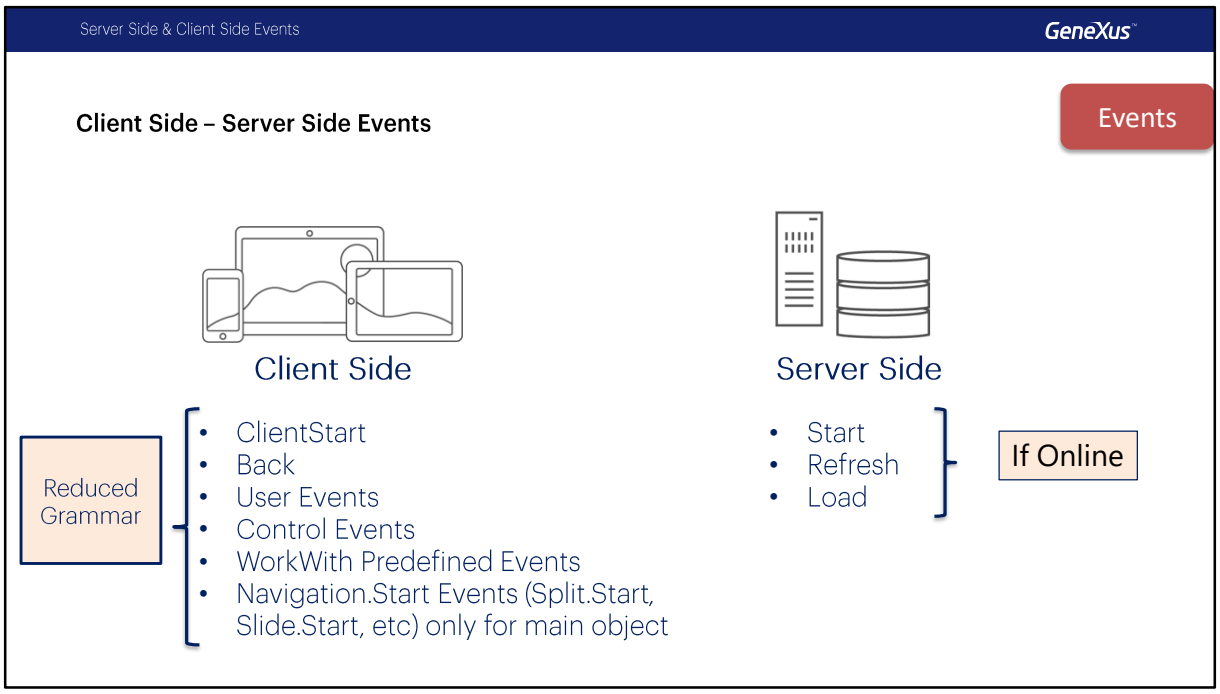

GeneXus™
The power of doing.

Server Side & Client Side Events

Behavior Development

GeneXus™ 16

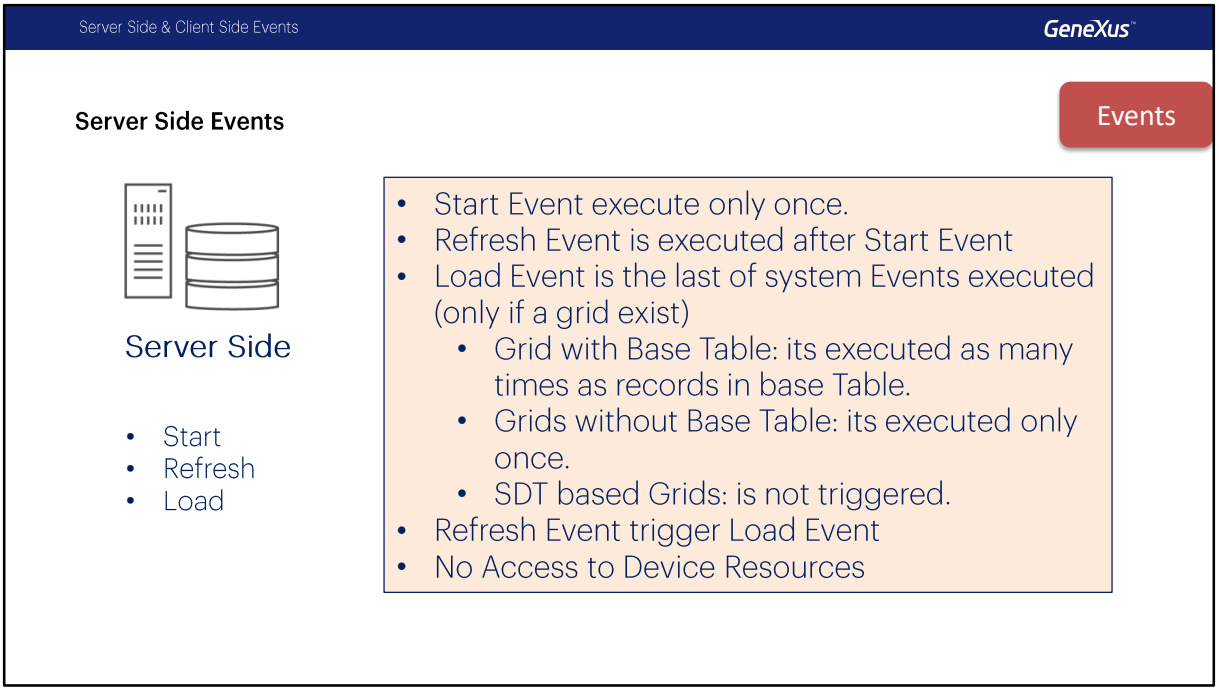


En este video, empezaremos a estudiar los eventos que podemos definir a nivel de los objetos Smart Devices, en el contexto de una aplicación Online, destacando los puntos en los que una aplicación Offline difiere de lo aquí estudiado.

Las aplicaciones para Smart Devices incluyen dos tipos de Eventos, los eventos del lado del cliente y los eventos del lado del servidor.
Si tienen experiencia en el desarrollo de aplicaciones para plataforma Windows o Web con GeneXus, ya estarán familiarizados con los eventos del lado del servidor: Start, Refresh Load.

Los eventos del lado del cliente incluyen los eventos del sistema ClientStart y Back, los eventos del usuario, eventos asociados a controles y algunos predefinidos por el patrón WorkWith como así también los eventos que estudiamos cuando vimos los estilos de navegación como Slide.Start por ejemplo.

Esta clasificación es importante puesto que necesitamos saber dónde se ejecuta cada evento, ya que dependiendo de eso, los recursos que podamos usar variarán, y también las restricciones aplicables.
También hay algunas diferencias con respecto a la gramática que debemos considerar.

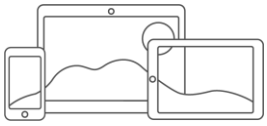


En estos eventos (Start, Refresh y Load) podemos utilizar todos los comandos aceptados por GeneXus.

- El evento Start se ejecutara solo una vez, cuando se abre el panel y no se ejecutara nuevamente a menos que salgamos del objeto y volvamos a ingresar en él.
- El evento Refresh se ejecuta luego del Evento Start, normalmente una sola vez, pero puede ser invocado nuevamente por medio del comando Refresh, en ese caso se ejecutara mas de una vez y será el primer evento ya que Start ya no se volverá a ejecutar.
- El evento load es el ultimo de los eventos del sistema a ser ejecutados y
    - Si tiene Tabla Base se ejecutará tantas veces como registros existan en la tabla base.
    - Si la grilla no tiene tabla base se ejecutará solo una vez
    - Y si la grilla esta basada en un SDT no se ejecutará el evento load.
- Cuando se invoque al evento Refresh al finalizar se disparará el evento Load (si corresponde)
- Estos eventos No tienen acceso a los recursos del dispositivo, por ejemplo la cámara, el GPS, etc.

Events

Client Side Events



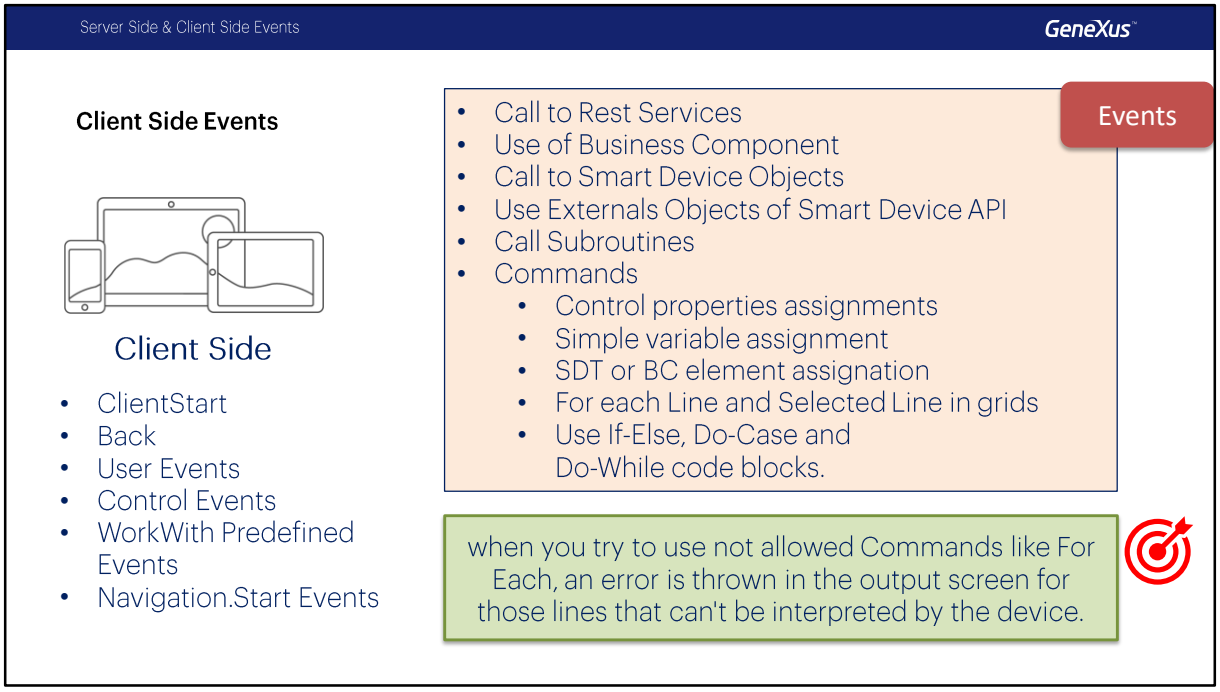
Client Side

- ClientStart
- Back
- User Events
- Control Events
- WorkWith Predefined Events
- Navigation.Start Events

- Types:
  - System Events: ClientStart, Back
  - User Events: User Defined
  - Control Events: Predefined for each control
- Access to Device resources
- Call to Rest Services to Access Server
- Do not trigger Server Side Events (unless Refresh command is used)
- Use different Grammar

Los eventos del lado del Cliente son la respuesta de la aplicación a la interacción del usuario.

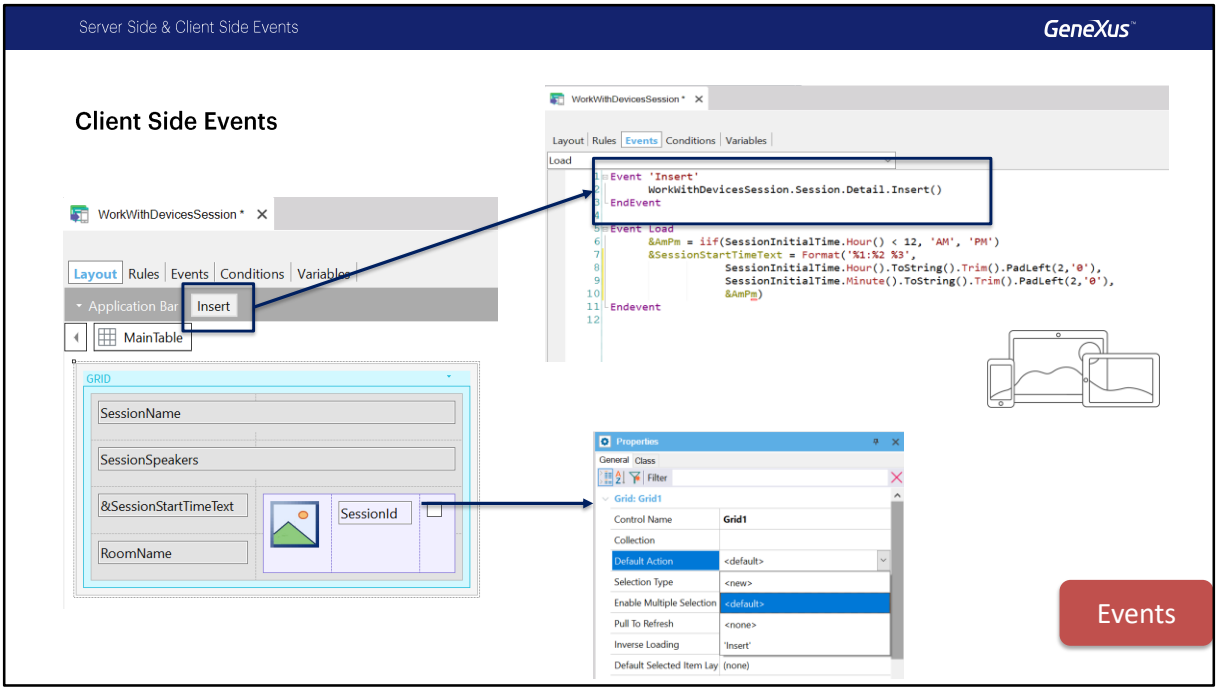
- Hay tres tipos de eventos de cliente: del sistema, de usuario y eventos de controles.
- A diferencia de los eventos Start, Refresh y Load que se ejecutan completamente en el servidor los eventos del lado del cliente tienen acceso a todos los recursos de nuestro dispositivo.
- El código asociado a estos eventos se ejecuta en el dispositivo, a menos que se requiera acceder al servidor (por ejemplo, cuando se debe invoca un procedimiento, en este caso el proceso será ejecutado siempre en el servidor por medio de una invocación a un servicio Rest, esto es transparente para el desarrollador).
- Durante la ejecución de un evento del lado del cliente, los eventos del sistema no se ejecutan a menos que se requieran explícitamente a través del comando Refresh.
- Como ya mencionamos estos eventos tendrán una gramática particular diferente a los eventos del lado del servidor. Ya trataremos en detalle este tema en otro video.



Que podemos hacer en un evento del lado del cliente?

- Podemos llamar a Servicios Rest: automáticamente cuando llamemos a procesos o Data Providers, estos serán expuestos como servicios Rest.
- Usar Business Components para recuperar o actualizar información, en este caso también esos Business Components serán expuestos como un servicio Rest en forma automática.
- Llamar a cualquier otro objeto para Smart Devices como nodos del WorkWith, Panels o Menu for Smart Devices.
- Podemos utilizar los Objetos Externos del Smart Device API
- Podemos llamar a subrutinas
- Y podemos utilizar los siguientes comandos
    - Asignación de propiedades a controles
    - Asignación de variables simples
    - Usos de variables SDT o Business Component
    - Ejecución de For Eeach Line y For each Selected Line en grillas
    - Uso de los bloques IF-Else, Do-Case y Do-While

En caso de que intentemos utilizar comandos no permitidos, veremos un error en la pantalla de salida para aquellas líneas que el dispositivo no puede interpretar.

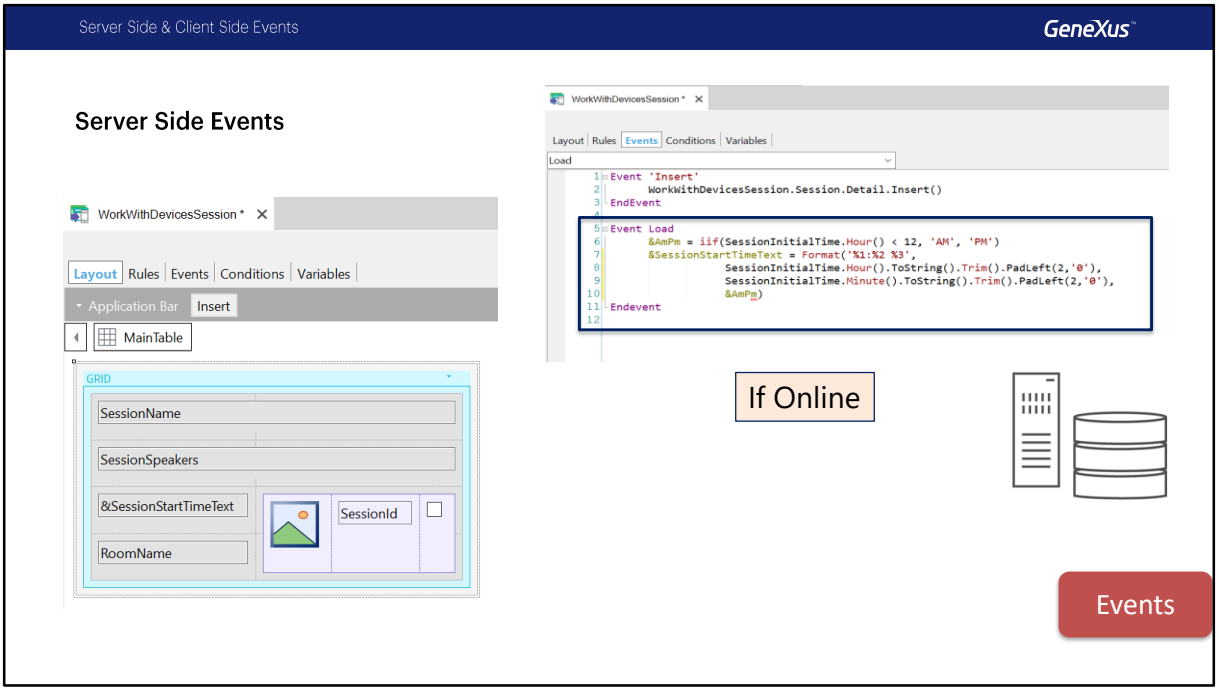


Veamos algunos ejemplos de eventos del lado del Cliente.

Por ejemplo, el evento Insert en el List es un evento creado por el patrón.

Otro evento que se ejecuta del lado del cliente ocurre cuando hacemos TAP sobre un ítem de la grilla, esto dispara un evento que aunque no lo vemos programado en los eventos (o sea que esta implícito en forma predeterminada), tiene un Call al Detail de ese WorkWith en modo Display, si quisiéramos podríamos cambiarlo y aun así seguiría siendo un evento del lado del cliente, ya que esta asociado al Tap sobre un control, en este caso la grilla.

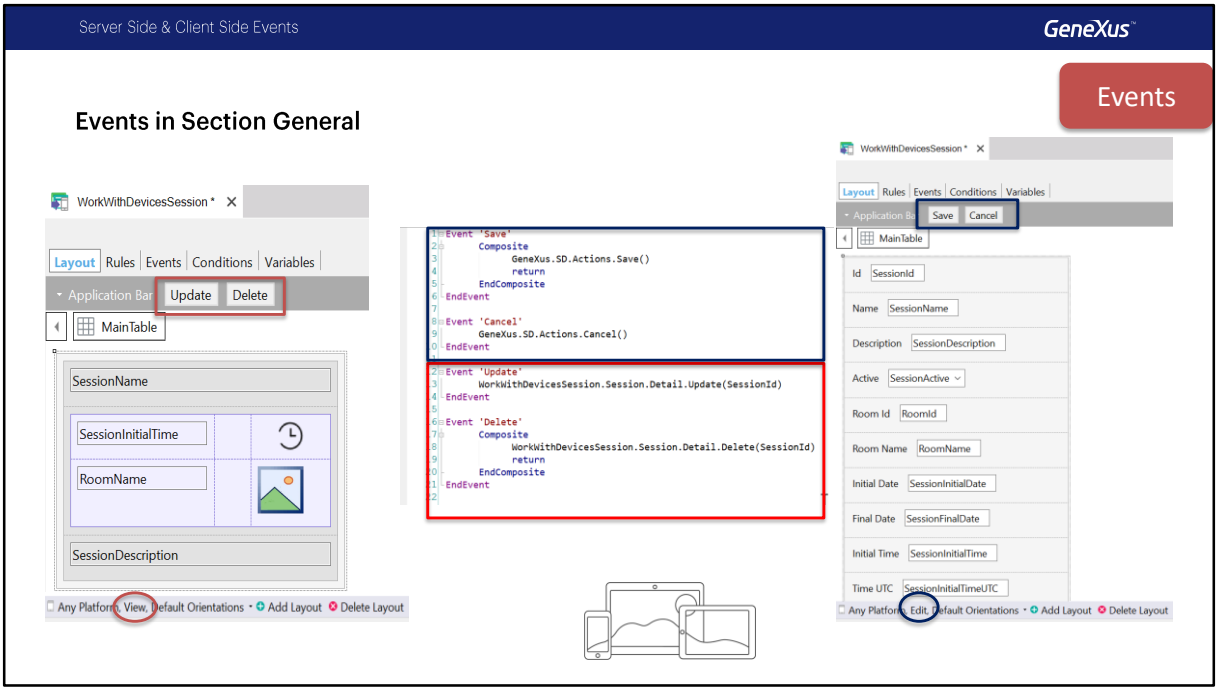
Todos estos eventos son del cliente, aunque invoquen a objetos que deban llamar a servicios (o data providers) del server para devolver los datos o grabarlos.



Veamos que pasa del lado del servidor,

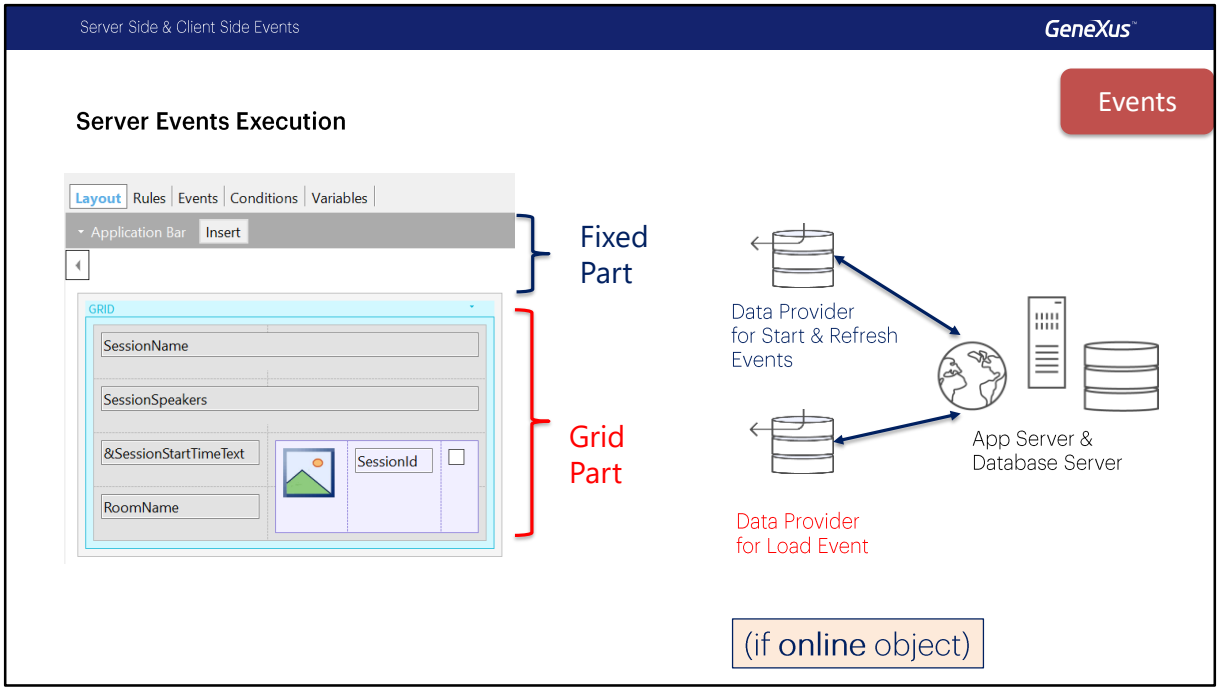
El evento Load, para cargar las líneas en el grid, en este ejemplo se programo para que en base al valor de la hora de inicio se cargue una variable que contendrá la hora de inicio y el indicador de AM o PM según corresponda.

Este evento será ejecutado en el server (si el objeto se ejecuta Online).



Aquí vemos los layouts que creó el pattern para la Section (General) y los eventos default. A la Izquierda vemos el layout en modo View y a la Derecha en modo Edit.

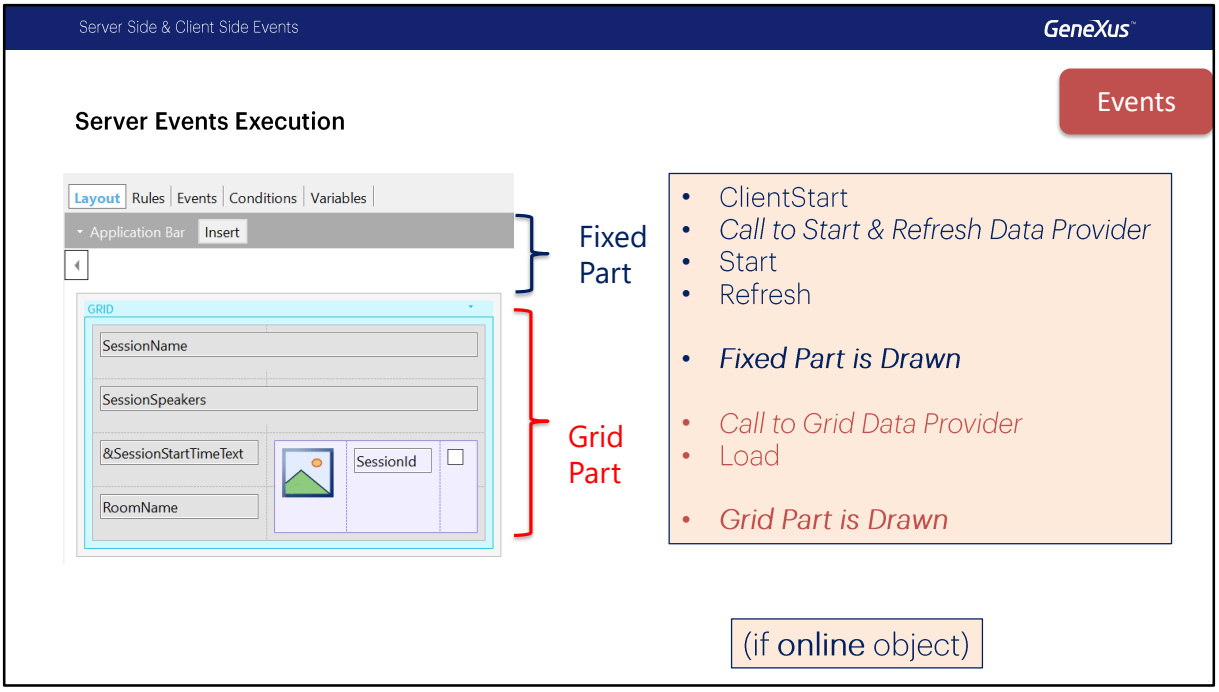
Son cuatro eventos en total: los dos primeros Save y Cancel aplican al modo Edit, y los otros dos, Update y Delete al modo View. Todos Estos también serán eventos del cliente.



En los objetos para Smart Devices podemos identificar dos partes distintas, la primera parte a la que denominaremos la parte fija o plana contendrá todo aquello que este en el formulario y que no este incluido en ningún grid, en el ejemplo que vemos en pantalla, la parte fija esta compuesta por el Application Bar con el botón Insert, si tuviéramos atributos fuera del grid estos serian de la parte fija.
La segunda parte la denominaremos parte del grid o variable, y estará compuesta en este caso por el grid que contiene los datos de la Conferencia.

Siempre tendremos una parte fija, y además tendremos una parte Variable por cada una de las grillas que tenga el panel, con lo cual esta parte podrá o no aparecer en nuestros objetos, de acuerdo a si contamos con un grid en el objeto o no, en este caso tendremos una parte Grid.

Para cada parte (fija y grid) GeneXus va a generar automáticamente Data Providers que resolverán el acceso a datos, estos Data Providers serán invocados con el protocolo Rest, no los veremos en la Base de Conocimientos ya que GeneXus se encargara de generarlos y mantenerlos, pero si podremos ver que datos acceden cada uno como veremos en un momento.



Ahora veamos conceptualmente que es lo que ocurre cuando ejecutamos por primera vez un panel para Smart Devices:

Primero se ejecuta el evento ClientStart, se ejecuta solo una vez y corre como ya vimos en el dispositivo, luego se ejecuta un data provider por medio de una llamada Rest que resolverá el acceso a datos necesario para la parte fija y se recuperan todos los atributos que sean necesarios, a continuación con el resultado retornado por el servicio se ejecutara el evento Start y luego el evento Refresh en el lado del servidor, luego se dibujara la parte fija del layout.

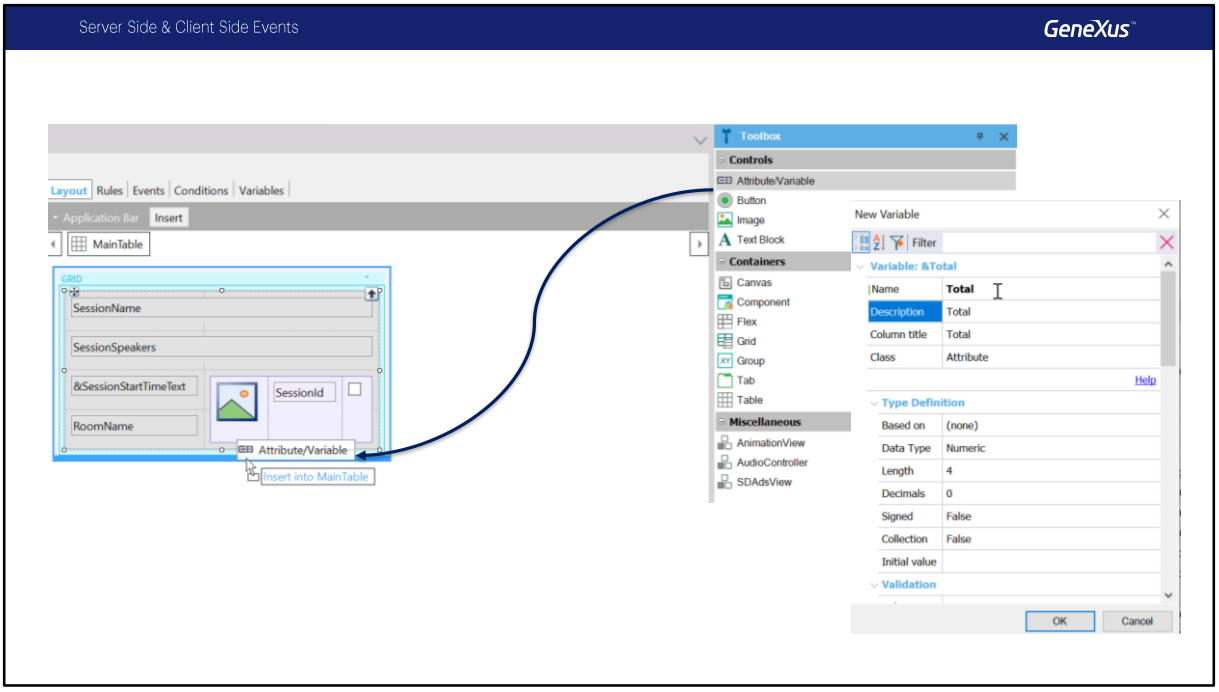
Luego se ejecuta un segundo Data Provider también llamado como un servicio rest, que resolverá la recuperación de los datos requeridos por la grilla y a continuación se ejecuta el evento Load, este evento se ejecutara N veces cuando la grilla tenga tabla base, una vez por cada registro.

En este momento se terminara de dibujar el layout con la parte del grid.

La particularidad de que la pantalla sea dibujada en dos momentos distintos tiene sus consecuencias como veremos a continuación.

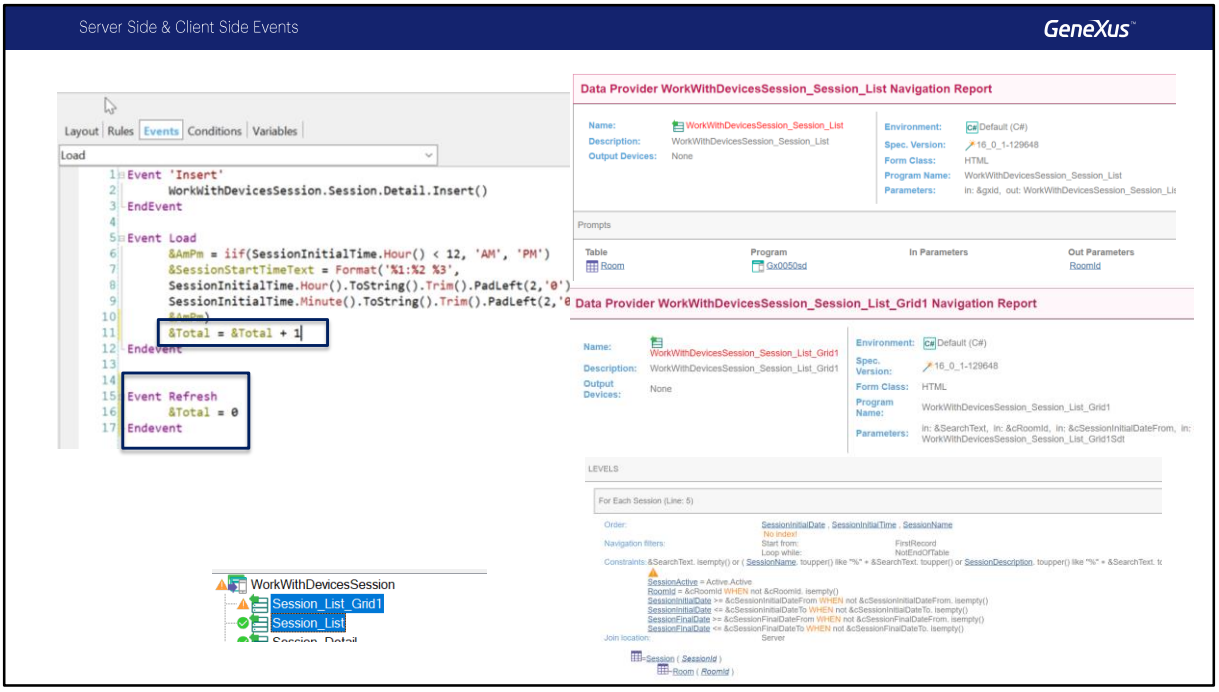
Demo: Server Events Execution

Vamos a ver un ejemplo en GeneXus



Supongamos que queremos mostrar en el list de Sessions la cantidad de conferencias,
si programamos este panel de la misma manera que un web panel pondríamos una variable abajo del grid ,
una variable total, vamos a crearla.

En esta variable vamos a mostrar la cantidad total de conferencias.



Vamos entonces a los eventos y lo primero que haríamos es inicializarlo en el evento refresh, vamos entonces al evento refresh y escribimos &Total = 0 . De esta manera inicializamos el valor.

En el evento load vamos a incrementar el valor por cada registro . Dijimos que el evento load se ejecuta N veces, entonces suena lógico incrementar aquí el valor de la variable &Total.

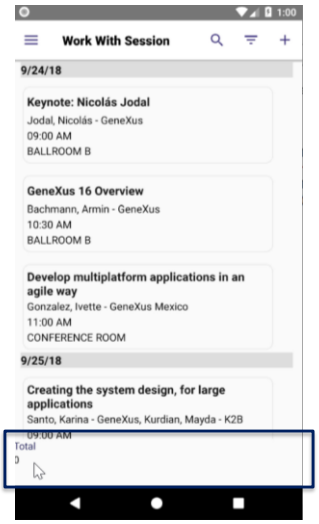
Vamos a ejecutar la aplicación.

Mientras se genera vamos a ver el Listado de Navegación:

Observemos que para el List tenemos dos Data Providers, el primero que vemos es el asociado al Grid1, el nombre es WorkWithDevicesSession_Session_List_Grid1, este Data Provider accede a la Tabla Session y a la tabla Room para resolver los datos del grid. Acá van a poder ver los filtros que se aplican con todas las condiciones correspondientes.

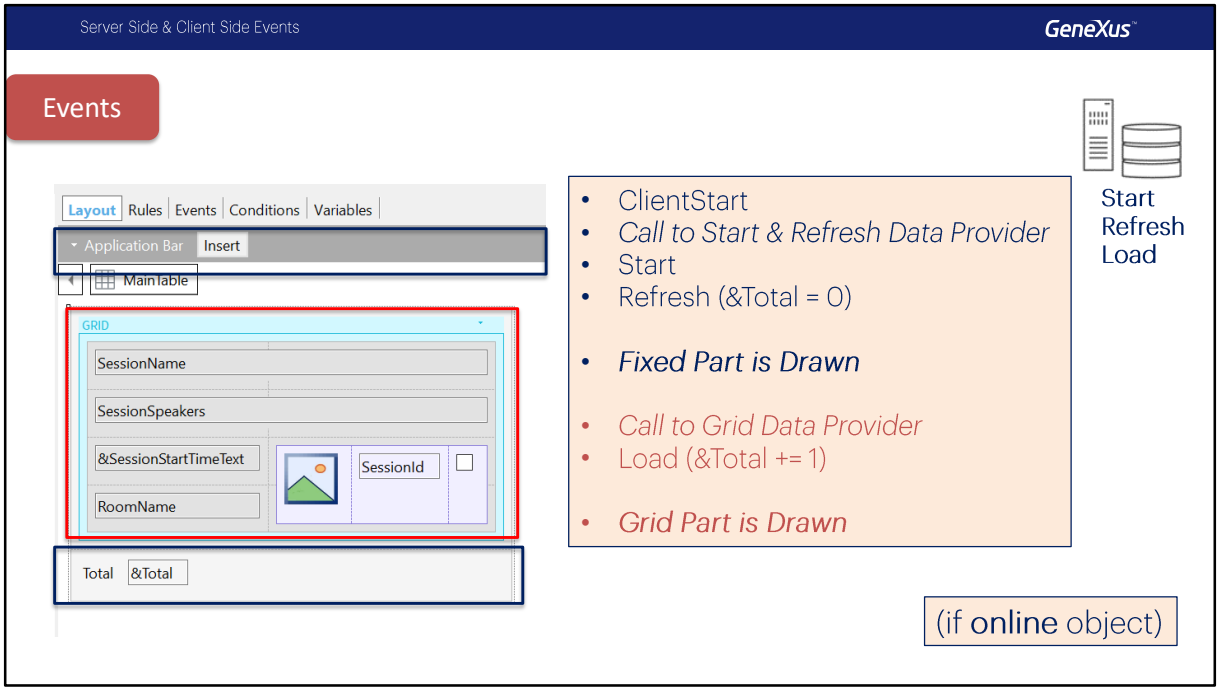
El que sigue es el que se genero para la parte Fija, su nombre es WorkWithDevicesSession_Session_List , noten que este data provider no esta accediendo a

ninguna tabla, ya que no tenemos en el panel ningún atributo que intervenga en la parte fija (ni en el grid, ni en los eventos Start o Refresh).



Ahora si , accedamos al emulador.

Bien, vamos a sessions.
Vean que ya me mostro la variable &Total y tiene valor cero, luego se cargaron los registros del grid.
Porque sucede esto?



Aquí podemos ver esquemáticamente lo que hicimos.

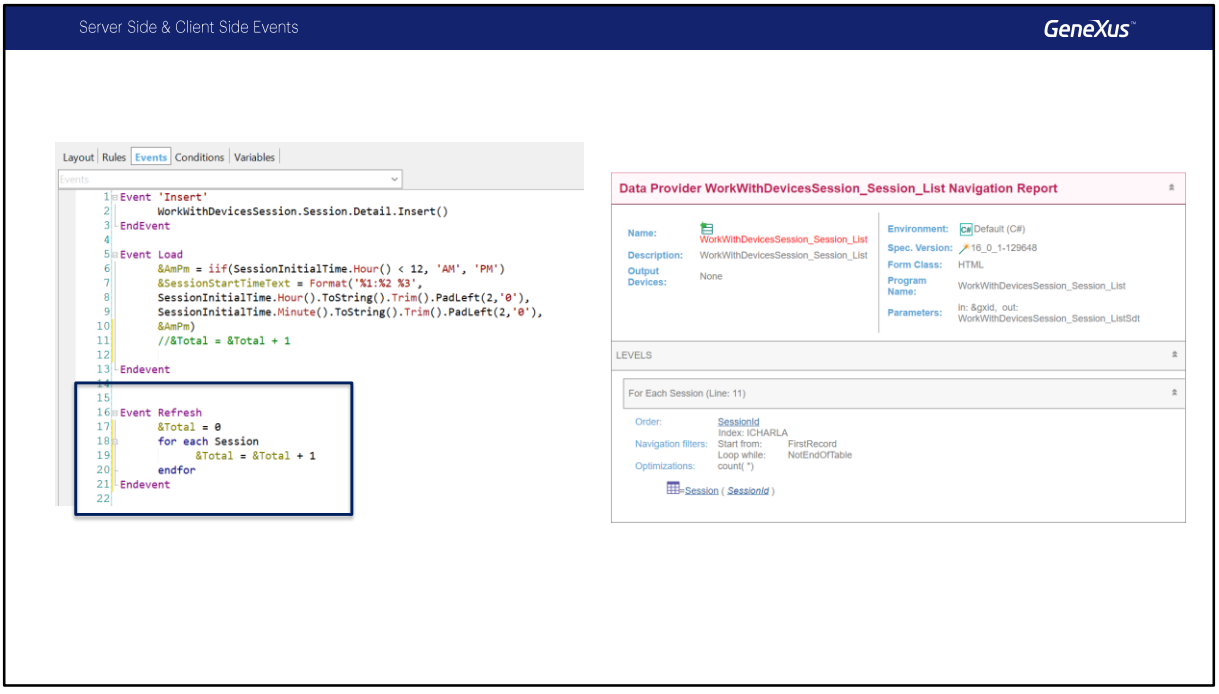
Si recordamos lo que vimos antes, al abrir el panel, se ejecutó el Evento ClientStart, en ese momento se llamó al Data Provider de la parte fija que en nuestro caso no recupera nada del servidor, luego se ejecutaron los eventos Start y luego Refresh donde se inicializó la variable &Total, a continuación se dibujó la parte fija, noten que la variable &Total en este punto valía cero.

Luego se llamo al Data Provider asociado al Grid, y se ejecutó el evento Load, aquí nosotros incrementamos el valor de &Total, pero la variable &Total ya estaba dibujada en el Layout con valor cero. Luego se dibuja el grid.

Obsérvese que esa es la razón del problema, el orden en el que se dibujan las distintas partes del objeto, no es el hecho de que los eventos Refresh y Load estén separados en dos programas (dos data providers independientes).

Demo: Server Events Execution

Volvamos a GeneXus.

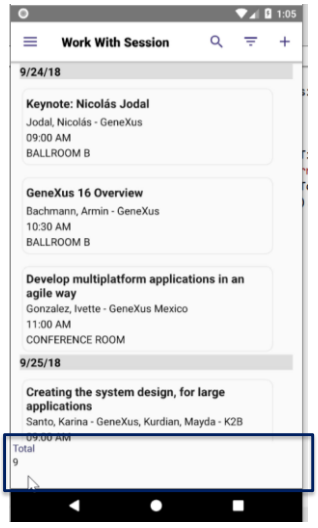


Por tanto una solución para conseguir mostrar el numero de registros correcto es dentro del evento refresh programar un for each que los calcule.

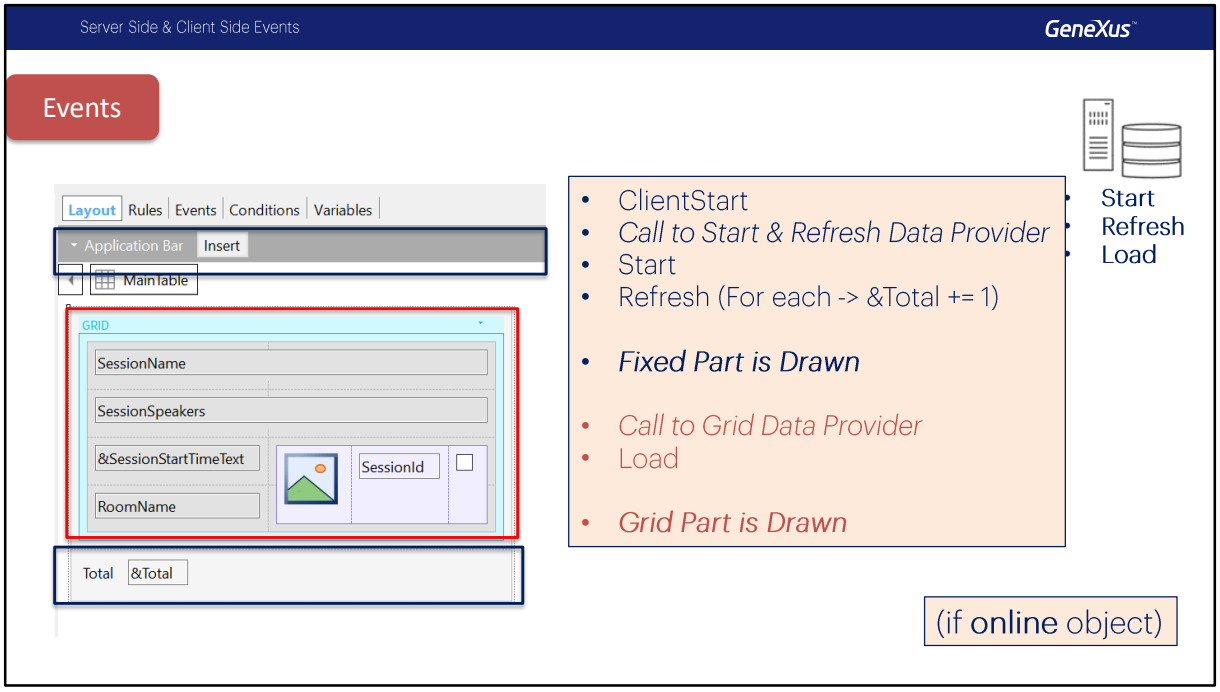
For each que tendrá como tabla base la tabla asociada al primer nivel de la transacción sesión.
Vamos a comentar entonces del Load la asignación y la vamos a poner dentro del for each.

Ahora si, vamos a correr nuevamente la aplicación y veremos que es lo que ocurre.

Veamos los listados de navegación ahora:
El asociado al Grid1 sigue igual, en cambio el asociado a la parte fija, ahora accede a Session y realiza un count(*) sobre la tabla.



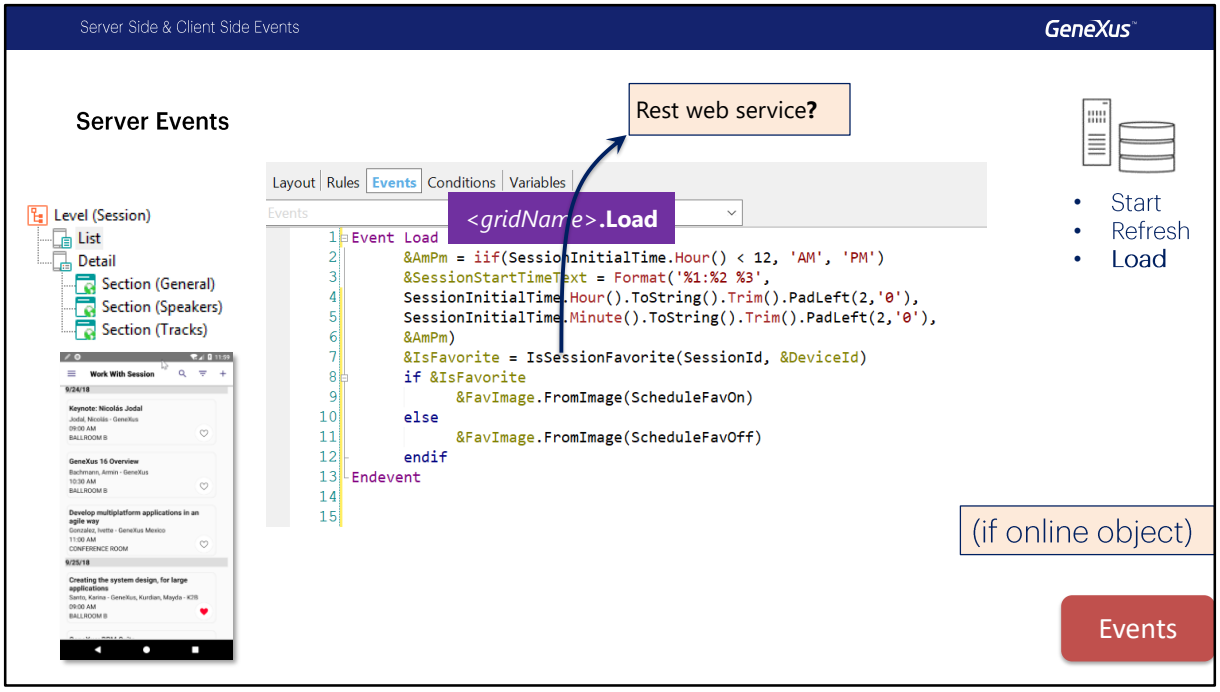
Vamos al emulador, entremos a sessions nuevamente.
Y ahora si luego de que se refresca la pantalla tenemos el total con los 9 registros que teníamos en la tabla, lo cual es correcto.



Vamos a ver que es lo que sucedió ahora:

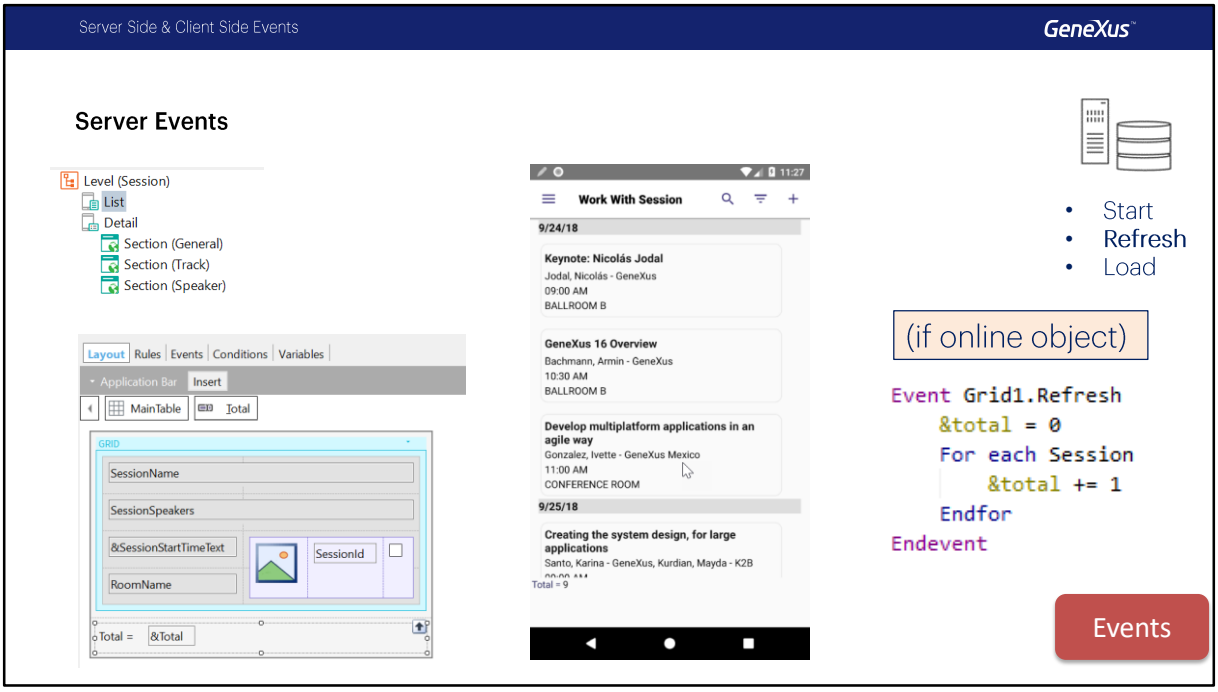
Se ejecutó ClientStart, en este momento se llamo al Data Provider que resolvió la parte fija que recuperó el resultado del For Each sobre la tabla Session donde hizo el Count, luego se ejecutó Start y luego Refresh donde se incrementó la variable &Total, esto se optimiza y directamente va a tener el resultado del count de la tabla porque se lo retornó el Data Provider que vimos, a continuación se dibujó la parte fija, noten que ahora la variable &Total vale nueve.

El resto siguió igual, se llamó al Data Provider asociado al Grid, y se ejecutó el evento Load n veces. Luego se dibujó el grid.



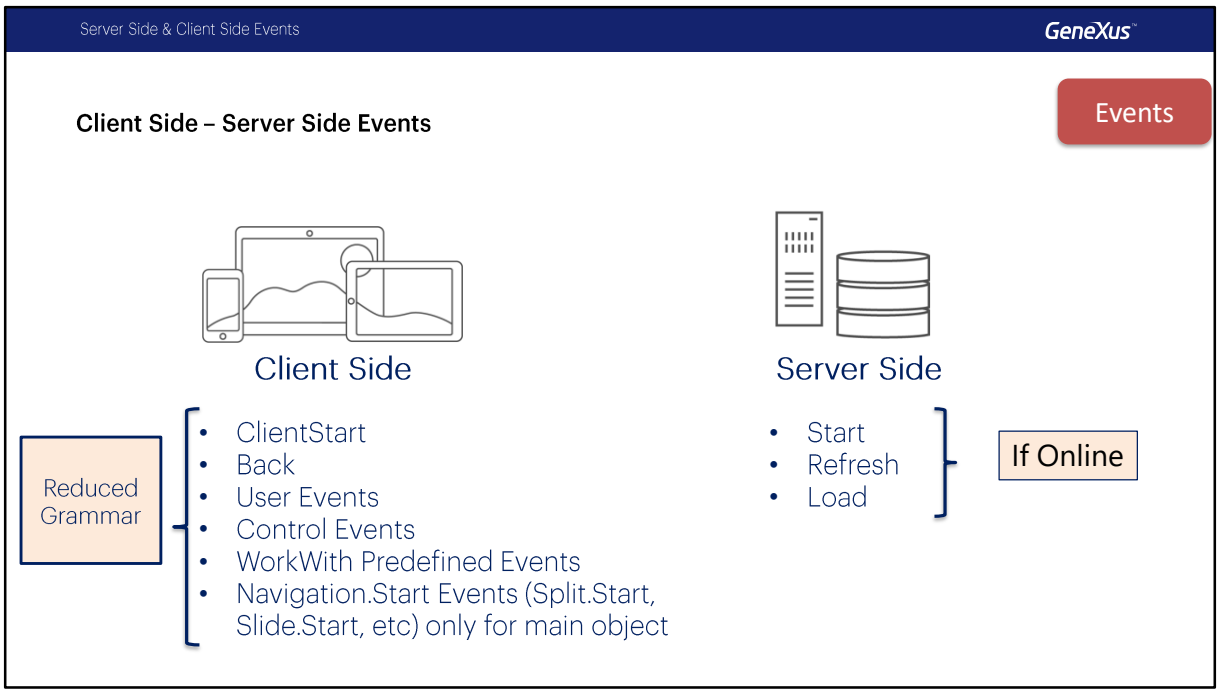
El evento **Load**, del sistema, no requiere ninguna consideración especial, puesto que es análogo al Load de un Web Panel. Como se pueden incluir varios grids con tabla base en el mismo layout, al igual que con web panels cuando tenemos más de un grid será necesario especificar el Load de qué grid estamos programando, escribiendo nombre de grid *punto* **Load**.

Aquí mostramos el ejemplo del Load para cargar los ítems del grid del List del WorkWithDevicesSession.
Este evento se ejecuta en el servidor, por lo que tiene a disposición todos los atributos de la tabla extendida, para ser utilizados.
Observe que el procedimiento IsSessionFavorite no tiene por qué estar expuesto como servicio Rest, dado que se está invocando desde el propio server.



El evento Refresh, del sistema, al igual que el Load tampoco requiere ninguna consideración especial, ya que es análogo al Refresh de un Web Panel.
También es válido, lo mismo que con web panels cuando tenemos más de un grid, es decir, será necesario en ese caso especificar el Refresh de qué grid estamos programando, escribiendo *<gridName>***.Refresh**.

Este evento, al igual que el Load, se ejecuta en el servidor, por lo que tiene a disposición todos los atributos de la tabla extendida, para ser utilizados.



Hemos visto entonces los eventos que se ejecutan en el cliente y los que se ejecutan en el servidor.
Es impotante diferenciarlos, dado que lo que podemos programar en los eventos del cliente sigue una gramática un poco más reducida que lo que podemos hacer en el servidor, como veremos específicamente en un video aparte.

Adelantamos que esto no afectará a la implementación de una aplicación offline. Es decir, en cuanto a la gramática, usted programará los eventos de la misma manera si su aplicación es online u offline. En ese sentido es transparente.

En el video siguiente, continuaremos estudiando la navegación y determinación de la Tabla Base en los objetos para Smart Devices.

GeneXus™

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