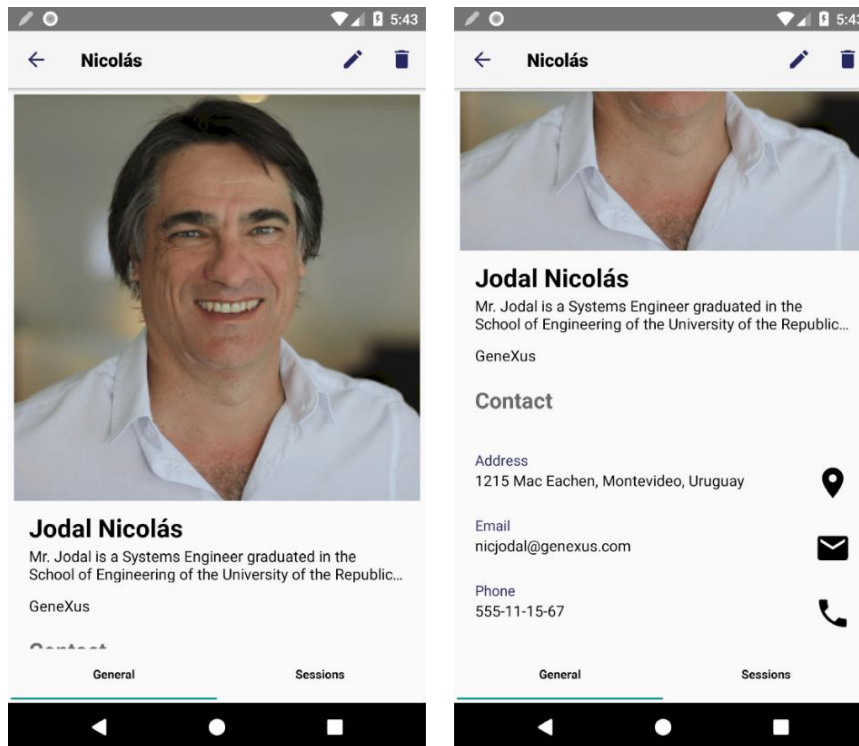
To do so:

Use the ViewImage class for the image and the property Auto Grow = True.

Assign the following classes to the attributes:

SpeakerFullName : class AttributeSpeakerNameT

SpeakerCVMini : class AttributeSpeakerBioT

CompanyName : class AttributeSpeakerCompanyT

Text Block Contact : class TextBlockTitleWidgetM

SpeakerAddress, SpeakerEmail, SpeakerPhone: Label Position Top, class AttributeSpeakerBioT

- Also, customize the Edit screen to make it look similar to the following:
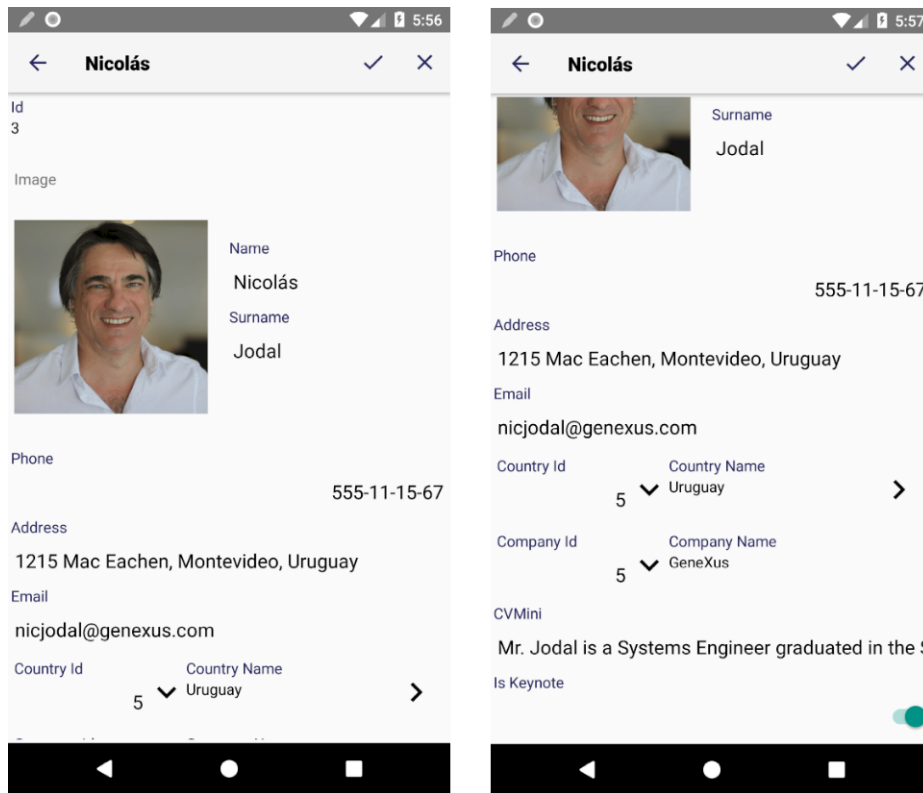
The objective is to become familiar with the different **layouts** of the same object. Try it with Live Editing. Once you obtain the desired result, save it and compile the main object.

Changing the Navigation Style

- Change the app's **Navigation style** so that when running on Android <u>phones</u> it is **Slide**.

- Run and test the changes made.

> Note that you can access the Menu by swiping your finger from the left edge of the screen, or with the hamburger menu in the Application Bar to the left:
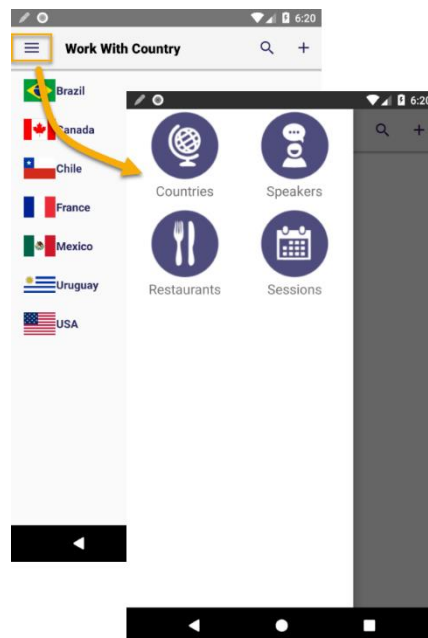

- Note that the **Countries List** is opened directly. Why does this list open and not another?

- If you want a **welcome screen** to be displayed in that case, instead of the List, how do you implement it? You will have to create a **Panel** to implement that screen. Now, at runtime, when you open the application, the screen will be displayed as shown in the figure to the left. Also, you will be able to access all the options by sliding the menu (screen on the right).
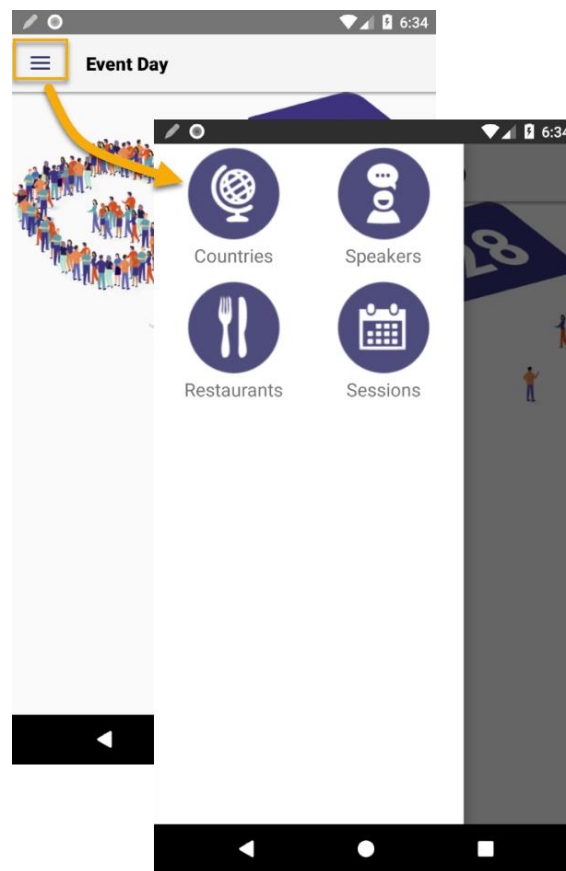
- If you have a device other than an Android Phone, try it. The welcome screen should not be opened.

- Return to the Default navigation style and note whether or not the welcome screen is executed.

Transitions

Make the Sessions List to open from below (Push up) when it is invoked from the menu and only in that case. If you haven't studied Invocations at the end of the design and behavior module yet, then implement it in static form.

> In static form: You will have to create a subclass of the Form class –for example, one called FormPushUp– and associate it with the MainTable of the Sessions List.
>
> In dynamic form: remember the CallOptions to configure an invocation. Don't forget about the "Effect" enumerated domain; also, to add more than one statement within a client event you will have to use the Composite command.

Orders and filters

- Open the List of Speakers and look at the order in which the information is being displayed. Go to GeneXus and change it to CompanyName. Try it at runtime.
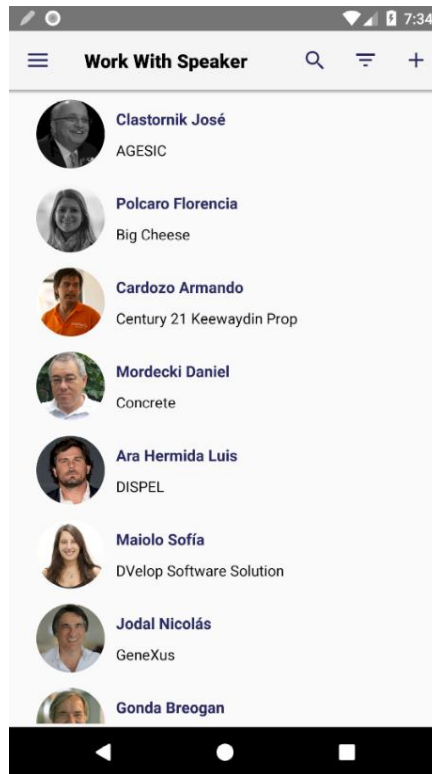
- What if now you <u>also</u> want to give the user the ability to sort by last name (SpeakerSurname)? Implement it.
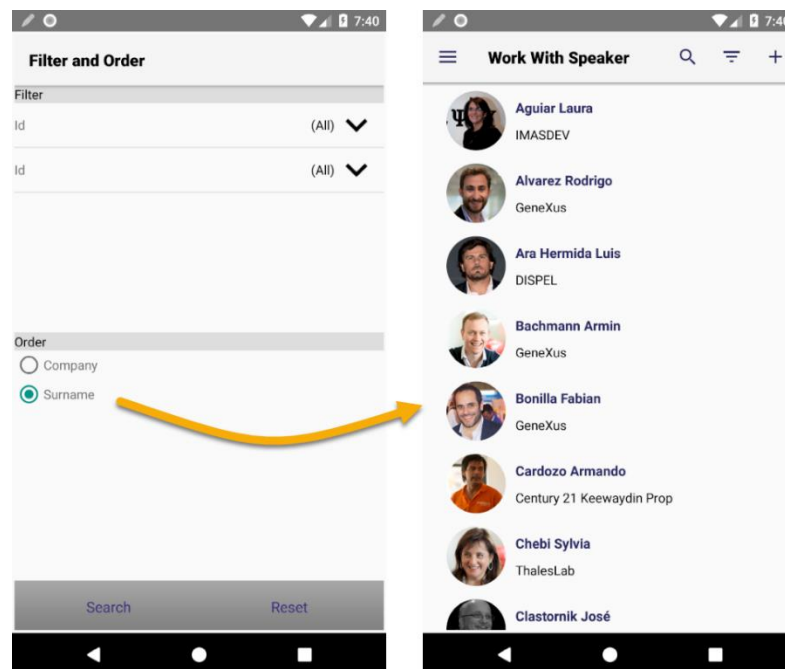
- What if you want the information to be grouped by Company (CountryName) and for each group sorted by Last Name (SpeakerSurname) as shown below?
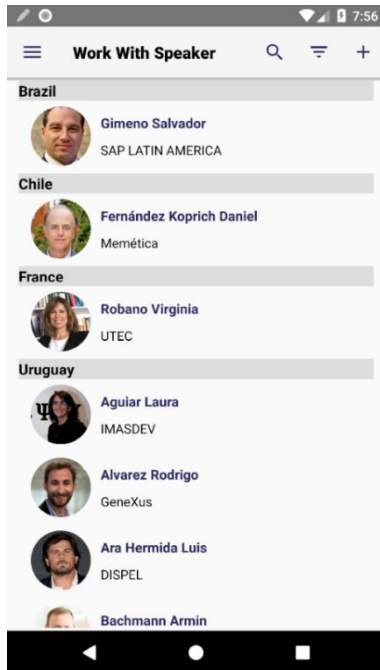

- Note the attributes by which you can search. Remove some of them (for example, CVMini) and test.

Integration with other device apps

- When viewing the detailed information of a session, you want to offer to share it, write a tweet, or enter it in the device calendar. Sharing options are required to be displayed in a Sheet type menu as in the figure below.
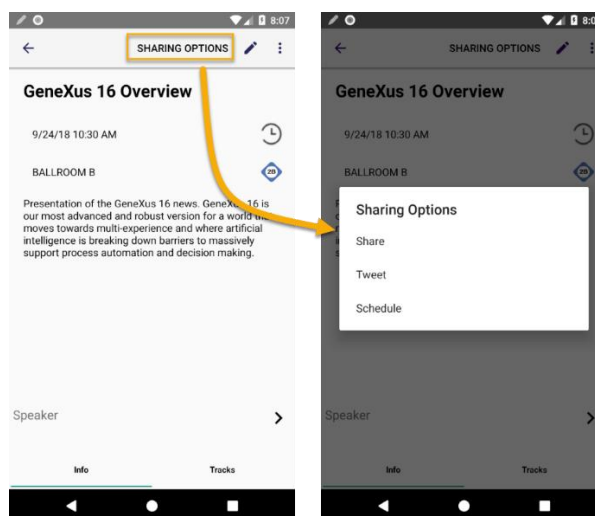
- Add the three buttons and program them.

Tip: You must add an Action Bar which must use a Sheet Control Type containing the three options. Remember that to program these options you must use the corresponding Share, Twitter and Calendar APIs.
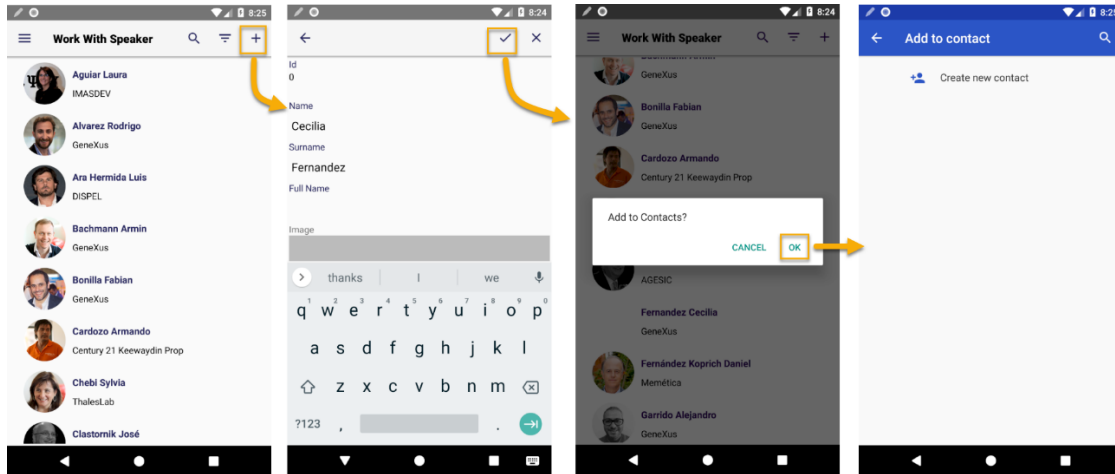
- When inserting a new Speaker into the system (through the List), it should immediately offer to add it to the device's address book. How do you implement it? You can ask the user for confirmation before attempting to insert the contact in the corresponding app.

Solution: You must use a &Speaker variable of Business Component type based on the Speaker transaction to retrieve the inserted data and then call the Contacts API.

```
Event 'Insert'
        Composite
                WorkWithDevicesSpeaker.Speaker.Detail.Insert(&Speaker)
                Confirm('Add to Contacts?')
                Contacts.AddContact(&Speaker.SpeakerName, &Speaker.SpeakerSurname,
                &Speaker.SpeakerEmail, &Speaker.SpeakerPhone, "", "","")
        Endcomposite
EndEvent
```

Call options: Popup

Now we want the Restaurants List to continue to be displayed as a standard grid, but give the user the ability to filter restaurants according to the maximum time available for lunch (note that the Restaurant transaction has the RestaurantQuickestServiceTime attribute of the enumerated domain that was imported with an xpz), and show the resulting restaurants on a map. How do you implement it? We want the screen that asks the user for that maximum time to be a Popup window.

You can create panels from scratch, or import the file 4-Restaurant_Panels.xpz

---
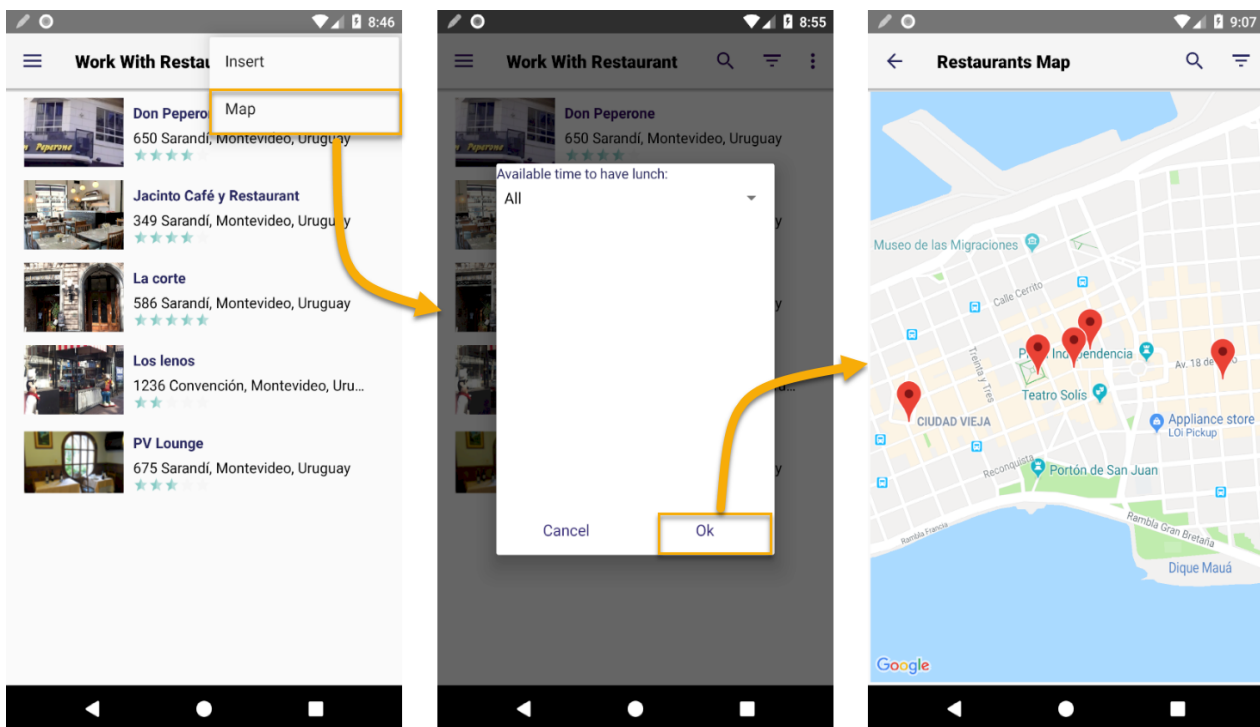
Solution: Add a Map button to the List of Restaurants and the variables &timing and &ok.

```
Event 'Map'
        Composite
                RestaurantFilter.CallOptions.Type = CallType.Popup
                RestaurantFilter.CallOptions.TargetSize = CallTargetSize.Small
                RestaurantFilter(&timing, &ok )
                If &ok = True
                        RestaurantsMap(&timing,0)
                endif
        endcomposite
Endevent
```

Also, note that conditions must be associated with the Restaurants Map grid to filter the restaurants according to the time received in a parameter.

5. ENABLING GAM

- Before continuing, **freeze** the current development version to have a backup of the current state of the KB.

- This is done from **View/Versions**, **Freeze** on the version.
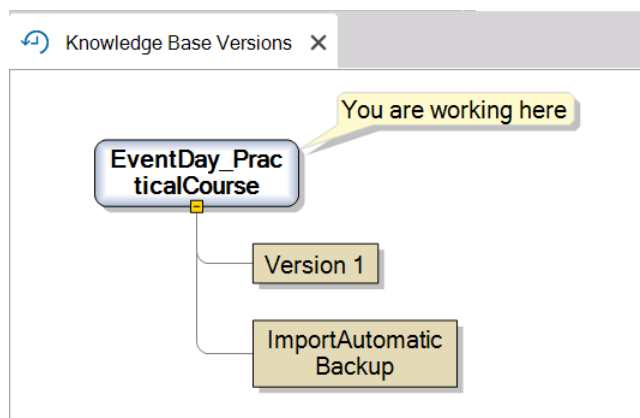
- Add the **GeneXus Access Manager (GAM)** to obtain an automatic access control.

  To do so: Configure the version-level property **Enable Integrated Security** to True, and click on Install:

Important Note: when you finish importing all the GAM objects, make sure that the property "Enable Integrated Security" is effectively set to True. If not, set it back to True and press Install again.

Notice:	
	• The version properties that are displayed after enabling integrated security.
	• The Data Stores. You will see a different one for the GAM database.
	• All the objects imported into the KB after the previous change (in the KB Explorer you will find, below the Root Module, two new modules: GAM_Examples and GAM_Library).
	• What objects are automatically configured in the "Login Object for Web" and "Login Object for SD" properties?

Enable Integrated Security	True
⌄ **Integrated Security**	
Integrated Security Level	Authentication
Application ID	1216487c-a513-4c6f-b7b7-9595e4ca6fe2
⌄ **Web specific**	
Login Object for Web	GAMExampleLogin
Not Authorized Object for Web	GAMExampleNotAuthorized
⌄ **SmartDevices specific**	
Login Object for SD	GAMSDLogin
Not Authorized Object for SD	GAMSDNotAuthorized
Change Password Object for SD	GAMSDChangePassword

- Select **Rebuild All** (it will take some time) and press **F5**. Confirm that you want to create the GAM database:
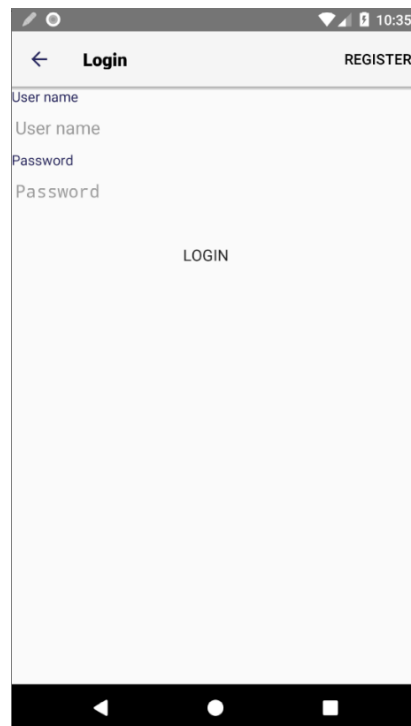
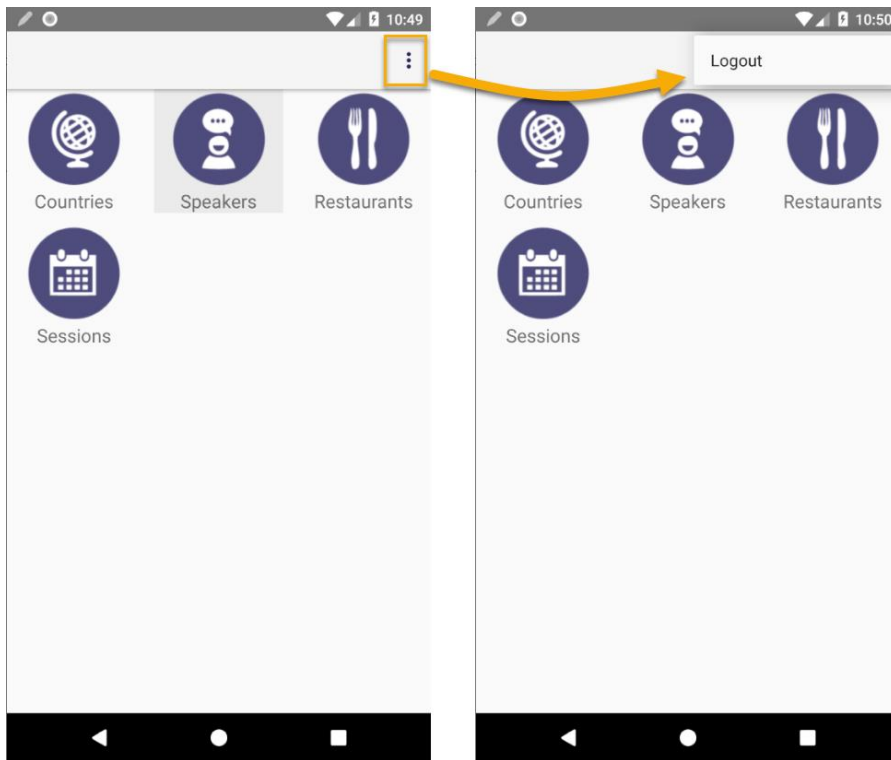| Confirm	
❓	GAM database not found. Would you like to create it?
	Yes No

- Run the application in the Smart Device or emulator. You will see that the login screen is now displayed as the home screen. Enter the following login details:

  User: **admin**

  Password: **admin123**

- Note the Logout button that will have been added to the Menu:
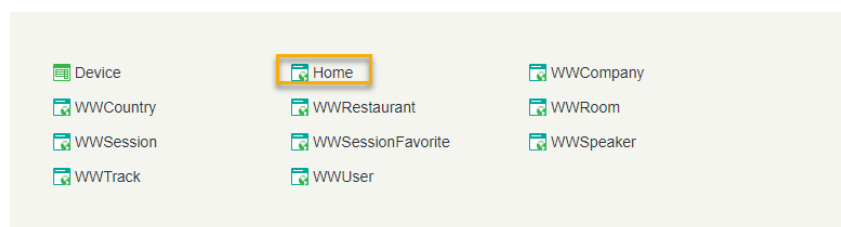
- Run the Developer Menu (web) — Menu Build option Set Developer Menu as Startup, or Menu View option Show QR Codes—, and try to access an object (for example, the Home object). You won't be able to do it before **Login**. Give it a try.

- Enter with the following details:

user: **admin** password: **admin123**

- From the Developer Menu, execute the **GAMHome** Web Panel and look at the Backend to manage **Security**:

- Create a new user from the **Users** option and use it to access the application in the emulator (you will have to Log out to be able to enter with another username).

Assign it the administrator role.

Configuring Security at the Object Level

From the mobile app, it should be possible to work with the Speakers of the event. But it should only be available for authorized users. For the rest of the app, we don't want any security features. Implement it.
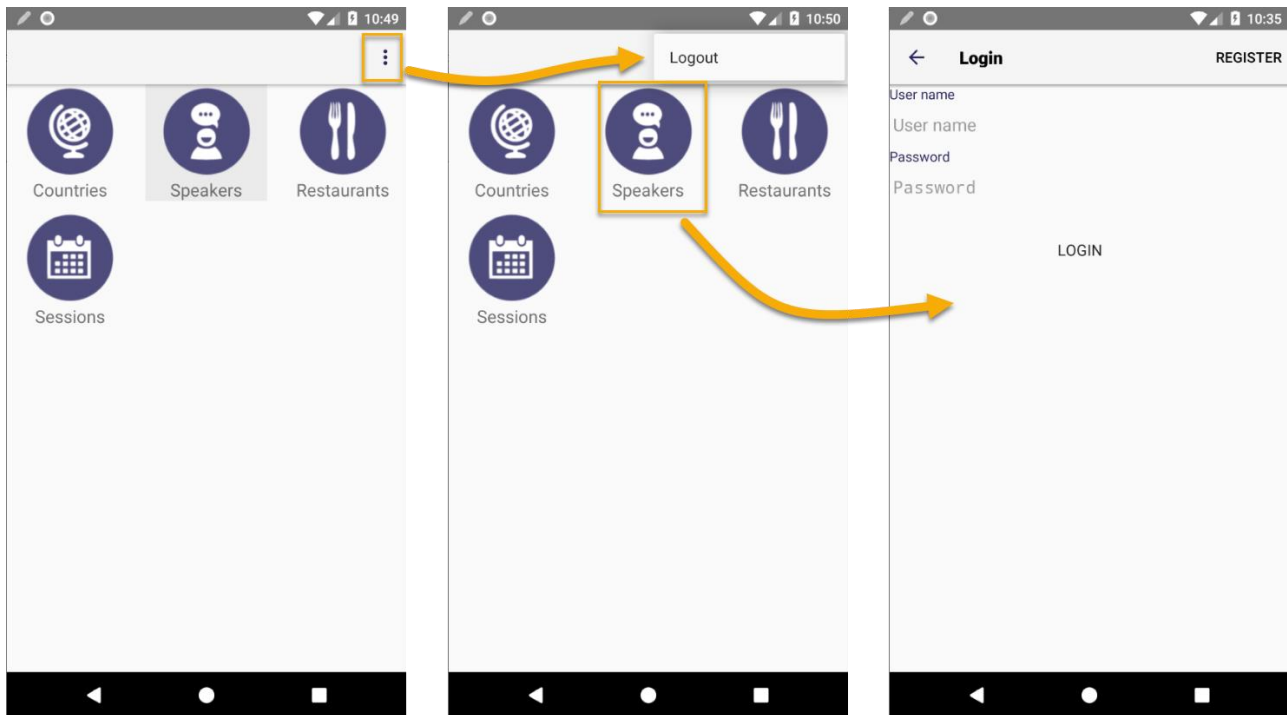
To do so:

- Configure the property **Integrated Security Level = None** at the Version level:

Integrated Security	
Integrated Security Level	**None**
Application ID	**1216487c-a513-4c6f-b7b7-9595e4ca6fe2**

- In the object WorkWithDevicesSpeaker configure the property **Integrated Security Level** with the value **Authentication**.

> With this configuration, you don't need to log in to access the Event options, EXCEPT to access the Speakers option, where the **login** screen will appear.

- Run a **Rebuild All**, configure the EventDay object as StartUp Object again and run the app to check the changes made.

Note: When accessing the Menu, first select **Menu/Logout** to log out the user previously used to log in. If the Slide navigation style is used, change it to Default and try it.

6. CONVERTING THE APPLICATION TO OFFLINE

- We want the entire application, except for the part that requires authentication (in our case, access to the event's Speakers), to be able to run offline. Implement it.

To do so:

Configure in the **Menu** the property **Connectivity Support = Offline**

> Remember that the GAM login can only be performed from an Online object.

- Execute a **Rebuild All** and check that the **Offline Database** object is created, indicating in the **Smart Devices Impact Analysis** the tables that will be created in the device. Which tables are not part of the Offline database? Why?

- Configure the data **Synchronization** mechanism (Send and Receive properties) you want. That is, when you want to send the data (in our case, the data that may have been entered/updated/deleted in the device through the Work With elements that have Edit modes available) and when and how to receive data from the server when you are connected.

- Run the application (remember that you must necessarily run it compiled. You will not be able to use the KBN when running on iOS) and note that the data is loaded into the **local database** of the device.
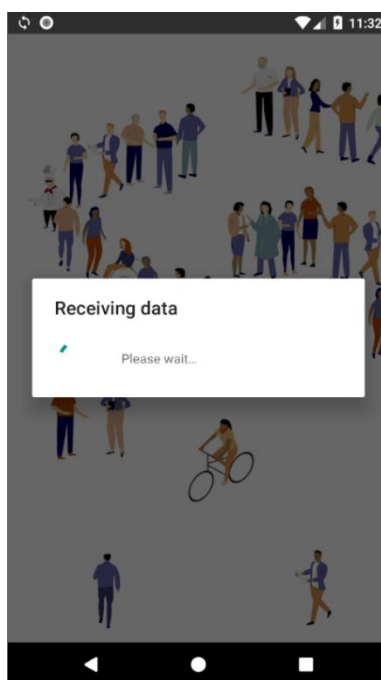
- **Disconnect** the device and check that the application works correctly. Enter a new record, for example, a new country.

> To simulate the device disconnection use Airplane Mode.

- Try to enter the **Speakers** option, and verify that the **login** only works if the device is **connected**.
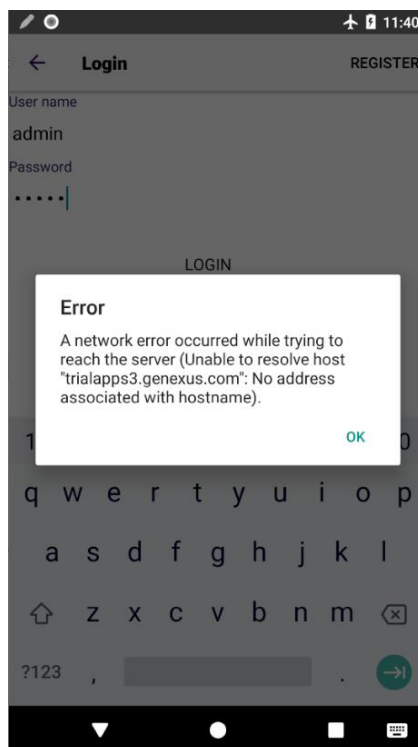
- Enter a new country through the web Developer Menu. Note that the country that you entered earlier from the device is not in the centralized DB and that, at the same time, the country that you've just entered into the centralized DB via the web app is not shown in the Country List of the device.

- Connect the device to the Internet and note that the data is synchronized, according to the criteria you defined for send and receive operations (for the receive operation, did you leave "On launch application" and after 600 seconds? What will you have to do to receive in the device the changes made through the Developer Menu?)

- Try changing the property "Data Receive Criteria" to "After Elapsed Time", enter a country through the Developer Menu, immediately refresh the List of countries in your device, wait for the number of seconds you have indicated on the property "Minimum Time Between Receives" and refresh again.