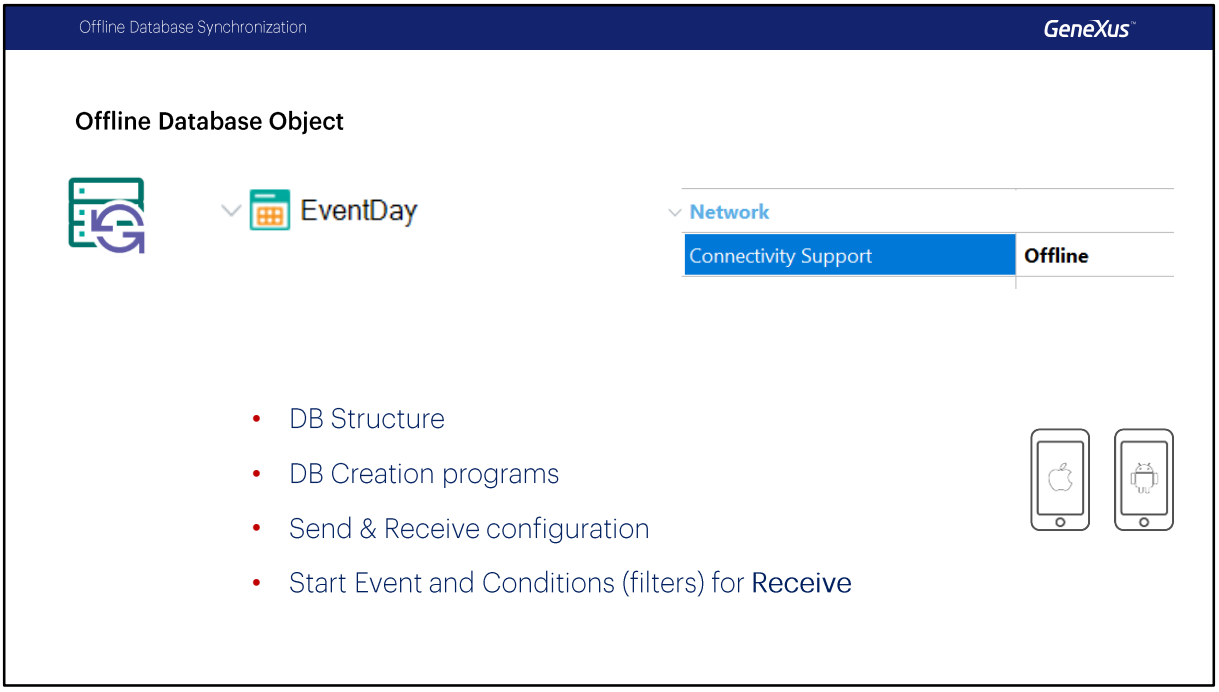

GeneXus™
The power of doing.

Offline Database Synchronization

Development – Offline Applications

GeneXus™ 16



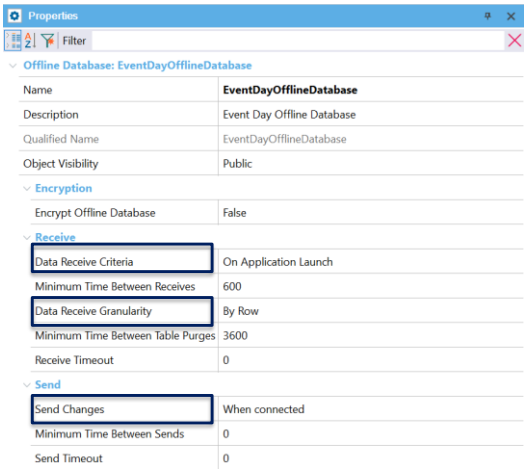
En este video continuaremos hablando de las aplicaciones Offline. Vamos a comenzar viendo el objeto Offline Database.

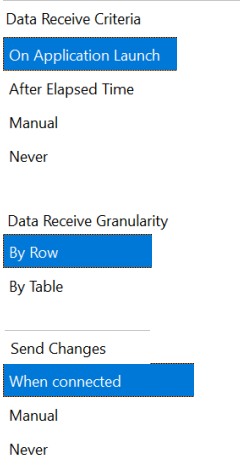
Para cada objeto main que tenga la propiedad Connectivity Support = Offline, se crea un objeto llamado Offline Database.
Éste se crea la primera vez que se hace un build sobre dicho objeto.
Al crearse, podremos ver cuáles son las tablas que se crearán en la BD local y se crean también en el lenguaje nativo del dispositivo, los programas para crear la base de datos local.

Este objeto es el encargado de determinar cuándo se produce la sincronización y también cuáles son los datos que se llevan a las mismas cuando se sincronizan con las tablas del server.

El objeto Offline Database dispone además de eventos y condiciones que permiten determinar su comportamiento, esto lo veremos en el nivel avanzado.

Offline Database Properties





Vamos a ver algunas de las propiedades mas importantes del objeto Offline Database.

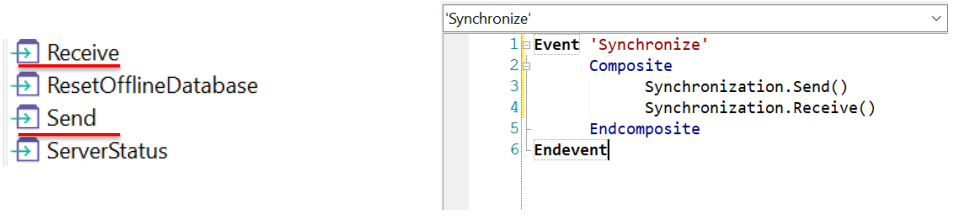
En el grupo Receive tenemos la propiedad Data Receive Criteria que se usara para la recepción, los valores que podemos utilizar son:
Cuando se lance la aplicación, que es el valor por default, indica que la aplicación iniciara la recepción de datos cuando esta se inicie.
Con el valor After Elapsed Time, la recepción se hará cuando se cumpla el tiempo indicado en la propiedad Minimum Time Between Receives, expresado en segundos, además también se realizara cuando la aplicación se inicialice.
El valor Manual indica que la recepción se efectuara solo en forma manual, aquí se deberá desarrollar una acción para lanzar la recepción.
Por ultimo, el valor Never indica que nunca se llevara a cabo la recepción de datos por defecto y GeneXus tampoco generara los programas de sincronización necesarios, los cuales deberán ser desarrollados manualmente en caso necesario.

Como ya habíamos visto, tenemos la propiedad Data Receive Granularity, que nos permite establecer cual será el mecanismo de recepción a implementar, puede ser by Row que es el valor por defecto o by Table.

Luego tenemos la propiedad Send Changes que nos permitirá establecer cuando deseamos

realizar el envío de los datos desde el dispositivo hacia el server, los valores que disponemos son:
When Connected, que es el valor default, indica que el envío se realizara en forma inmediata cuando se detecte que hay conexión.
El valor Manual, indica que el envío se efectuara solo en forma manual, aquí se deberá desarrollar una acción para lanzar la recepción.
Y el valor Never, indica que no se enviaran datos en forma automática por GeneXus, los registros modificados no se almacenaran en la tabla auxiliar GXPendingEvent, por lo que queda la responsabilidad del lado del desarrollador en caso de necesitar el envío, de programar toda la lógica necesaria.

Synchronization API



Ya sea en el caso de usar para el envío o la recepción el valor Manual o para darle mayor funcionalidad al usuario, necesitamos poder desarrollar acciones que lleven a cabo la sincronización.
Para ellos contamos con la API Synchronization.

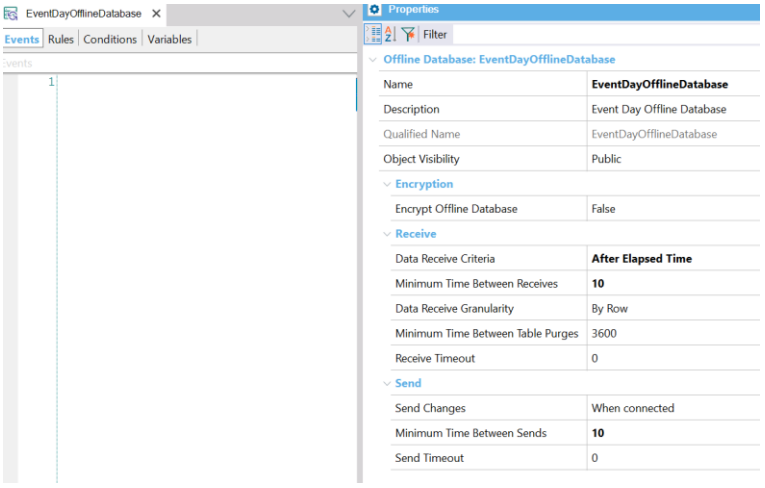
Esta API no se encuentra en las referencias como el resto de las APIs sino que es parte de la gramática.
Cuenta con los métodos Send y Receive para la sincronización, ServerStatus para determinar el estado del server, y ResetOfflineDatabase que retorna la base de datos local a su estado inicial, ya sea haciendo un Create Database para vaciar las tablas o cargando una base datos precargada.

Con el uso de esta API podríamos por ejemplo desarrollar en un panel opciones para que el usuario dispare la sincronización cuando el lo desee.
Aquí podemos ver un ejemplo muy simple donde se utilizan los dos métodos en un mismo evento.

Demo: Synchronization API

Vamos a GeneXus.

Offline Database properties



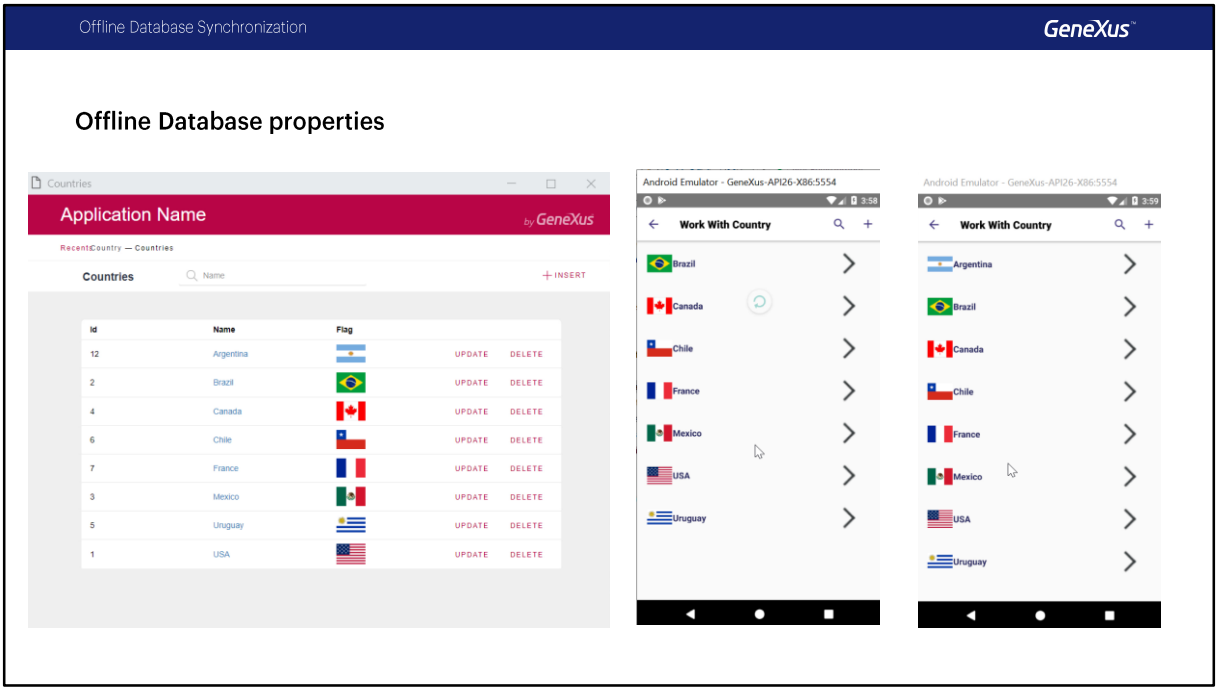
Primero vamos a configurar el objeto Offline Database para que la sincronización se realice cada 10 segundos.

Abrimos el objeto EventDatyOfflineDatabase.
Vamos a las propiedades y en la propiedad Data Receive Criteria vamos a poner el valor After Elapsed Time y en Minimum Time Between Receives vamos a utilizar el valor 10.

Ahora en el grupo Send, en la propiedad Minimum Time Between Sends también vamos a poner el valor 10.

Con esta configuración nuestra aplicación va a enviar y recibir los cambios cada 10 segundos.

Vamos a probar esto. grabamos y corremos la aplicación.

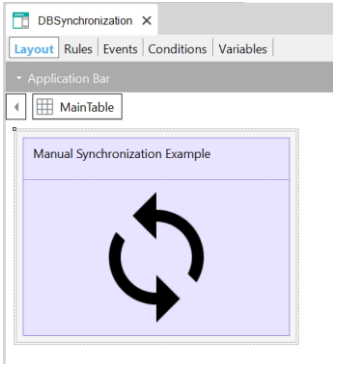


Bien, ya tenemos los cambios.
Vamos a ingresar a Countries en el dispositivo y acá tengo también el panel Web de países.
vamos a crear un país desde el backend web.
En el Nombre ingresamos Argentina, seleccionamos la bandera. y confirmamos.
Vamos a ir refrescando el panel, recuerden que indicamos 10 segundos.

Ok ya tenemos el país.
ahora ingresemos al país que creamos y vamos a borrarlo. confirmamos. se elimino desde el dispositivo.
Vamos al web y vamos a refrescar. Listo.

ahí ya esta actualizada en el server.

Synchronization API



DBSynchronization

Layout | Rules | Events | Conditions | Variables

Image1.Tap

```
Event Image1.Tap
      Composite
            GeneXus.Common.UI.Progress.Title = "Synchronizing Database"
            GeneXus.Common.UI.Progress.Description = "Wait until Synchronization is done."
            GeneXus.Common.UI.Progress.Type = ProgressIndicatorType.Indeterminate
            GeneXus.Common.UI.Progress.Show()
            Synchronization.Send()
            GeneXus.Common.UI.Progress.Description = "Send operation Ended."
            Synchronization.Receive()
            GeneXus.Common.UI.Progress.Description = "Receive operation Ended."
            GeneXus.Common.UI.Progress.Hide()
      Endcomposite
Endevent
```

Pero además, vamos a programar un panel que nos permita realizar la sincronización a pedido del usuario.
Yo ya cree un Panel, DBSynchronization vamos a verlo. lo vamos a agregar al menú EventDay. grabamos.

Vamos a ver este panel. En el layout tengo simplemente una imagen y queremos que cuando el usuario haga Tap sobre el icono se lance la sincronización, vamos a programar entonces el evento Tap.

En este evento como es del lado del cliente, vamos a usar composite. recuerden que siempre la sincronización se va a iniciar desde el cliente.

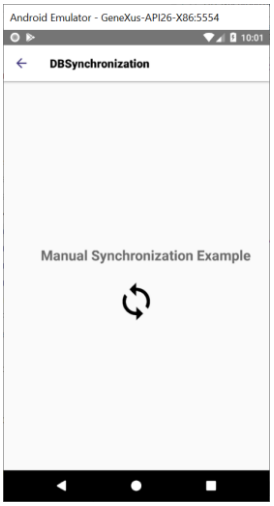
Entonces escribimos Syncrhronization.Send() esto va a realizar el envío de los datos del dispositivo hacia el server
y Syncrhronization.Receive(), vean que tenemos otros métodos, de los cuales ya hablamos, ServerStatus, ResetOfflineDatabase.
bien, vamos a usar Receive para hacer la recepción.

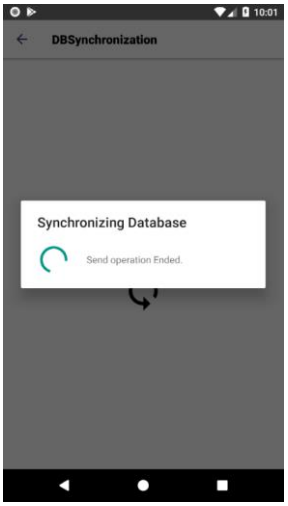
pero además vamos a brindarle al usuario algún feedback sobre la acción que esta realizando.

Para esto vamos a utilizar Progress.
Progress es un objeto externo que nos permite mostrar como un panel donde vemos lo que esta pasando.
El titulo vamos a poner "Synchronizing Database“, este va a ser el titulo.
Y vamos a poner una descripción
entonces de vuelta, Progress.Description y vamos a poner "Wait until Synchronization is done.“
Tenemos que indicar el tipo de progreso que vamos a usar, vamos a usar indeterminado, que se usa cuando no sabemos, no podemos indicar el grado de avance del proceso.
Luego mostramos el control con Progress.Show()
A continuación haremos el Send. dejamos esta línea.
Cuando termine el Send podríamos cambiar la descripción
entonces vamos a poner
Progress.Description = "Send operation Ended.“
Luego hacemos el Receive vamos a dejar esta línea.
Y nuevamente cambiamos la descripción para indicar que termino
Progress.Description = "Receive operation Ended."
Y lo ultimo que tenemos que hacer es ocultar el control.
entonces ponemos Progress.Hide() que es el método que lo oculta.

Bien.

Synchronization API





vamos a ejecutar la aplicación.

Bien. Ya tenemos la aplicación en el emulador.
Entonces vamos a probar directamente la nueva opción de sincronización.
Hacemos Tap en el icono y ahí se abre el indicador de proceso, notaron como se mostro la descripción y luego se oculta.

Vamos a acceder a la web, y vamos a agregar nuevamente un país, el país Argentina de vuelta, seleccionamos la bandera y Confirmamos.
Vamos al emulador, no vamos a esperar a que sincronice entonces ahora lo vamos a hacer en forma manual.
hacemos tap sobre el icono, ahí nos indica que el send ya termino, no llegamos a ver el siguiente mensaje.
Y ahora vamos a countries y ya tenemos Argentina.

ahora vamos a borrarlo desde el dispositivo, confirmamos. no vamos a esperar, vamos a synchronization hacemos tap y ahora si, apenas refrescamos
ya vemos los cambios.

Bien, con esta pequeña demo pudimos ver como configurar algunas propiedades del objeto

Offline Database y además vimos como es posible utilizar la API Syncrhonization para realizar las operaciones de envío y recepción a pedido del usuario.

Con esto terminamos el tema.

GeneXus™

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