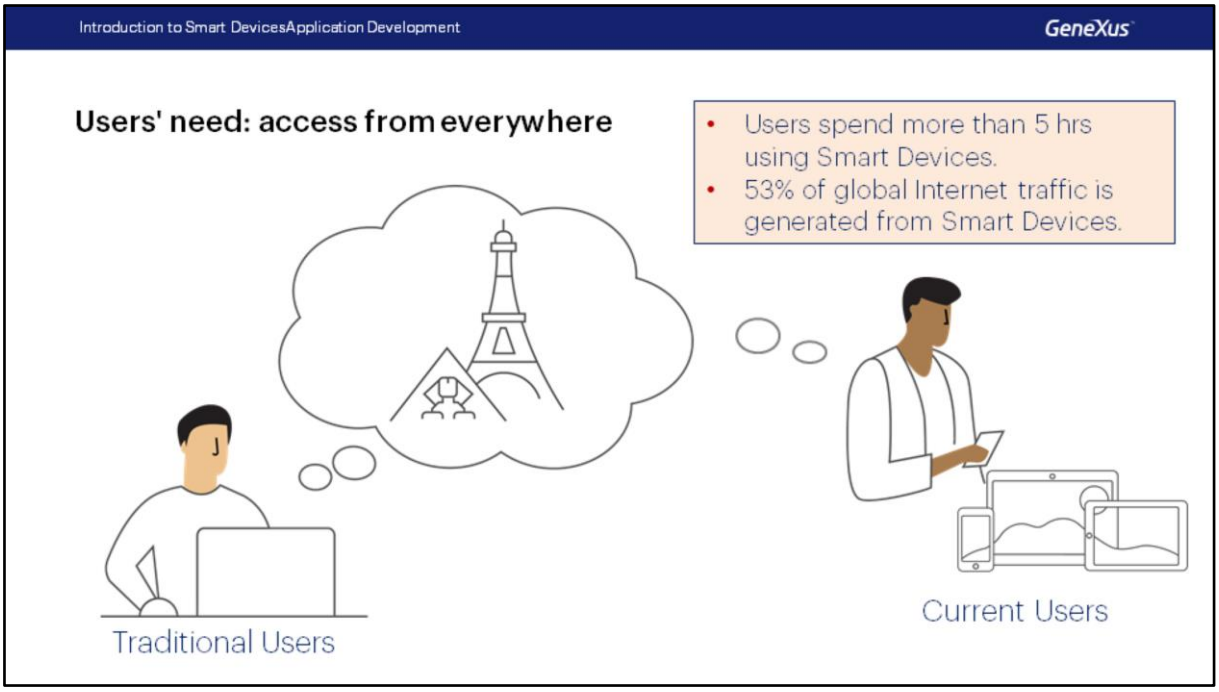

GeneXus™
The power of doing.

Introduction to Smart Device Application Development

Development

GeneXus 16



Until a few years ago, users used our applications from a PC, but today's users demand to be able to perform their tasks with the device they have at hand. At the office, they will still use a PC, but while walking, or on the commute to work they will use smaller devices such as cell phones or tablets, or even smart watches. Whey they are at home they can even use a smart TV.
All these devices will provide the ability to reach users in a way that was previously impossible. According to publications, it is estimated that users spend more than 5 hours using their mobile devices, and more than 53% of global Internet traffic is currently generated from these devices.

But... why do we need an application for Smart Devices if we already have a web app that users can access from their devices because it is also Responsive?
In this video, we'll try to answer this question. Also, we'll see the most important features of the apps we can develop with GeneXus, and at the end we'll see an application and how it was developed.

Responsive Web Applications

Pros

- Available for all users
- Easy to update and mantain
- Lower development costs

Cons

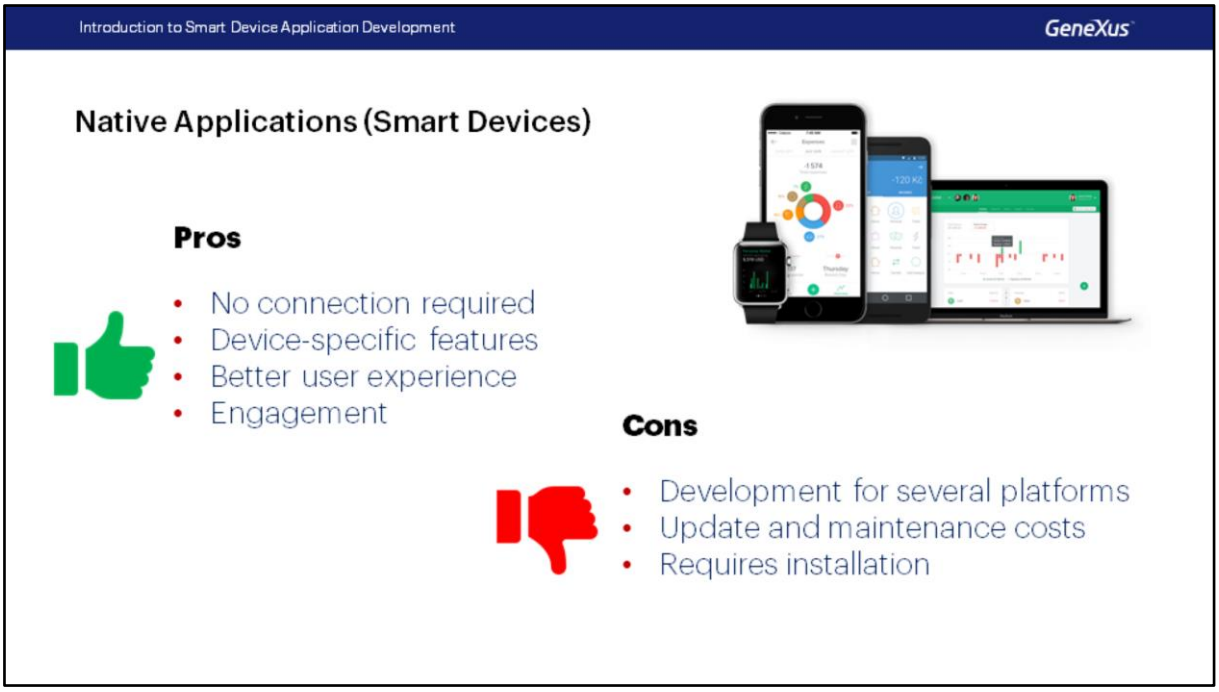
- Must always be connected
- Lack of natural navigation
- No device functionalities
- Less engagement

First, let's see Responsive Web applications.
The advantages of this type of applications are that:

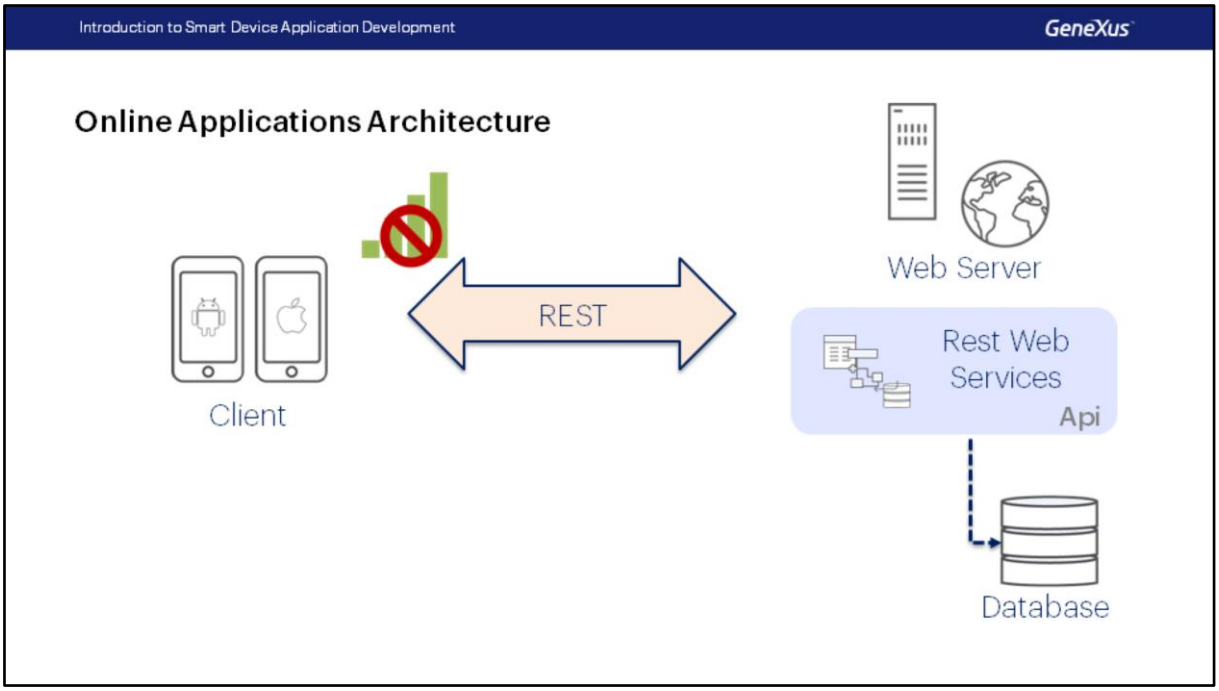
- They are available for all users,
- They are easy to maintain, especially with GeneXus,
- And have lower development costs.

On the other hand, their disadvantages are as follows:

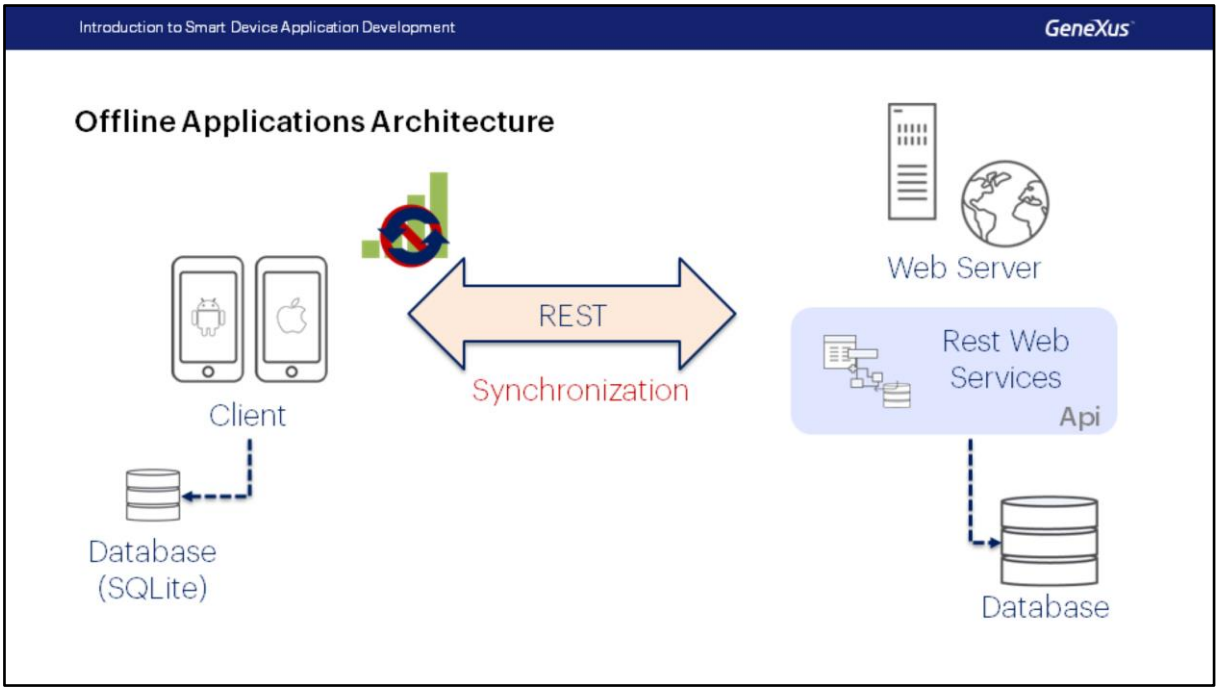
- They can only work online; they always need an Internet connection,
- The navigation method is not natural; it isn't native to the device, because the browser navigation is used,
- They won't provide access to certain device features such as the camera, GPS,
- Lastly, it can be said that they generate less user engagement with the app, because it isn't an app installed on the device.



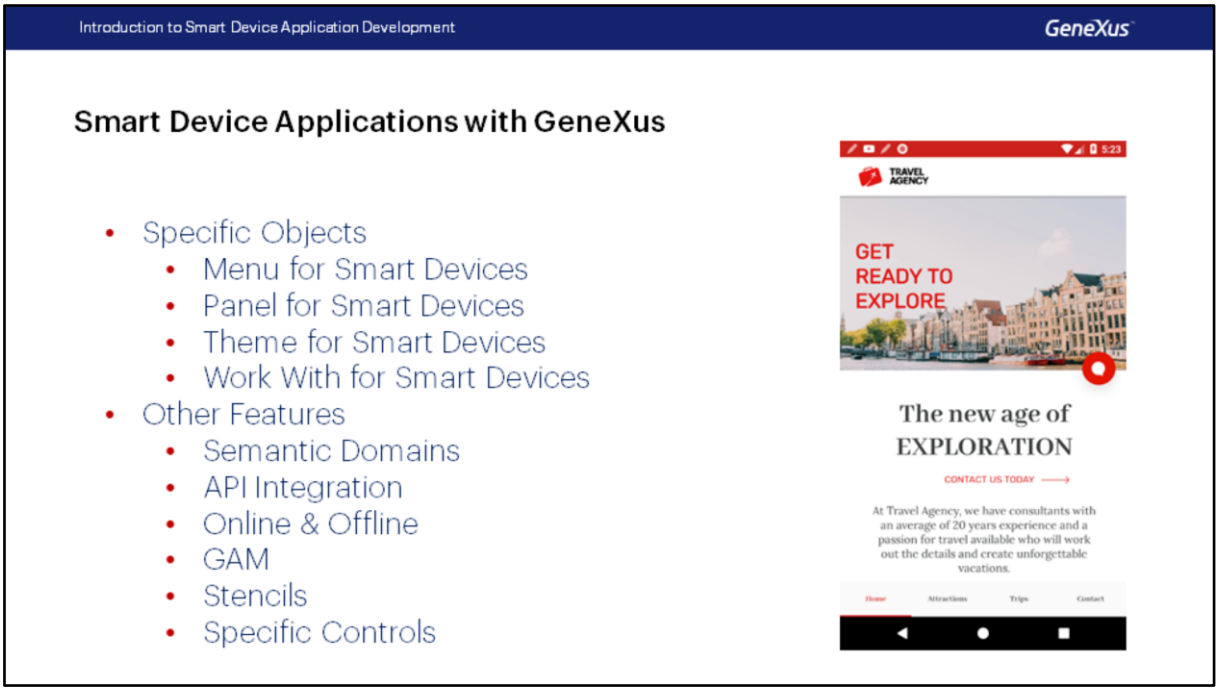
About native apps for Smart Devices:
The advantages are that a connection is not always required; we can have offline applications (partially connected or totally disconnected), we can use hardware features such as camera, GPS, fingerprint reader, everything in a natural way. This will offer a better user experience. In addition, since these applications must be installed on the device, they generate greater user engagement.
On the downside, we need to develop for multiple platforms (Android, iOS), something that is solved by GeneXus. They have higher update and maintenance costs and, as we've mentioned, they must be installed.



This diagram shows the architecture of online apps developed with GeneXus.
On one hand, we will have a native language application that will be installed on devices, either Android or iOS and will be the clients in this architecture.
On the web server side, there will be a layer of web services that use the Rest protocol. These services or APIs that the clients will use are made up by Data Providers, Processes and Business Components, which implement the entire data access required by the application.
As it is an Online application, when there is no connection we will not be able to use the application because it won't be possible to access the server to execute the Rest services that provide the data to the application.

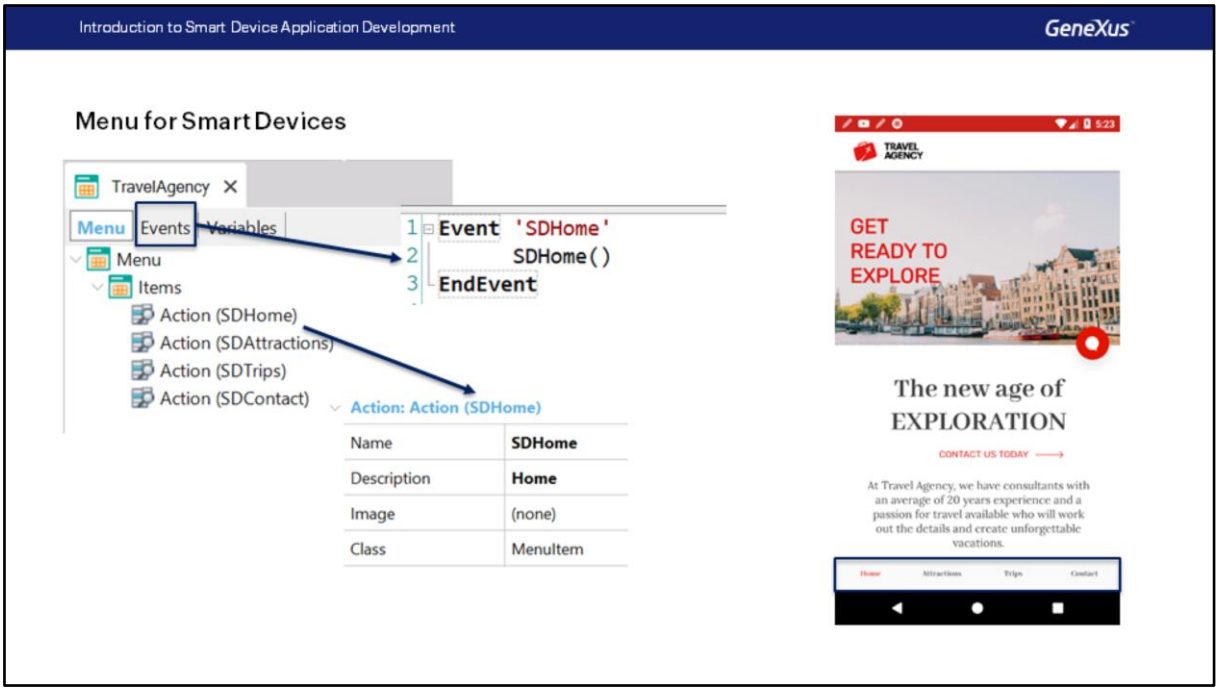


On the other hand, in offline applications, on the client side in addition to the application we will have a SQLite database with all the tables that the application needs.
The entire data access logic will be in the application itself and all the operations performed on them will be done in the client on this local database.
In the web server we will have access to the centralized database, and in the Rest services layer we will have the necessary synchronization services to provide the client with the updated data whenever it requires them. In addition, there will be Business Components that will be used for the client to inform us of the changes made locally.
In this scheme, when there is no connection the application can continue working normally, but it won't have the latest version of the data, which will be automatically synchronized when the application detects that the connection was re-established.

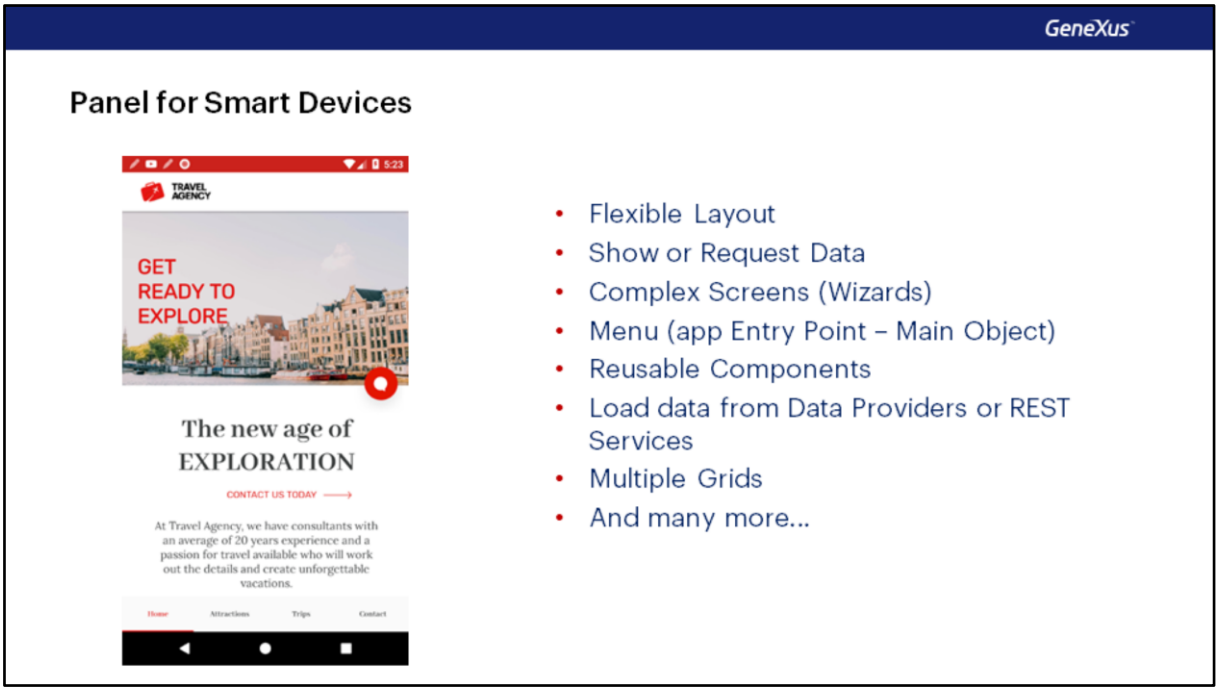


GeneXus allows developing applications for Smart Devices, almost in the same way we develop Web applications:

To this end, we have specific objects for Smart Devices which we will see in detail below, as well as other features such as Semantic Domains, integration via APIs with the device hardware, the possibility of generating connected or disconnected applications, security through GAM, use of Stencils and a wide range of specific Controls for Smart Devices.

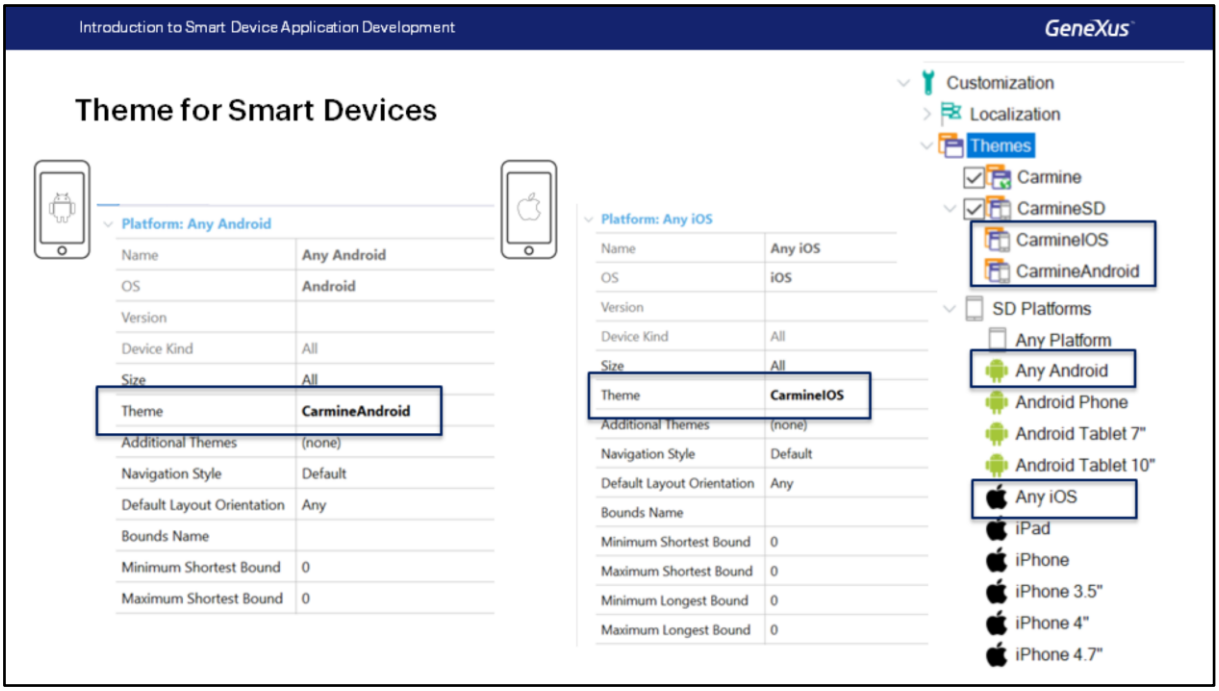


Let's see now the specific objects for the development of applications for Smart Devices.
The Menu for Smart Devices objects allow us to define menus for the applications which can be displayed in different ways, such as Tabs, a Table or a List.
To the right we can see the structure of the object, where we can define the items that this menu will have –each item has a name, a description and we can associate an icon that is an image from the Knowledge Base, and object of Image type. In the events it is then determined what happens when we access each specific item, what object will be called.
Normally, these objects will be the entry points to the application, so by default they will be Main objects.
We'll see the menu at runtime. In this case, we used the Tabs control type as you can see at the bottom.



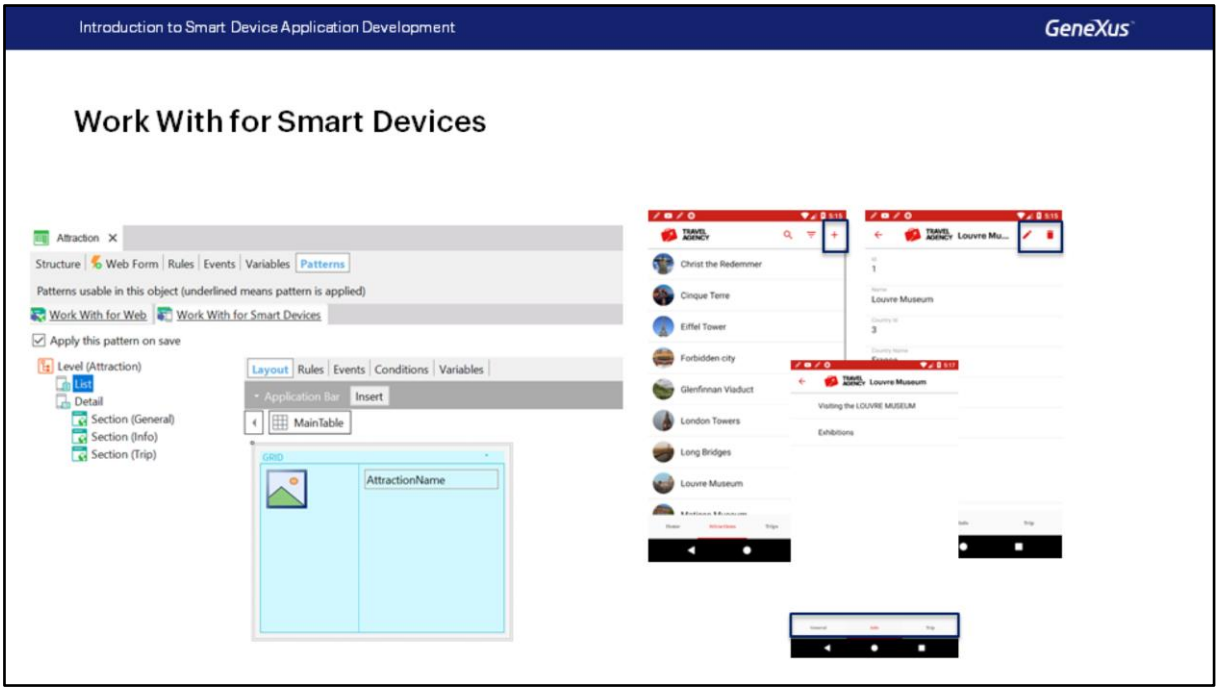
Panel for Smart Devices objects are similar to the Web Panels we use in Web development. They have an abstract Layout in which we'll insert the necessary controls such as tables, text blocks, grids, images, etc.
This panel we see here was built with this layout in a Panel for Smart Devices object.
Its main features are as follows:

- Its layout is completely flexible. When creating an object of this type, its implementation will be left to the developer.
  In this way, we can use it to:
- show information to the user, or for the user to enter information, or even a combination of both options at the same time.
- We can create complex screens, for example, to create an Assistant with several steps.
- We may use it to create a Menu, as it can also be a Main object and therefore an Entry Point for our application.
- We can use it as a reusable component similar to the way we use Web Components.
- We can use it to show information returned by Data Providers or REST services.
- We can implement multiple Grids.
- And many other uses in addition to those mentioned here.



Our apps for Smart Devices will be multi-platform. That is to say, we create basically the same app and do the minimum work required to customize it so that it runs on Android and iOS devices, following their design and behavior guidelines.

To customize the Look & Feel in each Knowledge Base there will be two themes already created, CarmineIOS and CarmineAndroid, and by default these themes are associated with the Android and iOS platform.

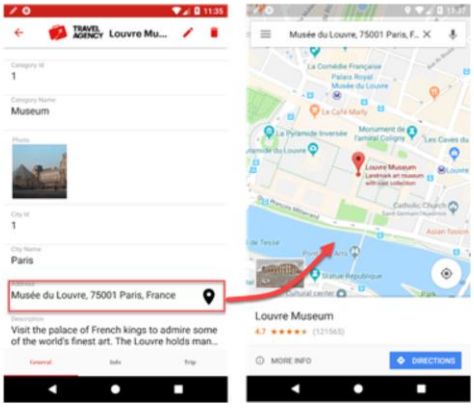


The Work With for Smart Devices pattern is a type of object specifically used for smart device application development; it implements all the features required to manage the data of an entity, including:

- Accessing a list,
- Accessing an item of that list to see its detail,
- Adding new records and editing them.
- Also, it is capable of generating additional screens and links, analyzing related data such as sublevels, foreign keys and relationships with other subordinated tables.

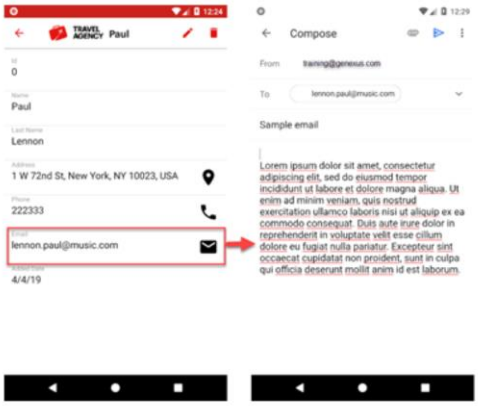
Next, we'll see a demo and use this pattern for you to see how it simplifies development tasks.

Semantic Domains: Address



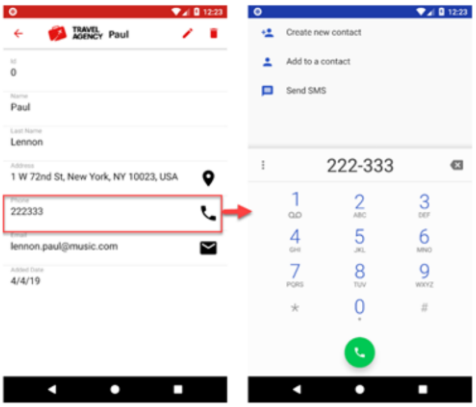
To take advantage of the integration with the device hardware, we have semantic domains: For example, the Address domain is used to store addresses at runtime. Tapping on it takes you to the default map application installed on the device and locates that address.

Semantic Domains: Email



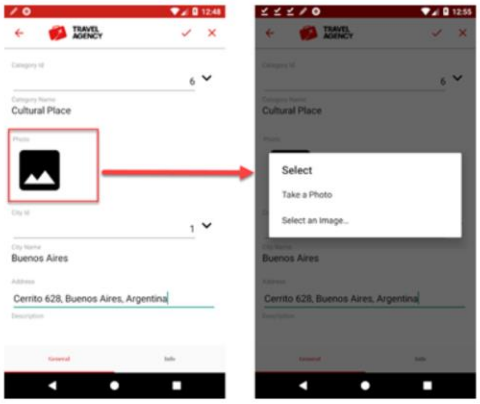
The Email domain is similar but is used to store email addresses. Also, tapping on it will open the default application to send email messages.

Semantic Domains: Phone



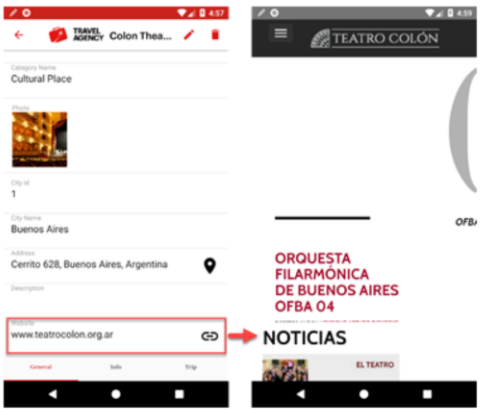
The Phone domain is used to store phone numbers and tapping on it will open the default phone call application.

Semantic Domains: Image



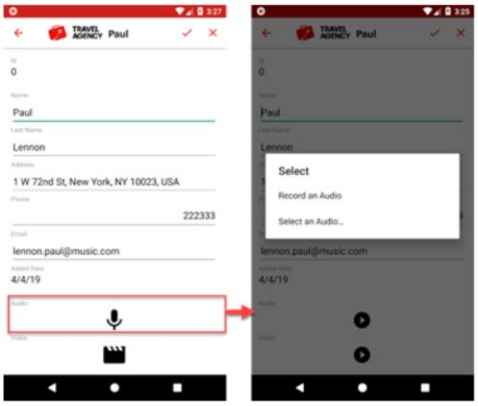
The Image domain is used to store images. In Edit mode, the user will be asked if he/she wants to take a photo with the device camera, or if he wants to select an image from the gallery.

Semantic Domains: URL



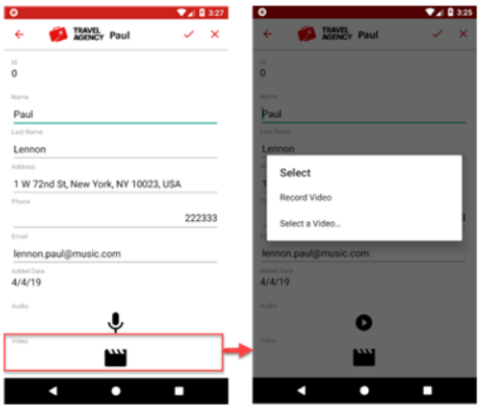
The URL domain is used to represent Web addresses. Just like the previous ones, Tapping on them will open the default browser.

Semantic Domains



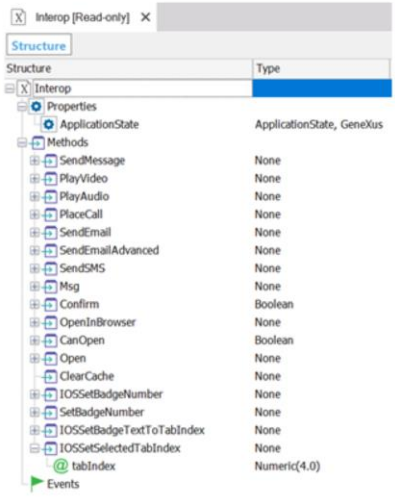
The Audio domain allows you to manage content in audio files or allows you to record an audio using the device microphone.

Semantic Domains



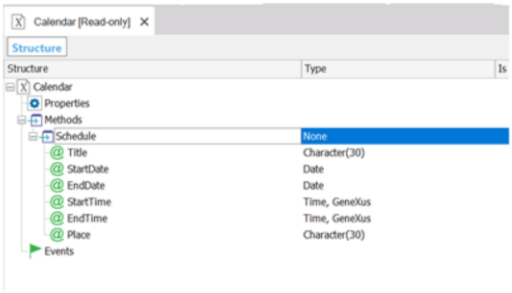
The Video domain is similar to Image but for video files. We can record a video with the camera or select one from the gallery.

API Integration: Interop



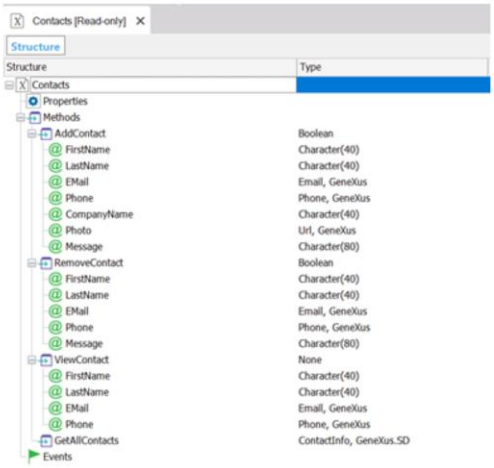
We can also integrate through APIs; let's see some of the most used ones.
Interop: this API offers an entire series of Methods to interact with common functions like SendMessage to send a text message; PlayVideo or PlayAudio to play multimedia files; PlaceCall to make a call; SendEmail to send an email message; Msg to show a message to the user; Confirm to ask the user for confirmation before executing an action, etc.

API Integration: Calendar



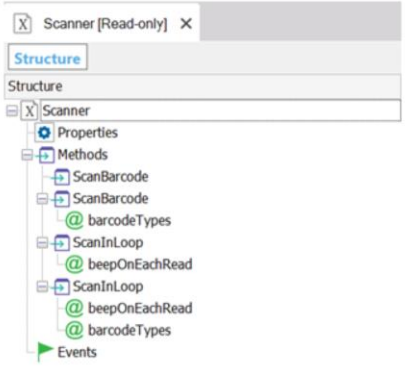
Calendar: allows generating an event in the device's default calendar application.

API Integration: Contacts



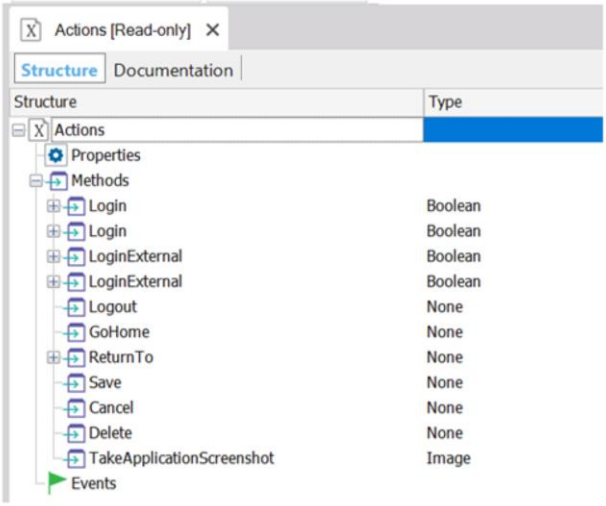
Contacts: it allows us to interact with the contacts application of the device; you can add contacts, delete them, see the information of a contact or get the entire list.

API Integration: Scanner



Scanner: allows you to use the camera of the device to scan barcodes or QR codes. Also, it has methods to scan a single code; in that case, it returns the value in a variable of Char type or in a loop and at the end returns a list of all the codes scanned in a collection based on the SDT ScannedBarcodes.

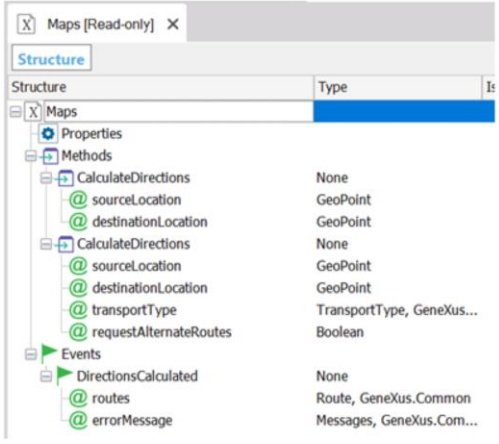
API Integration: SD Actions



SD Actions provides standard methods to:

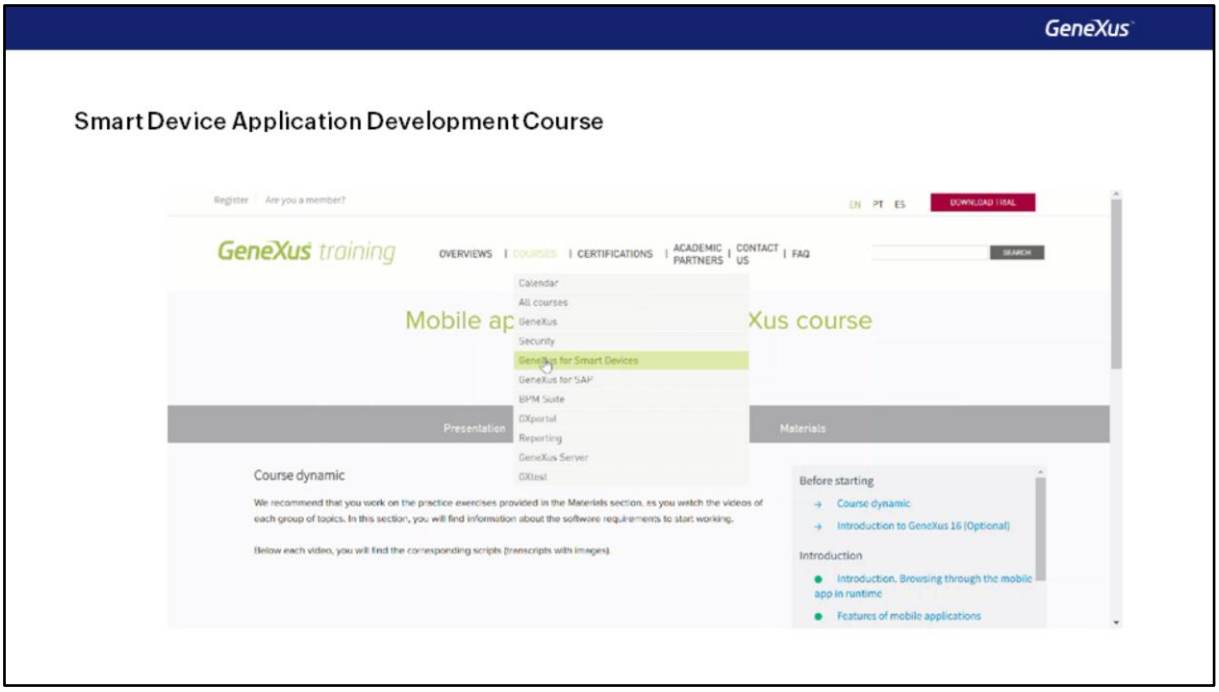
- Call the Login and log out from the app.
- Go to Home, which returns to the initial object.
- ReturnTo allows you to return to a specific object.

API Integration: Maps



Maps API: it can be used in Web and SD apps. It allows you to calculate paths and get the routes between two points.
There are many more APIs to use. You can find more information on the wiki or in the Course on Developing Mobile Applications with GeneXus.

Let's open GeneXus to see some of the things we've been talking about.



As you may have noticed, Smart Device application development is a very broad topic and covers many other issues that in this video and the demo we've seen in a simplified way, or just mentioned.
I invite you to continue studying in greater depth all these topics and many more in the course on Mobile Application Development with GeneXus from the training website.

GeneXus™

The power of doing.

Videos	training.genexus.com
Documentation	wiki.genexus.com
Certifications	training.genexus.com/certifications