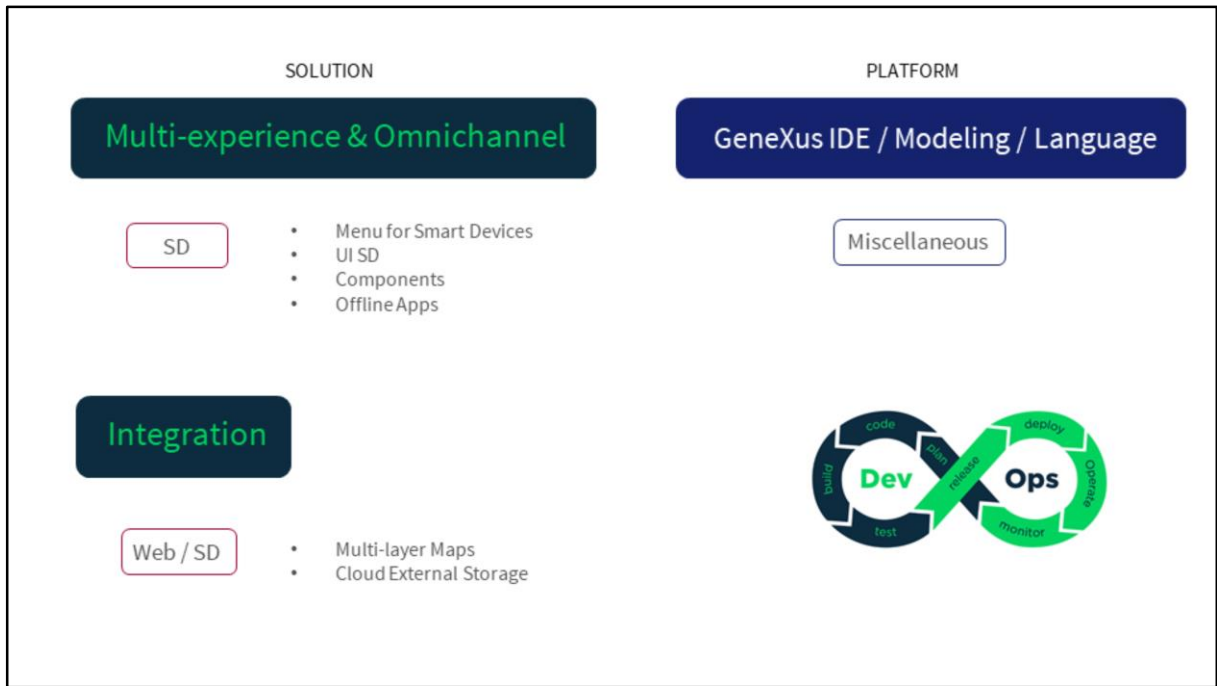


GeneXus™
The power of doing

Web/SD
Miscellaneous

GeneXus™ 16



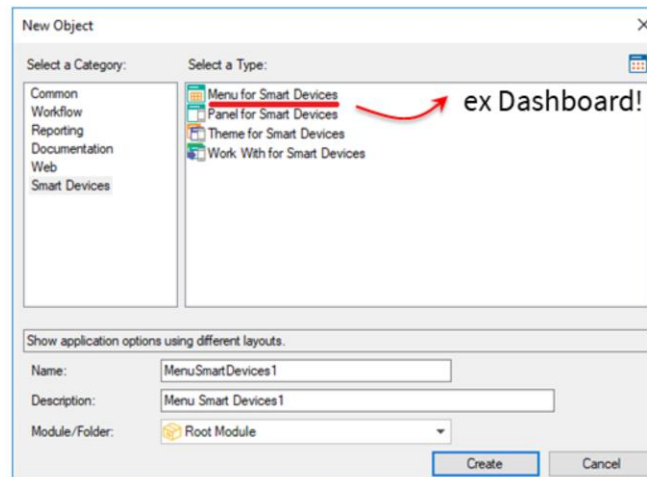
En esta sección veremos más funcionalidades que se han ido incorporando a partir de los diferentes upgrades de GeneXus 15 hasta llegar a la versión 16.

Estudiaremos lo nuevo en lo que podemos hacer con los mapas, y también algunas mejoras en lo que hace al almacenamiento externo de archivos multimedia en las nubes soportadas.

También veremos algunos aspectos relacionados con la plataforma GeneXus.

Menu for Smart Devices

Menu for Smart Devices

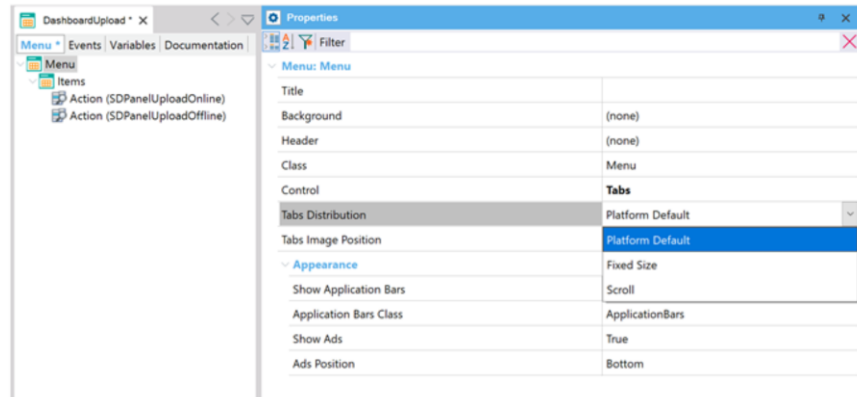


GeneXus

El objeto que llamábamos Dashboard cambia su nombre por Menu for Smart Devices.

Menu for Smart Devices / Tabs Control

- Tabs Distributions property



GeneXus

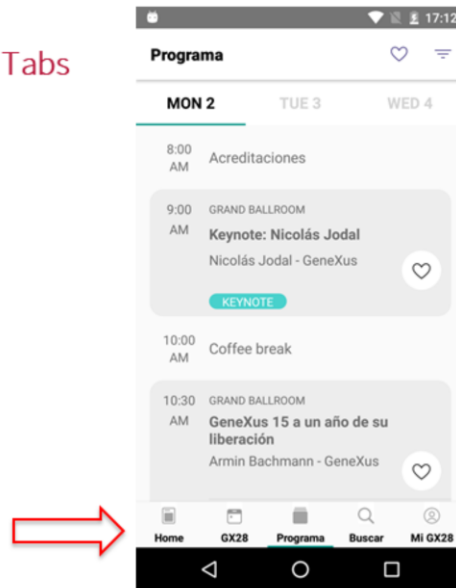
Se agrega una nueva propiedad a nivel del Menú para indicar la distribución de los ítems, cuando se muestran como Tab Control (Control = Tabs).

De acuerdo a las guías de diseño el comportamiento es Fixed para iOS y Scroll en el caso de Android si no hay espacio suficiente.

Si hay menos de 5 opciones, entonces la opción Fixed se puede utilizar para distribuir las opciones en forma uniforme.

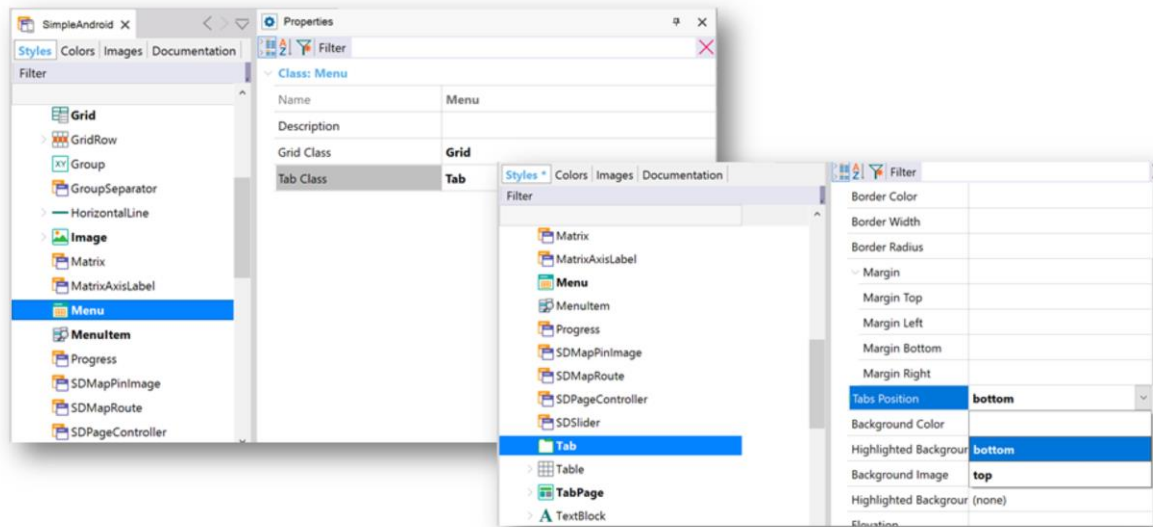
Menu for Smart Devices / Control Tabs

- Control Tabs for Android
 - Top / Bottom



En Android, a diferencia de iOS, un menú como Tab Control se puede mostrar tanto arriba como abajo. Estas opciones se configuran desde la class asociada al Menú.

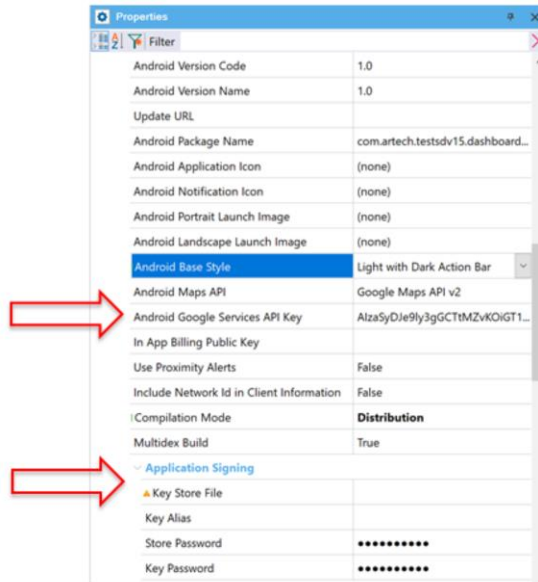
Menu for Smart Devices / Control Tabs / Android



En la class Menu se puede configurar la propiedad Tab Class y en ésta a su vez la Tabs Position.

Properties on Main Objects

Also, some properties at Smart Devices generator level have been moved or copied to the main object properties level.



GeneXus

Además, y esto aplica a cualquier MAIN, se reestructuraron las propiedades en el objeto Main:

- Se movieron propiedades que antes estaban a nivel del generador, por ejemplo:
 - Android Maps API
 - Las del grupo Application Signing.
 - A su vez, estas propiedades ahora aparecen únicamente si la app se genera en modo Distribution (Compilation Mode)
 - El beneficio aquí es poder tener más de un Main en la misma App con diferentes propiedades.
- Se actualizaron los valores predeterminados de otras, por ejemplo la Android Base Style. El default hasta la v15u5 era Dark y ahora Light with Dark Action Bar.
 - Esto para acompañar el nuevo Theme (Camine SD incluido a partir de la versión 15 u6)

Animations

Un recurso interesante de UI que se puede incorporar a partir de la versión 15 U8, son las animaciones. Específicamente las animaciones Lottie, que es una biblioteca creada por Airbnb, que provee una api JSON para integrar animaciones en nuestras aplicaciones.

Muchas herramientas de diseño ofrecen la posibilidad de exportar archivos JSON en formato Lottie. Pero además, y aquí lo interesante, es que existen repositorios donde podemos encontrar cientos de animaciones para descargar.

Veamos algunos ejemplos.

Curso:

<https://wiki.genexus.com/commwiki/servlet/wiki?37189,HowTo%3A+Include+animations+in+applications>

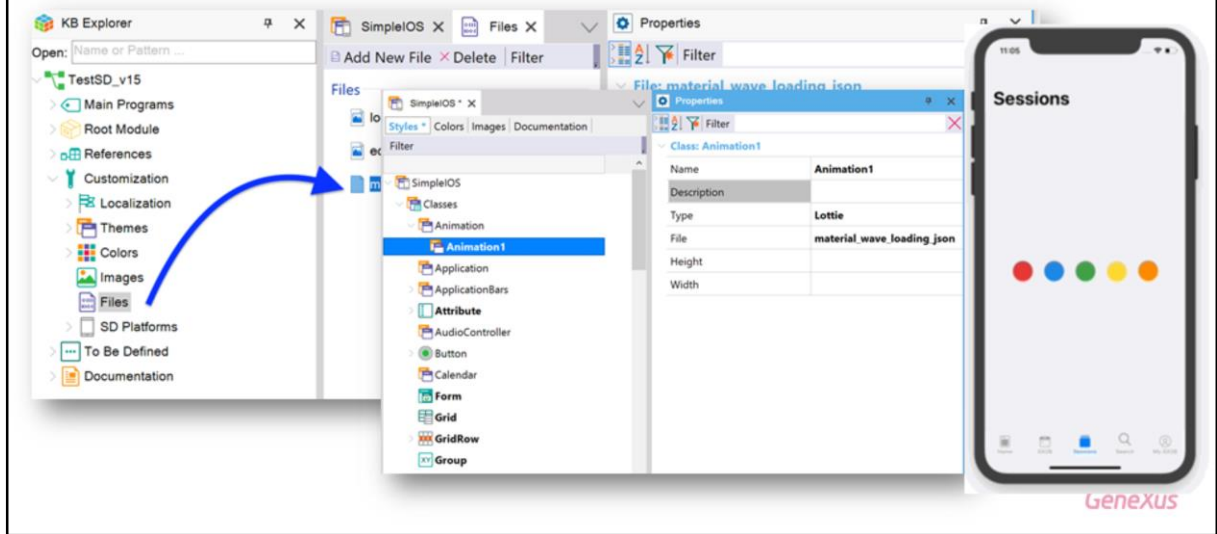
Lottie Animations

<https://www.lottiefiles.com>

GeneXus

Este es uno de los repositorios donde podemos encontrar varios Lotties (json) para descargar.

Animations : how to incorporate to the KB



Lo que tenemos que hacer entonces para incorporar una animación en nuestra app:

1. Obtener el JSON e importarlo en la KB como File.
2. Creamos una class en la categoría Animation Classes.
3. A dicha class le indicamos que es Type Lottie y en la propiedad File le atribuimos el archivo importado en el paso 1.
4. ¡Listo! Ahora estamos en condiciones de utilizar esa clase en diferentes partes de nuestra aplicación.

Veamos ahora dónde podemos utilizarlo.

Lottie Animations

There are four ways to make the Integration:

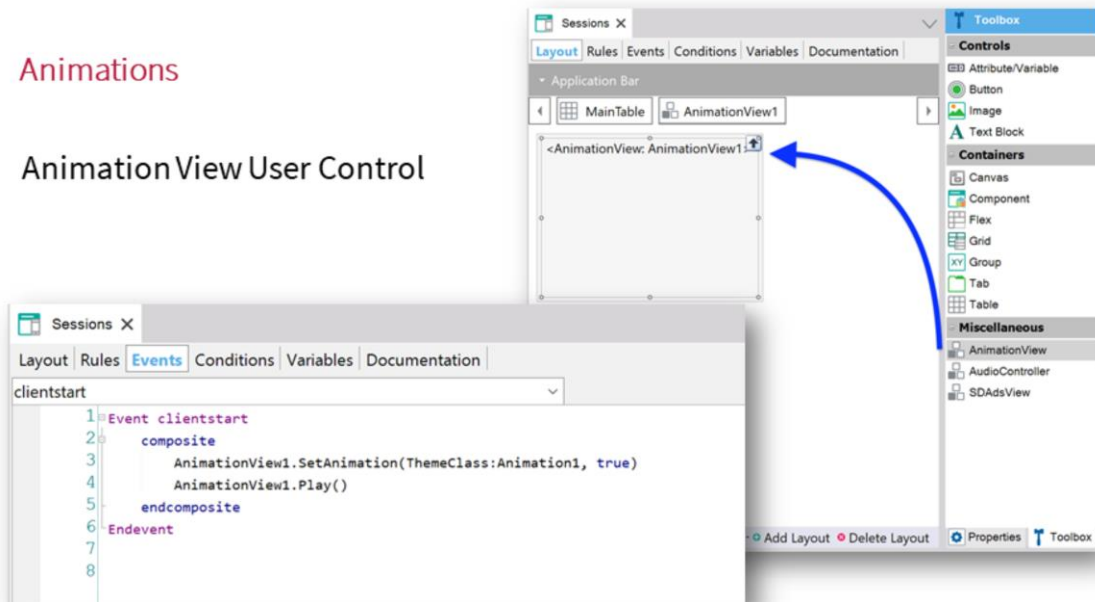
1. Animation View User Control

GeneXus

Podemos colocar una animación en cualquier lugar del layout, a través del User Control Animation View.

Animations

Animation View User Control



GeneXus

Available methods:

- SetAnimation (*AnimationClass*, *Loop*)
- SetProgress (*Progress*)
- Play (*FromPosition -optional-*, *ToPosition -optional-*)
- Pause ()

More info:

<https://wiki.genexus.com/commwiki/servlet/wiki?37185,Animation+View+User+Control,#SetAnimation>

Lottie Animations

There are four ways to make the Integration:

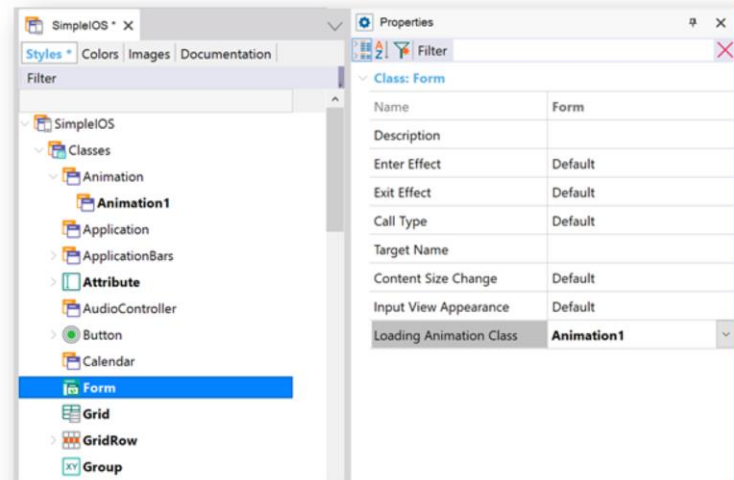
1. Animation View User Control
- 2. Customize the loading of Form and Grids**
- 3. Customize the Launch Screen**

GeneXus

Además podemos cambiar la animación predeterminada que se muestra al cargar el Form y el Grid...

Animations

Customize the loading of
Form and Grids



GeneXus

Esto se hace cambiando la propiedad Loading Animation Class en la clase correspondiente al Form o Grid.

Lottie Animations

There are four ways to make the Integration:

1. Animation View User Control
2. Customize the loading of Form and Grids
3. Customize the Launch Screen
- 4. Customize the Progress Indicator control**

GeneXus

Por último podemos usar animaciones para personalizar el control Progress Indicator

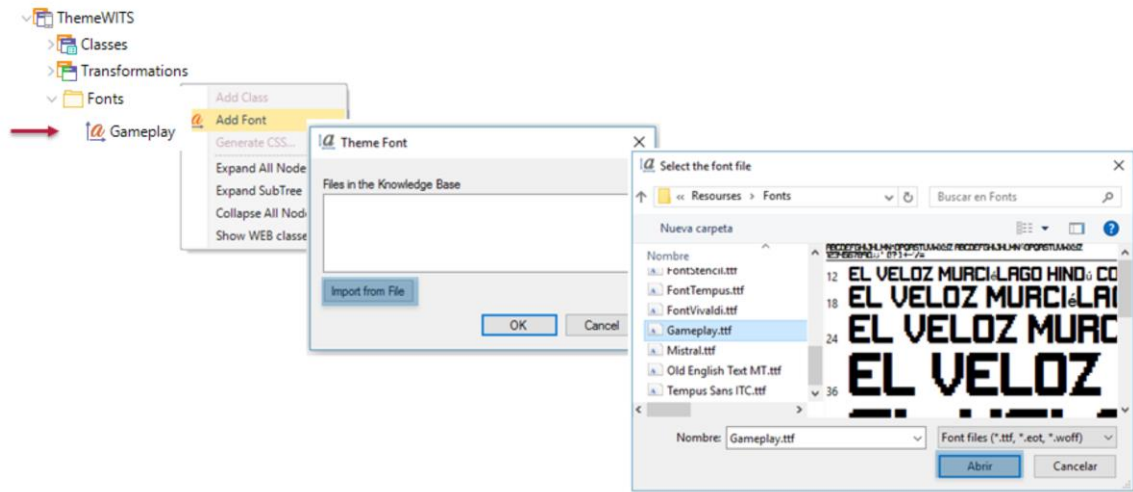
More info:

<https://training.genexus.com/actualizacion/actualizacion-de-genexus-evolution-3-a-genexus-15?es#integrar-animaciones-provistas-por-apis>

<https://wiki.genexus.com/commwiki/servlet/wiki?37189,HowTo%3A+Include+animations+in+applications>

Fonts

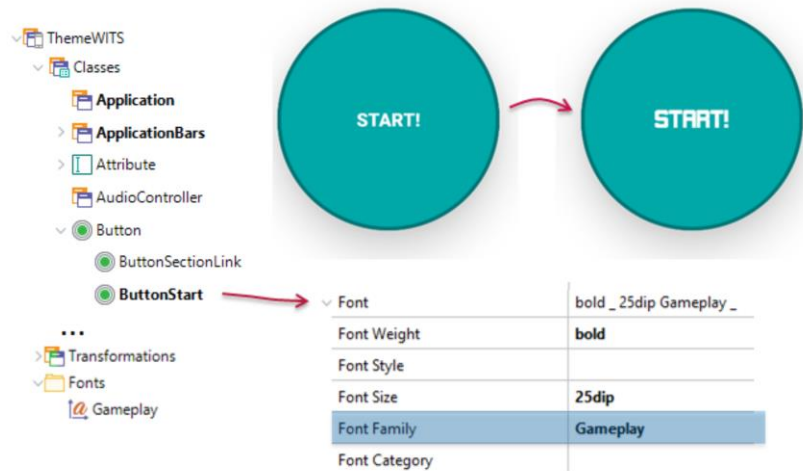
Adding Fonts



El objeto **Theme for Smart Devices** agrega un nuevo nodo **Fonts**, que ayuda al desarrollador a agregar fonts personalizados a la KB.

Esto permite que todos los recursos necesarios por la aplicación estén contenidos en la KB.

Adding Fonts



GeneXus

En el ejemplo agregamos una Font Gameplay al theme.

Luego en una clase button del theme podemos ver la propiedad **Font Family property** que muestra la nueva fuente en el combo-box.

Components

SD Components created in runtime too

Panel for Smart Devices

PanelGamePlay * X

Layout * Rules Events * Conditions Variables * Documenta...

Application Bar

MainTable QuizComponent

<Component: PanelGameQuizz>

&Hits &Misses &Temporizer

Hit Miss Time left

Chronometer

Component: QuizComponent

Control Name QuizComponent

Object PanelGameQuizz

Parameters

Panel for Smart Devices

PanelGameQuizz X

Layout * Rules Events Conditions Variables Documentation

Application Bar

MainTable

GRID

&RandomSongs.item(0).SongName

Now you can use **Component.Create(...)**!

GeneXus

Se había incorporado a la Toolbox el control Component. Una vez arrastrado al layout del Panel del que se trate (por supuesto puede ser también el layout de un Work With for Smart Devices), accediendo a sus propiedades se encontrará la de nombre **Object**, que es la que permite especificar qué objeto se cargará allí como componente. En nuestro caso cargaremos el Panel for Smart Devices PanelGameQuizz, que es el que carga las cinco canciones de la ronda actual. Nuestro PanelGameQuizz no tiene parámetros, pero si los tuviera, es en la propiedad **Parameters** del control Component donde éstos se especifican.

Estas propiedades, a diferencia de en Web, solo podían especificarse aquí, es decir, en el diseño. No podían especificarse dinámicamente, en runtime. ¡Ahora sí!

Por otro lado, si bien en Web un panel componente debía especificarse explícitamente de ese tipo, en SD no. Cualquier Panel for Smart Devices, e incluso Work with for Smart Devices podrá ser utilizado como componente.

Offline Applications

Features in GX 16

- More things/functions are generated on the device (offline generators)
- Security Improvements
- Support for multiple offline apps in the same KB
- Synchronization by TimeStamp

GeneXus

- More things/functions are generated on the device (offline generators)
 - iOS ahora se puede llamar a un proc Online desde uno Offline.
 - Los métodos ServiceEnabled() y Authorized() de Geolocation EO ahora se pueden usar offline en Android.
 - Se implementaron las funciones GetSOAPErr() y GetSOAPErrMsg() para aplicaciones iOS Offline
 - En aplicaciones offline se implementó el soporte para el tipo de dato Expression.
 - Se agrega la posibilidad de disparar un evento global desde un procedimiento offline para aplicaciones Smart Devices.
 - Progress Indicator se puede usar en objetos offline
- Security Improvements
 - Mayor seguridad en los servicios de sincronización con GAM
 - Propiedad "Encrypt Offline Database" para encriptar la base de datos sqlite
- Support for multiple offline apps in the same KB
- Synchronization by TimeStamp

Synchronization Receive Granularity Criteria

- By Table
- By Row
 - Using Hash
 - Using Timestamp

GeneXus

Antes de explicar el motivo por el cual se ha agregado la sincronización por Timestamp, hagamos un repaso sobre las diferentes opciones para la propiedad Receive Granularity Criteria, que es la que gobierna cómo se envían los datos desde el servidor al dispositivo (el Receive). Para luego ver los diferentes escenarios de uso de cada propiedad.

La propiedad tiene 2 valores: By Table y By Row

La diferencia entre TABLE y ROW es la cantidad de información que se envía del servidor al dispositivo.

Receive By Table

For each table involved in the synchronization, if the table was changed, it sends **all table records** from the server to the device.



GeneXus

By Table

Si se cambia algún registro en el server de la tabla que participa de la app offline, al sincronizar se borra todo el contenido de la tabla local y se envían todos los datos nuevamente.

Es el mecanismo más simple, prácticamente no hay procesamiento del lado del servidor, pero hay que tener cuidado con el tráfico de datos.

Útil cuando:

las tablas que participan tienen pocos registros

en cada sync se requieren actualizar la mayoría de los datos.

en caso que sean muchos registros, que se tenga una conexión acorde y los intervalos de sync no sean pequeños

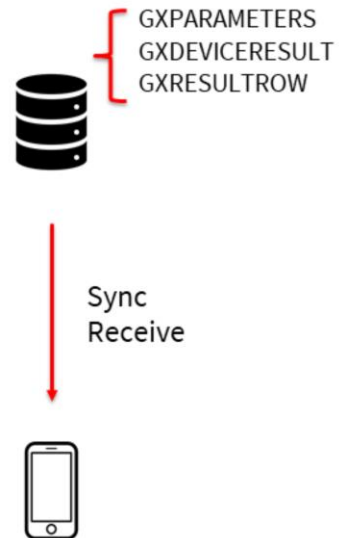
Ejemplo: Una app para vendedores, donde tengo la lista de clientes a visitar cada día. La lista cambia casi por completo cada día.

Cuando las tablas que participan de la sync son grandes, este mecanismo podría resultar impráctico por el incremento en el tráfico de datos.

Receive By Row using Hash

For each table involved in the synchronization, if the table was changed, it sends **only the modified rows** (inserted, updated or deleted) from the server to the device.

Differences are computed by using **row hashes**.



GeneXus

By Hash

En el segundo modo “By Row”, el objetivo es enviar únicamente los cambios realizados en el Server a cada cliente, en forma periódica, minimizando el volumen de tráfico, aunque esto requiere de procesamiento adicional del lado del server.

Útil cuando:

Las tablas son medianas o grandes.

Muchos usuarios (ej. App publicada en las stores)

En cada sincronización (intervalo) las tablas cambian algunos registros (no la mayoría).

Con esto se logra menos tráfico en cada sincronización

Requiere de tablas auxiliares del lado del server, pero GeneXus se encarga de crearlas y mantenerlas. No es necesario actualizar las tablas del modelo.

Requiere de más procesamiento y storage del lado del servidor para calcular y mantener los hash de cada tabla y registro sincronizado (un hash para cada registro enviado a cada device).

Si la tabla que participa de la sync (en realidad el datasets que se sincroniza para esa tabla) es muy grande, calcular esos hash en cada sync puede sobrecargar el servidor.

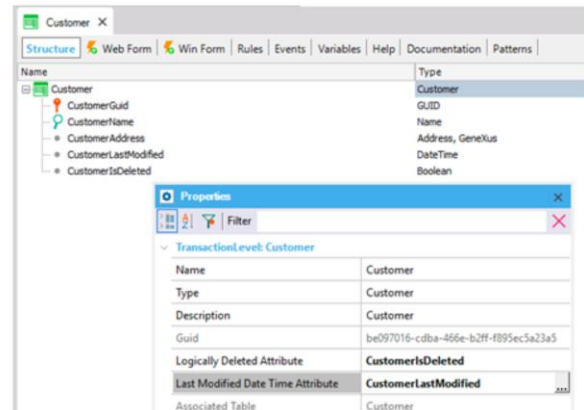
Receive By Row using Timestamp

Sends **only the modified rows** (inserted, updated or deleted) from the server to the device.

Differences are computed by using **row timestamp**.

Requires the addition of two attributes:

- Last Modified Date Time Attribute
- Logically Deleted Attribute



GeneXus

By Timestamp

Los casos anteriores puede ser poco performantes si las tablas que participan de la app Offline (el datasets) son muy grandes (es decir, se requiere de la mayoría de los datos en el device), cambian algunos pocos datos en el servidor y las sincronizaciones son frecuentes.

La sync por tabla no es adecuada porque son muchos registros para enviar cada vez.

La sync por hash tiene el inconveniente de que requiere calcular y mantener actualizados muchos hash. Si el datasets que se requiere en el device para la tabla es muy grande, y la misma fue actualizada en el servidor, por más que sean pocos los registros alterados, se tienen que calcular todos los hashes de cada registros nuevamente para saber cuales cambiaron.

Aquí es donde puede ser útil esta nueva opción de sincronización por TimeStamp.

En este caso la sincronización continúa siendo por registro (by row) pero no usa los hashes para computar las diferencias. Se obtienen directamente aquellos registros que cambiaron desde la última sync con cada device consultando por su fecha de modificación.

Por este motivo este algoritmo requiere de dos atributos adicionales: [Last Modified Date Time attribute](#) y [Logically Deleted attribute](#).

Aquí es necesario entonces la intervención del desarrollador para agregar esos atributos y las reglas necesarias para actualizar cuando corresponda. Además para esa tabla ya no se puede realizar un borrado físico, tiene que ser lógico.

La ventaja es que este algoritmo se puede aplicar por tabla, entonces algunas pueden mantener la sincronización por hash y en aquellas que sea necesario la sincronización por timestamp cambiar solo éstas.

Mas info:

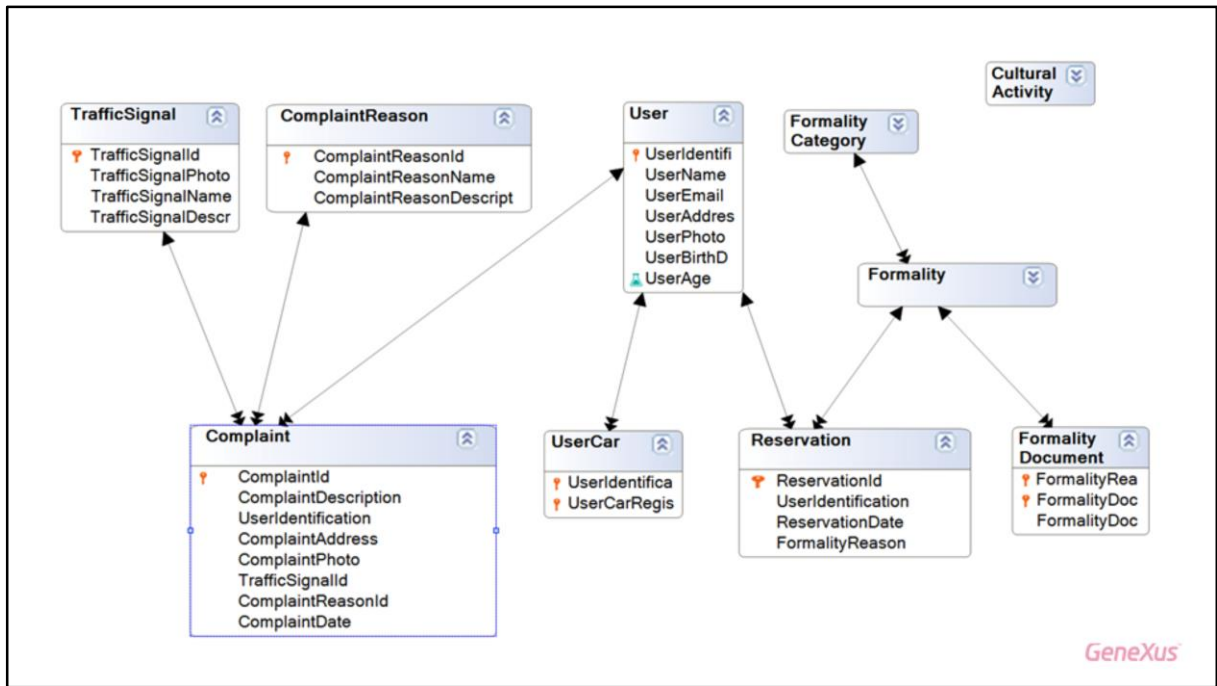
<https://wiki.genexus.com/commwiki/servlet/wiki?37109,Smart+Devices+offline+synchronization+granularity+alternatives>,

Offline Applications

Special considerations about Synchronization.Receive

Ahora les quiero comentar sobre algunas consideraciones a tener en cuenta cuando desarrollamos apps Offline.

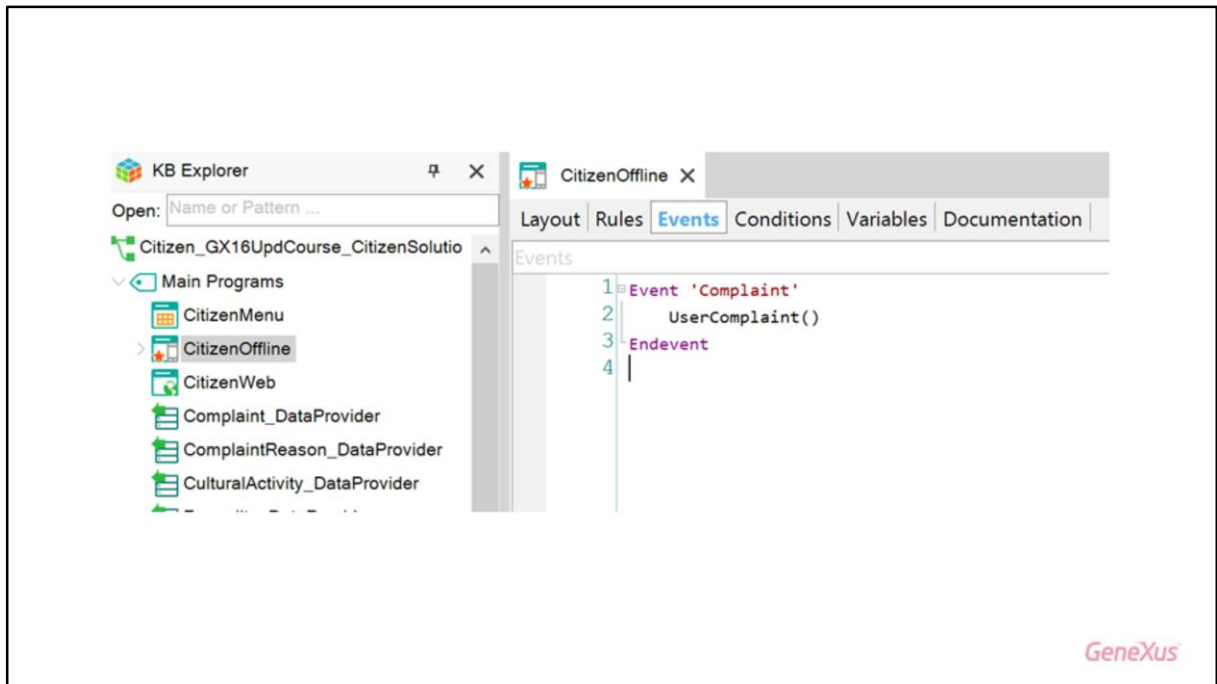
Para lograr que la misma sea eficiente y únicamente se estén llevando al dispositivo los mínimos datos necesarios.



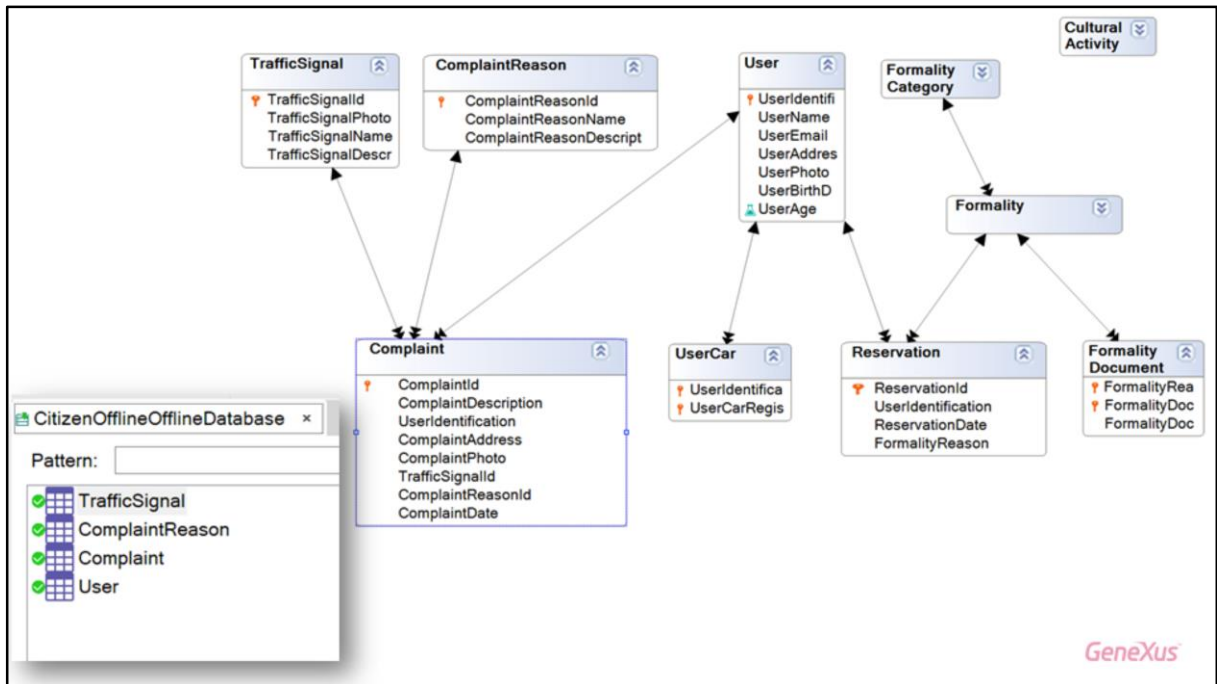
Pongamos como ejemplo el ejercicio del práctico, del sistema del gobierno local / intendencia / prefeitura

Supongamos que queremos desarrollar una app donde simplemente queremos que los usuarios registren los reclamos.

La misma tiene que funcionar offline porque el usuario puede encontrarse en algún lugar con poca conectividad.



Entonces la app solo llama al objeto SD donde se permite ingresar un nuevo reclamo.

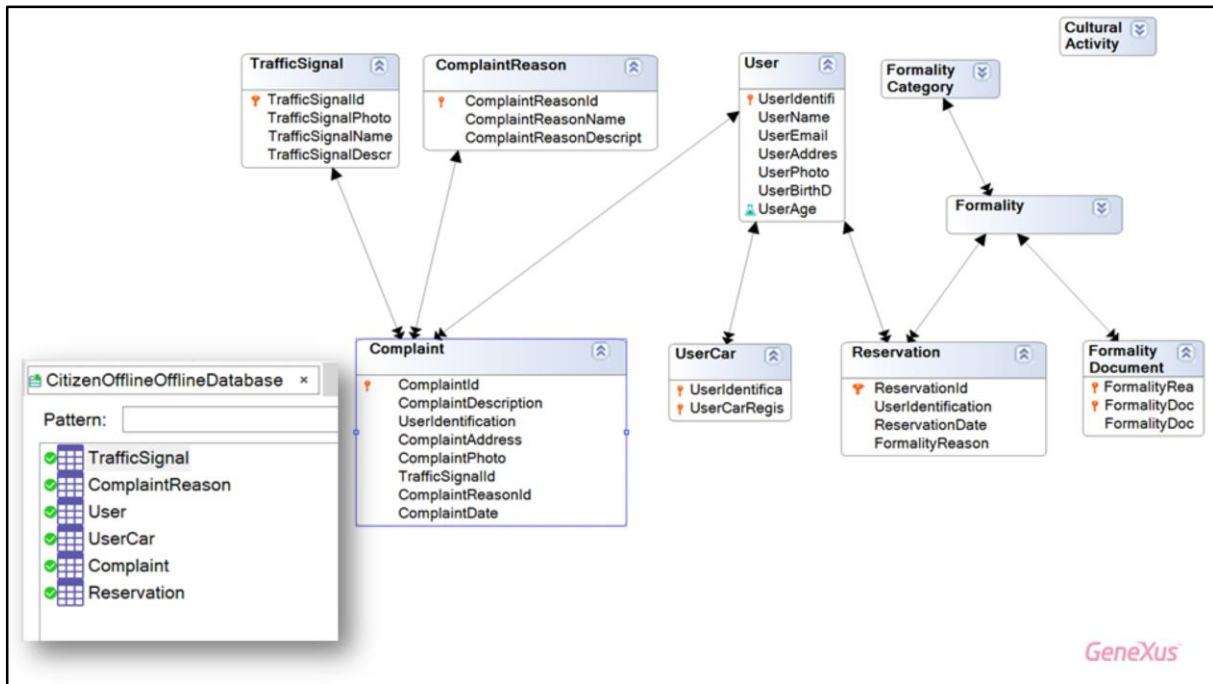


Cuando realizamos el build de la app Offline se muestra el análisis de impacto de las tablas que van a participar en nuestra app, es decir las tablas a ser creadas en localmente en el SQLITE.

Este es el primer punto donde debemos prestar atención:

¿Están siendo consideradas el mínimo conjunto de tablas necesarias para la app funcionar offline?



Tener tablas innecesarias implica que los algoritmos de sincronización estén realizando cálculos para detectar los cambios de registros que no nos interesa.



Esta es el IAR cuando cree el objeto Offline mencionado antes.

Vean que se agregaron tablas que no me interesan en la app (Reservation y UserCar).

Eso fue porque en el objeto donde se inserta el reclamo tenía una variable basada en el BC User que no era necesario. Entonces eso agregó las otras tablas relacionadas por la TRN User, porque la TRN asociada al BC invocaba al WWUser y éste a su vez a las otras transacciones.

Event Synchronize by hash for Complaint (Line: 13)	
For Each Complaint (Line: 13)	
Order:	ComplaintId
	Index: ICOMPLAINT
Navigation filters:	Start from: FirstRecord
	Loop while: NotEndOfTable
 =Complaint (ComplaintId)	
Load into Complaint	
Event Synchronize by hash for User (Line: 18)	
For Each User (Line: 18)	
Order:	UserIdentification
	Index: IUSERS
Navigation filters:	Start from: FirstRecord
	Loop while: NotEndOfTable
 =User (UserIdentification)	
Load into User	

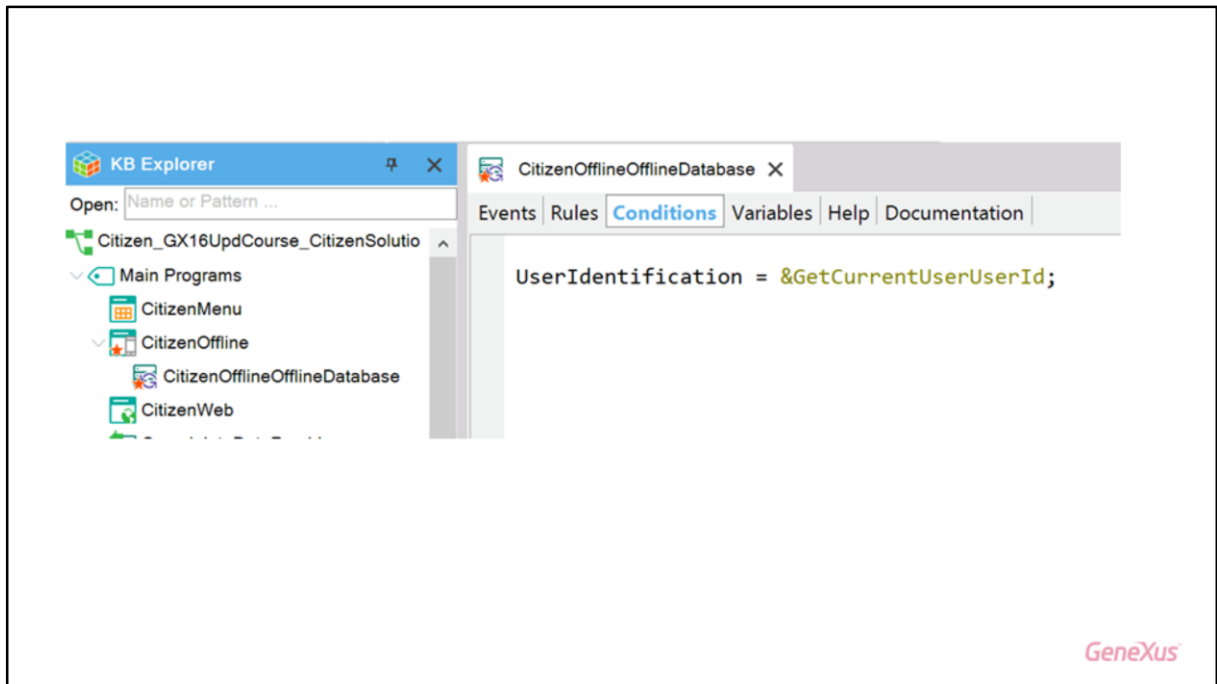
ieXus

No solo es importante entonces considerar el mínimo conjunto de tablas, también el mínimo conjunto de registros necesarios.

Notar que esta navegación del objeto OfflineDatabase nos está indicando que se están llevando, a todos los devices, todos los registros de la tabla de reclamos (Complaint) y todos los usuarios de mi sistema!




¿Por qué esto es importante?

No solo por aspectos de seguridad o de espacio requerido por la app en el device. Además porque los algoritmos de sync por Row no consideran toda la tabla para computar las diferencias, sino solo aquellos registros que participan de la sync (datasets). Por lo tanto, menos registros para sincronizar, menos hash para computar. Esto es, menos procesamiento del lado del server y las tablas auxiliares para la sync no crecen tanto.



¿Cómo resolvemos esto?

Agregando las condiciones necesarias en el objeto OfflineDatabase.

Event Synchronize by hash for Complaint (Line: 13)	
For Each Complaint (Line: 13)	
Order:	<u>UserIdentification</u> Index: ICOMPLAINT1
Navigation filters:	Start from: <u>UserIdentification</u> = &GetCurrentUserUserId Loop while: <u>UserIdentification</u> = &GetCurrentUserUserId
	
	 =Complaint (<u>ComplaintId</u>)
	Load into Complaint
Event Synchronize by hash for User (Line: 18)	
For First User (Line: 18)	
Order:	<u>UserIdentification</u> Index: IUSERS
Navigation filters:	Start from: <u>UserIdentification</u> = &GetCurrentUserUserId Loop while: <u>UserIdentification</u> = &GetCurrentUserUserId
	
	 =User (<u>UserIdentification</u>)
	Load into User

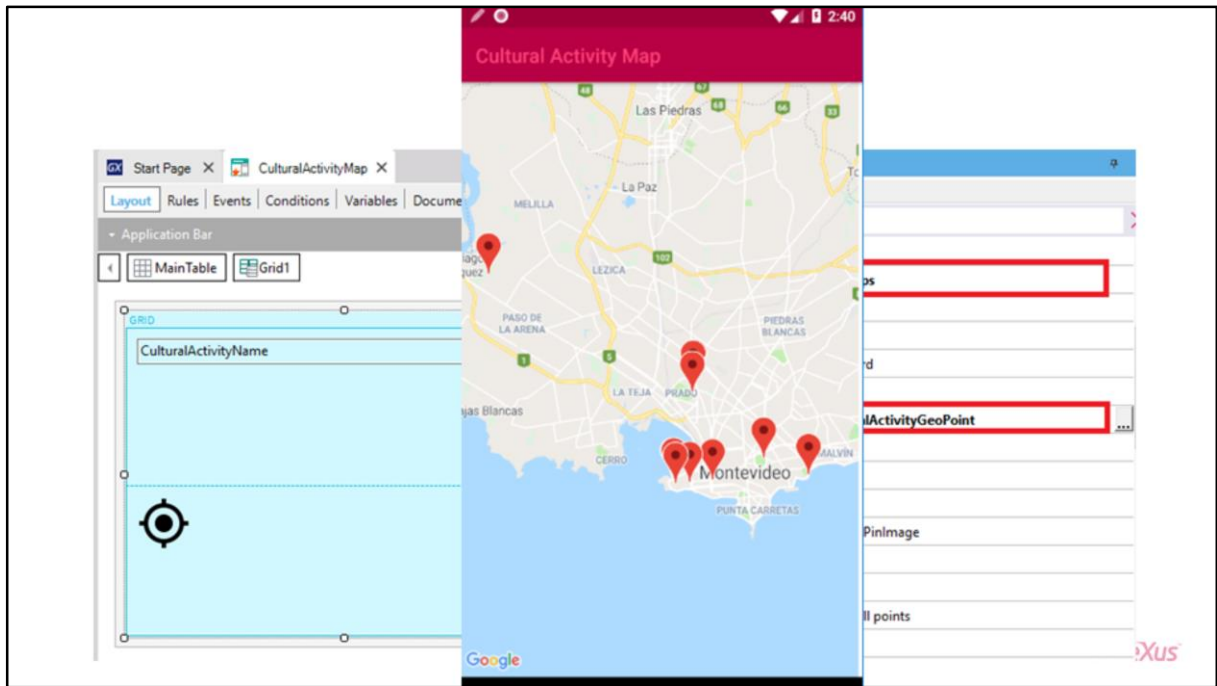
reXus

Con esa simple condición, logramos que se transfieran al device únicamente los datos de cada usuario o device, y que los algoritmos de sincronización solo tomen en cuenta los cambios en ESOS registros para computar las diferencias.

Synchronization Receive Summary

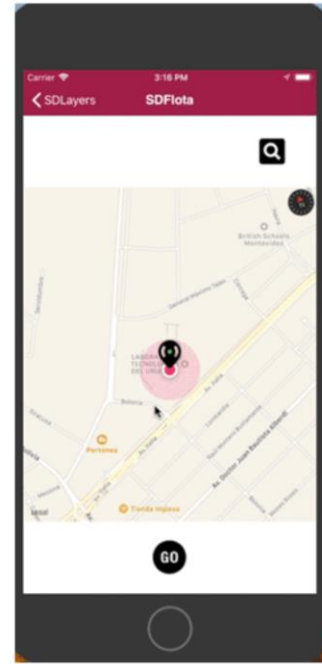
- Check involved tables and their load conditions
 - <Main>OfflineDatabase database report
 - <Main>OfflineDatabase navigation report
- Minimum time between receives
- Data Receive granularity

Multi Layers - SDMaps



<https://wiki.genexus.com/commwiki/servlet/wiki?15309,SD+Maps+Control>,
<https://wiki.genexus.com/commwiki/servlet/wiki?36729,PickLocation%20Method>

Transport application



Maps, tracking & localization al estilo Uber

Selection Layer

General Class

Control Type

Auto Grow

Show My Location

Map Type

User Can Choose M

Location Attribute

Pin Show My Locat

Pin Image

Pin Image Attribute

Pin Image Class

Show Navigation o

Show Traffic

Initial Zoom

Center

Selection Layer

Location Selection Target Image

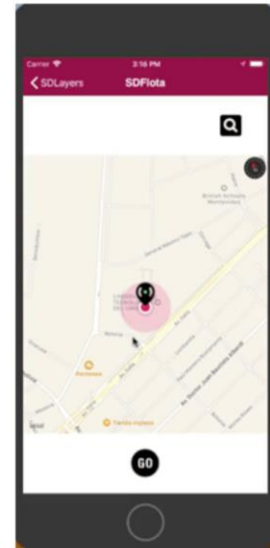
Selection Target Image Class

Directions Layer

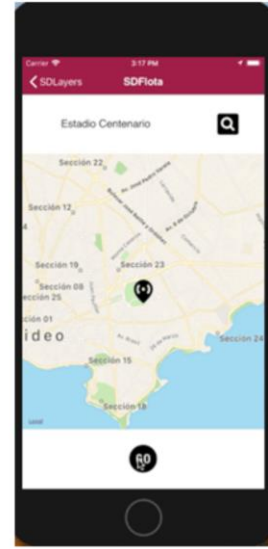
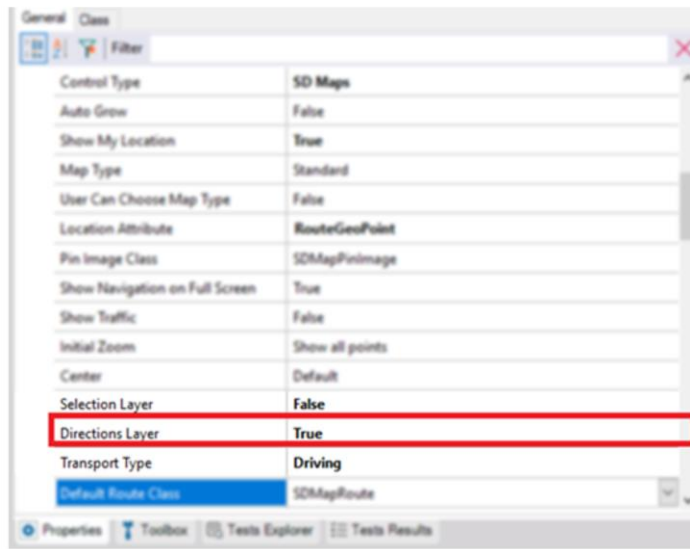
```
event Grid1.ControlValueChanged(&geopoint)
composite
  msg("controlvaluechanged")
  msg(&geopoint.ToString())
endcomposite
endevent

Event Grid1.ControlValueChanging(&geopoint)
...
endevent
```

Show all points
My location
True
locationBlack
SDMapPinImage
False



Direction Layer



Direction - External object

The screenshot displays a software development environment with two main panels. The left panel, titled 'Structure', shows a tree view of the 'Maps' object. It includes a 'Methods' section with two 'CalculateDirections' entries, each with parameters for sourceLocation, destinationLocation, transportType, and requestAlternateRoutes. It also includes an 'Events' section with a 'DirectionsCalculated' event, which has parameters for routes and errorMessage. The right panel shows the code for these methods and events. The first event, 'Go.Tap //GO', calls 'Maps.CalculateDirections(&geopoint,&mygeopoint)'. The second event, 'Maps.DirectionsCalculated(&routes, &messages)', is a composite event that checks if the number of messages is zero. If so, it iterates through the routes, drawing a geoline for each. Otherwise, it displays a message.

```
Event Go.Tap //GO
|
| Maps.CalculateDirections(&geopoint,&mygeopoint)
|
Endevent

Event Maps.DirectionsCalculated(&routes, &messages)
composite
  if &messages.Count = 0
    &i = 1
    do While &routes.Count >= &i
      &geoline = &Routes.Item(&i).geoline
      Grid1.DrawGeoline(&geoline,"Class")
      &i= &i + 1
    enddo
  else
    msg(&messages.Item(1).Description)
  endif
endcomposite
endevent
```

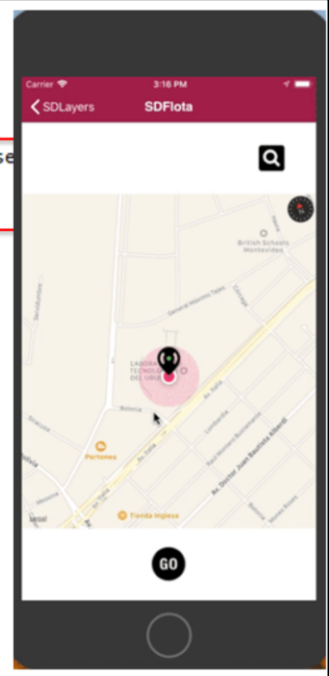
How to programming Transport application

```
Event ClientStart
  composite
    &GetMyLocation = GeneXus.Common.Geolocation.GetMyLocation(0,0,false
    &myposition = &GetMyLocation.Location
    InsertRouteGeoPoint(&myposition)
  endcomposite
Endevent

event Grid1.ControlValueChanged(&destinationpoint)
  ...
endevent

Event ImgGo.Tap
  composite
    InsertRouteGeoPoint(&destinationpoint)
    Grid1.SelectionLayer = false
    Grid1.DirectionsLayer = true

    Grid1.Refresh()
  endcomposite
Endevent
```



Maps, tracking & localization al estilo Uber

IOS Map Providers



Main Object Property – Apple Maps API



Cloud External Storage

Cloud external storage

Storage Provider Property

- AmazonS3
- Microsoft Azure
- Ibm Bluemix
- Google Cloud Storage
- Local (default)
- OpenStack

Generator: Default (Java)

Name	Default
User Interface	Web

General

Use Native Soap	No
Java package name	com.teststore
Use decimal arithmetic	Yes

Java specific

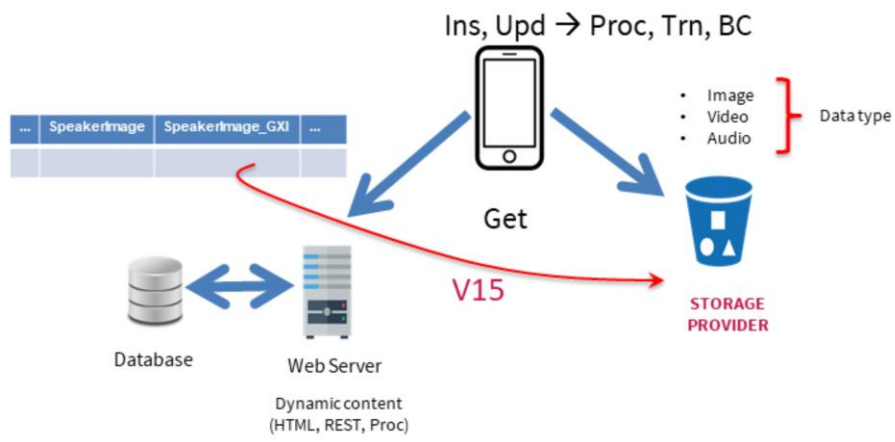
SMTP server (for mail functions)	
Log JDBC Activity	No

Services

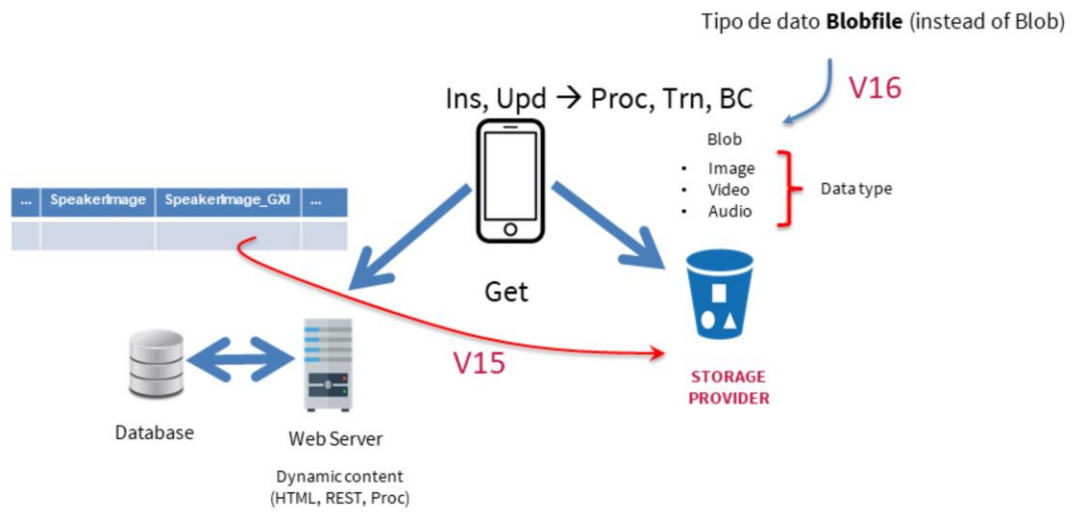
Storage configuration

Storage Provider	Amazon S3
Bucket Name	genexuss3test
Folder Name	test
Storage Access Key ID	1234567890
Storage Secret Access Key	*****
Storage Region	US Standard/US East (N. Virginia)

Cloud external storage



Cloud external storage



GeneXus IDE

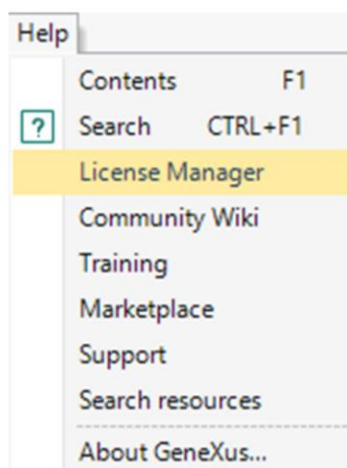
Associated Tables

The screenshot displays the GeneXus IDE interface. On the left, the 'KB Explorer' pane shows a tree structure under 'Entities_Trns'. The 'CulturalActivity' entity is expanded, revealing its 'Associated Tables' section. The 'CulturalActivity' table is selected. The main pane shows the 'Structure' tab for this table, displaying a list of fields with their names, types, and descriptions.

Name	Type	Description	Formula
CulturalActi... Id	Id	Cultural Activity Id	
• CulturalActi...	VarChar(60)	Cultural Activity Name	
• CulturalActi...	VarChar(1K)	Cultural Activity Descri...	
• CulturalActi...	CulturalActivitiesC...	Cultural Activities Cate...	
• CulturalActi...	Image	Cultural Activity Photo	
• CulturalActi...	Boolean	Cultural Activity Featured	
• CulturalActi...	GeoPoint	Cultural Activity Geo P...	
• CulturalActi...	Image	Cultural Activity Static ...	

GeneXus IDE

License Manager



Language/Modeling - For each

Foreach *BaseTransaction*

SKIP *expression* **COUNT** *expression*

order *att*₁, *att*₂, ... , *att*_n [when *condition*]

order *att*₁, *att*₂, ... , *att*_n [when *condition*]

unique *att*₁, *att*₂, ... , *att*_n

using *DataSelector*(*parm*₁, *parm*₂, ... , *parm*_n)

where *condition* [when *condition*]

where *condition* [when *condition*]

where *att* IN *DataSelector*(*parm*₁, *parm*₂, ... , *parm*_n)

blocking *N*

Main_code

When duplicate

When_duplicate_code

When none

When_none_code

Endfor

Paging

```
Event Grid1.Load()
  for each Session
    skip &pageSize*(&pageNumber-1) count &pageSize
    order SessionName
      &SessionName = SessionName
      Load
    endfor
  Endevent
```

Aquí tenemos la sintaxis completa del comando for each, donde podemos ver que se han agregado las cláusulas SKIP y COUNT, para tener la posibilidad de paginar.

Con SKIP indicamos que queremos saltarlos los primeros X registros (siendo X el resultado de evaluar la expresión que sigue a SKIP) y quedarnos con los siguientes Y, siendo Y el valor de la expresión indicada luego del COUNT.

Vemos un ejemplo donde estamos elaborando el paginado de un grid sin tabla base.

Dirigirse a nuestro wiki para recordar/profundizar en el comando y sus cláusulas:
<http://wiki.genexus.com/commwiki/servlet/wiki?24744,For+Each+command>

Modeling

Unique Grid Property

Layout Rules Events Conditions Variables Documentation

Application Bar

MainTable Grid1

GRID

CulturalActivityName

Control Name Grid1

Collection

Default Action <default>

Selection Type Platform Default

Enable Multiple Selection False

Pull To Refresh False

Inverse Loading False

Default Selected Item Layout (none)

> Control Info

> Appearance

> Data Selector

> Layout Behavior

> Cell Information

> Refresh timeout

< Data

Orders (0 orders)

Search (0 filters)

Conditions

Base Tm

Unique CulturalActivityName

Data types

- DateTime data type: soporte de milisegundos
- File data type: New GetURI method added (Storage Provider API)
- Video data type: Youtube videos supports start-time through "t" and "start" query strings

<https://youtu.be/frdj1zb9sMY?t=1m51s>

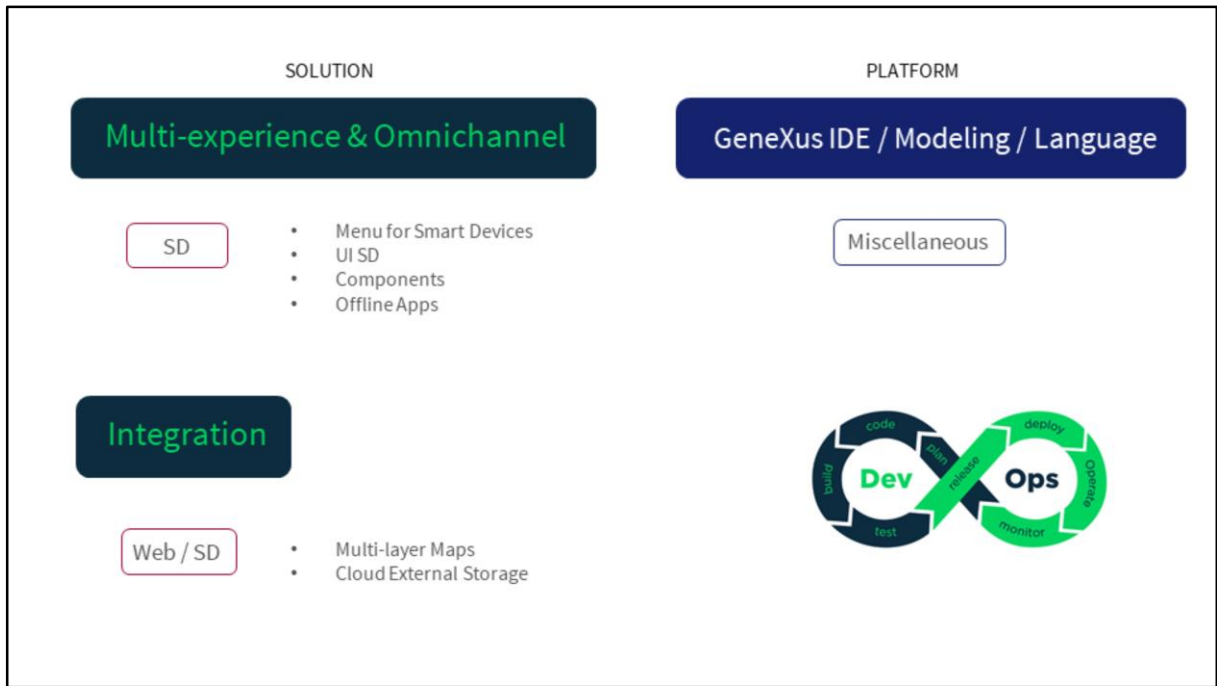
<https://www.youtube.com/embed/frdj1zb9sMY?start=111>

- Blobfile data type

File data type: New GetURI method added

(for cases when working with files returned by StorageProvider API methods (u9)), The consistency of GetPath, GetName and GetAbsolute methods have been improved (Results differ respect to previous versions when working with files returned by Storage Provider API methods) (u9)

<https://wiki.genexus.com/commwiki/servlet/wiki?40420,BlobFile%20data%20type>



En esta sección vimos todo esto. Pasemos a ver algunos aspectos más de integración.

GeneXus™
The power of doing