

Objeto Web Panel. Carga de datos y eventos

GeneXus™

WWAttractionsFromScratch

```

1 Event Load
2   &trips = count(TripDate)
3   &totalTrips = &totalTrips + &trips
4 -Endevent
5
6
7 Event Refresh
8   &totalTrips = 0
9 -Endevent
10
11
12 Event Start
13   &update.FromImage(UpdateIcon)
14 -Endevent
15
16
17 Event &update.Click
18   Attraction(TrnMode.Update, AttractionId)
19 -Endevent
20

```

```

1 //Event Refresh
2 //   &totalTrips = 0
3 //Endevent
4
5 Event Start
6   &update.FromImage(updateIcon)
7   &newTrip = "New trip"
8 -Endevent
9
10 Event &update.Click
11   Attraction(trnMode.Update, AttractionId)
12 -Endevent
13
14 Event Grid1.Load
15   &trips = Count(TripDate)
16   &totalTrips = &totalTrips + &trips
17 -Endevent
18
19 Event Grid1.Refresh
20   &totalTrips = 0
21 -Endevent

```

Estábamos construyendo nuestro web panel WWAttractionsFromScratch. Habíamos visto cómo condicionar los datos que se mostraban en el grid, y cómo ordenarlos. El grid tenía tabla base. Habíamos visto en ese caso cómo cargar una variable del grid para cada línea con el evento Load. Habíamos utilizado el evento Refresh, así como el Start, y también habíamos programado un evento a nivel de las líneas, para llamar a la transacción Attraction en modo update.

Además habíamos visto que al tener un grid podíamos utilizar el evento Load del grid en lugar del genérico, para anticiparnos a la posibilidad futura de tener que insertar otro grid. Y también que teníamos un Refresh del grid. Por ahora, para no marear, seguiremos con los genéricos.

Country Id

Attraction Name From

Attraction Name To

GRID

Id	Attraction Name	Country	Photo	Trips	newTrip
AttractionId	AttractionName	CountryName		&trips	<input type="text" value="&newTrip"/>

Total Trips

```

1 Event Load
2   &trips = count( TripDate )
3   &totalTrips = &totalTrips + &trips
4 Endevent
5
6 Event Refresh
7   &totalTrips = 0
8 Endevent
9
10 Event Start
11   &update.FromImage(updateIcon)
12   &newTrip = "New Trip"
13 Endevent
14
15 Event &update.Click
16   Attraction( TrnMode.Update, AttractionId )
17 Endevent
18
19 Event &newTrip.Click
20   |
21 Endevent

```

Ahora agreguemos otra acción a nivel de las líneas, pero una que no llame a otro objeto con interfaz, como era el caso de la invocación a la transacción Attraction.

Por ejemplo, imaginemos que queremos dar la posibilidad de que desde una línea (una atracción) se pueda crear un nuevo trip en la base de datos, con esa atracción.

Agreguemos primeramente una nueva variable al grid, de nombre newTrip, caracter de 10.

Cambiémosla a ReadOnly. Querremos que contenga el texto "New Trip", así que se lo asignamos en el evento Start, puesto que no va a variar por línea. Y programemos para esa variable el evento click.

¿Qué es lo que queremos hacer cuando el usuario haga clic sobre New Trip?

```

1 param( in: &AttractionId, out: &trips );
2 |

```

```

1 new
2   TripDate = Today()
3   TripDescription = "Created automatically from WWAttractionsFromScratch"
4 endnew
5 &TripId = TripId
6 new
7   TripId = &TripId
8   AttractionId = &AttractionId
9 endnew
10 &trips = Count( TripDate, AttractionId = &AttractionId )
11
12 |

```

WWAttractionsFromScratch:

```

Event &newTrip.Click
  &trips = NewTrip(AttractionId)
Endevent

```

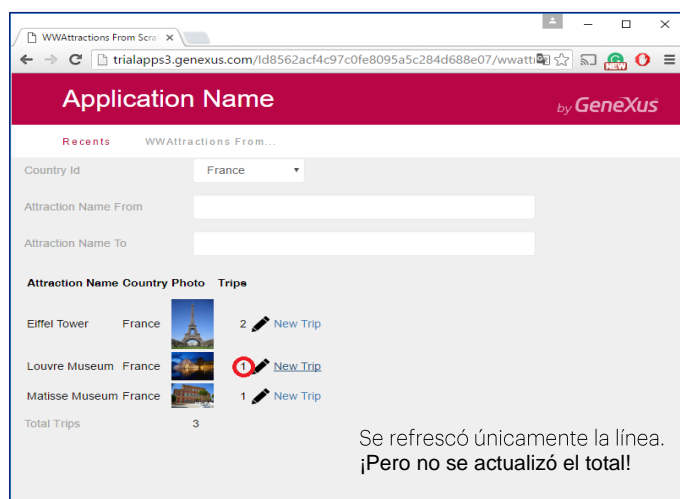
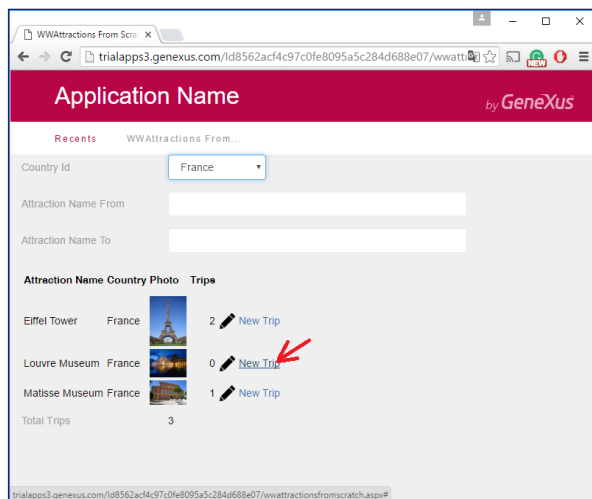
Por ejemplo, llamar a un procedimiento al que le pasamos el identificador de la atracción de la línea y crea el trip con esa atracción.

Observar que lo hemos implementado con el comando **new**, para crear directamente un registro en la tabla Trip y uno en la tabla TripAttraction. Usamos esta solución para mostrar el uso de estos comandos. Sin embargo la solución más aconsejable sería utilizar el business component de Trip para insertar. En este curso no vimos cómo cargar un business component de dos niveles, pero es muy sencillo.

Veamos que luego de insertar cabezal y línea de Trip, calculamos la cantidad de trips en los que esa atracción se encuentra. Como la fórmula inline se está disparando sin que se esté posicionado en tabla alguna (está "suelta" en el código), tenemos que indicar la condición explícita de que queremos filtrar los trips de la atracción.

Y ese valor es el que se le devuelve a quien llama a este procedimiento. Entonces, desde el evento click del control (variable) &newTrip invocamos a este procedimiento, pasándole el AttractionId de la línea desde la que se hace el clic, y como el parámetro que devuelve es el que necesitamos mostrar en la columna &Trips, directamente se lo asignamos a la variable.

Lógicamente, cuando el usuario haga clic sobre New Trip deberá ver para esa línea, en la columna Trips, el valor que tenía antes del clic, más uno. Es decir, deberá refrescarse la línea.



```

Event &newTrip.Click
    &trips = NewTrip(AttractionId)
Endevent

```

Ejecutemos para probar.

Por ejemplo, filtremos por Francia, y para la atracción museo Louvre, que por ahora no está en ningún trip, hagamos clic sobre New Trip. Vemos que automáticamente se refrescó la línea.

Ahora muestra cantidad de trips del Louvre: 1.

Sin embargo lo que no se refrescó fue la cuenta del total, que debería decir 4, y sin embargo conserva el valor anterior. ¿Por qué?

Es que al ejecutar el evento click asociado a la variable &NewTrip, solamente se ejecutó su código, y como dentro del mismo se asignaba valor a una variable del grid, se refrescó la línea, **esa** línea, únicamente. Sin ejecutar ningún otro evento. Ninguno, ni siquiera el Load.

Comando Refresh

Alternativa 1

```

Event &newTrip.Click
  &trips = NewTrip( AttractionId )
  &totalTrips = &totalTrips + 1
Endevent

```

Alternativa 2

```

Event &newTrip.Click
  &trips = NewTrip( AttractionId )
  Refresh
Endevent

Event Load
  &trips = count( TripDate )
  &totalTrips = &totalTrips + &trips
Endevent

Event Refresh
  &totalTrips = 0
Endevent

```

Por lo tanto, tenemos dos alternativas para mantener actualizado el Total de Trips cargados en el grid cuando se produce este evento.

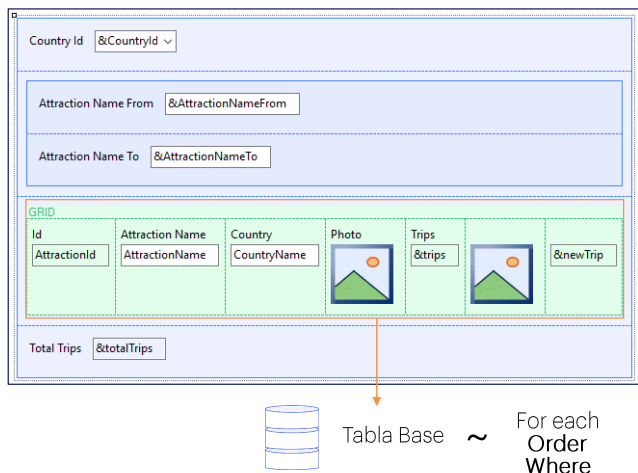
Una posibilidad es a &totalTrips sumarle uno, puesto que el procedimiento solamente ha agregado un trip para esa atracción.

Pero si luego el procedimiento cambia y se agregan más trips, tendremos que recordar hacer el cambio en forma consistente en este evento.

Una mejor solución parecería ser poder pedirle al web panel que vuelva a ejecutar los eventos Refresh y Load. Para ello, contamos con el comando Refresh. Al hacer esto, se volverá a ir a la base de datos a cargar el grid.

Resumen

Web Panel con grid (con atributos)



1ra vez

Start

Refresh

Load (N veces)

N veces

User / Control Event

Ejecutemos para probar:

Agreguemos un trip para el museo Matisse:

Vemos que ahora actualizó todo. Es que volvió a ejecutar los eventos Refresh y Load. Este último se ejecutó 7 veces, una vez por cada línea a ser cargada.

Hagamos un repaso de todo lo visto.

Para el caso de un web panel con un grid con atributos, GeneXus entiende que ese grid tiene una tabla base asociada, es decir una tabla que deberá navegar para cargar las líneas del grid. Cargará una línea por cada registro de esa tabla base. Para filtrar, tenemos la propiedad Conditions del grid, para ordenar, la propiedad Order. Un grid con tabla base es análogo a un for each.

En los web panels se producen eventos del sistema a los que les podemos programar código para que sea ejecutado en el momento en que se disparan. Vimos tres de estos eventos, que se disparan siempre que se abre un web panel es decir, en su primera ejecución:

El Start se dispara esta única vez. Allí pueden inicializarse variables, por ejemplo.

El Refresh se dispara antes de cargarse la información de la pantalla. Tras

este evento se producirá el acceso a la base de datos para traer los datos de la tabla base y su extendida.

Por cada registro de la tabla base que está por ser cargado en el grid, se produce un evento Load. Por tanto aquí será el momento de programar todas las acciones que queramos que se ejecuten antes de que la línea sea efectivamente cargada en el grid. Los datos que se cargan en el grid son exclusivamente los de las columnas visibles o invisibles que se hayan incluido en él.

Después de esto, el web panel estará cargado con la información que extrajo de la base de datos y se desconecta de ella.

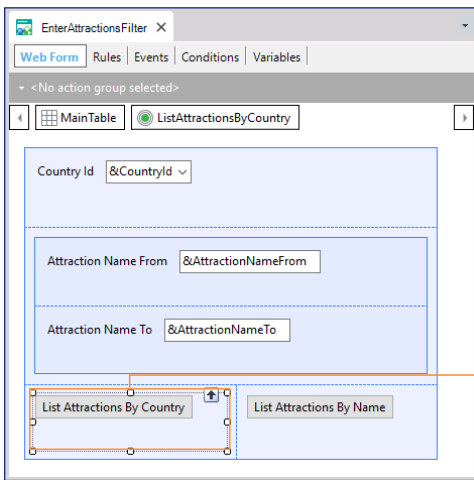
Ahora es cuando el usuario comienza a interactuar con la pantalla. Por ejemplo, modifica el valor de una de las variables que se utilizan para filtrar la información del Grid. Automáticamente se realiza un Refresh por el que otra vez se invoca al server para que ejecute esta vez el evento Refresh y el evento Load.

En nuestro caso, si CountryId corresponde al de Francia, entonces se ejecutará el Refresh y enseguida el Load para cada atracción de Francia. Observemos que la gran diferencia con la primera ejecución es que aquí el Start no se vuelve a ejecutar.

Pero los web panels permiten definir otros eventos, que se dispararán también luego de la primera vez, es decir, una vez que el web panel ya ha sido cargado, y siempre a partir de la acción del usuario.

Por ejemplo, el evento Click que programamos sobre esta imagen o el Click sobre New Trip,

Resumen



1ra vez

Start

Refresh

Load (N veces)

N veces:

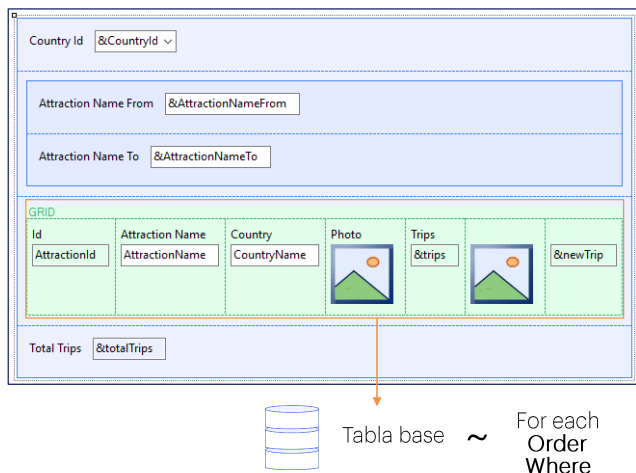
```

Event 'List Attractions By Country'
  AttractionsList(&CountryId)
  //AttractionsReport(&CountryId)
Endevent
  
```

o incluso en el web panel que habíamos definido muchas clases atrás, el evento de usuario que asociamos al botón, al que dimos este nombre

Resumen

- Web Panel con grid (con atributos)



1ra vez

Start

Refresh

Load (N veces)

N veces:

User / Control Event

Comando Refresh

Estos eventos son conocidos como **eventos de usuario**, o **eventos de los controles** (como el Click que vimos).

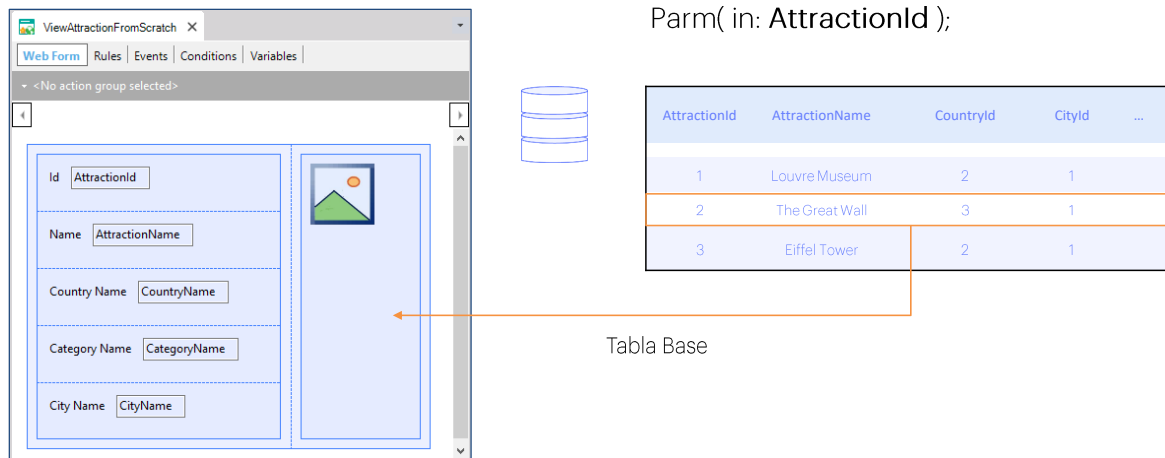
Cuando el usuario provoca un evento de estos, únicamente se ejecuta su código, sin refrescarse la pantalla. La única excepción es cuando el evento se produce a nivel de una línea del grid, como el caso que vimos de &newTrip, y dentro de su código se asigna valor a una variable del mismo grid, que en nuestro caso era &Trips. En ese caso, el valor de la variable se refresca en la pantalla, para esa línea, para que se muestre actualizada.

Si necesitamos que se vuelva a ejecutar el Refresh y se vuelvan a cargar las líneas en el grid a partir de la base de datos (por ejemplo para actualizar el total de las líneas), entonces podemos escribir el **comando Refresh** en el evento.

El comando Refresh provoca la ejecución de los eventos Refresh y Load.

Resumen y más

- Web Panel sin grid pero con atributos en el form



ViewAttractionFromScratch X

Web Form Rules Events Conditions Variables

<No action group selected>

Id

Name

Country Name

Category Name

City Name

Tabla Base

AttractionId	AttractionName	CountryId	CityId	...
1	Louvre Museum	2	1	
2	The Great Wall	3	1	
3	Eiffel Tower	2	1	

Parm(in: AttractionId);

Estudiamos el caso de un web panel con un grid con atributos. Pero, ¿qué pasa si lo que tenemos es un web panel sin grid pero con atributos en el form?

Supongamos que queremos que, haciendo clic sobre el nombre de atracción, en nuestro web panel WWAttractionsFromScratch... se llame a un web panel que muestre todos los datos de la atracción.

Para ello, ya hemos implementado este web panel... en cuyo form hemos insertado los atributos de una atracción que nos interesa mostrar. Además, hemos escrito una regla parm, para recibir un parámetro. Veamos que en vez de recibir en una variable, elegimos recibir en el atributo AttractionId.

Lo que deseamos es que cuando desde el evento click de AttractionName en nuestro otro web panel ... se llame a este Web panel y se le pase el id de atracción de la línea del grid, automáticamente GeneXus vaya a la tabla de atracciones, encuentre la atracción con ese id y para esa atracción, muestre la información de los atributos que se colocaron en el form.

En definitiva, un web panel que no tiene ningún grid pero que tiene atributos en el form también tendrá tabla base. ¿Cómo la determina GeneXus, siendo que ahora no tenemos transacción base para indicar? No lo veremos en este curso, pero es análogo al caso de un for each en el que no se indica Transacción Base.

Pero en este caso, como no tenemos un grid, se tendrá que cargar únicamente UN registro de esa tabla base y de su extendida. ¿Dónde indicamos el filtro que permita devolver ese registro?

En nuestro caso fue recibiendo en el atributo AttractionId. Allí estamos especificando el filtro automático, para que sepa con cuál de todos los registros de la tabla base debe quedarse.

Resumen y más

- Web Panel sin grid pero con atributos en el form



Base Table

Parm(in: &AttractionId);

AttractionId	AttractionName	CountryId	CityId	...
1	Louvre Museum	2	1	
2	The Great Wall	3	1	
3	Eiffel Tower	2	1	

Si necesitamos establecer otro tipo de condición, o, si por ejemplo hubiéramos recibido en variable, en lugar de en atributo, tenemos la solapa de Conditions para establecer el filtro.

Application Name

Recents

WWAttractions From... — View Attraction Fr...

Attraction Id	17
Attraction Name	Eiffel tower
Country Name	France
Category Name	Monument
City Name	Paris
	<input style="width: 100%;" type="text"/>

Event Load

Order: AttractionId

Index: IATTRACTION

Navigation filters: Start from: AttractionId = @AttractionId

Loop while: AttractionId = @AttractionId

Join location: Server

```

Attraction (AttractionId)
  -Country (CountryId)
    -CountryCity (CountryId, CityId)
      -Category (CategoryId)
        -count_TripDate_navigation
          -TripAttraction (AttractionId)
            -Trip (Incid)
          
```

```

1 | Event Load
2 |     &trips = Count(TripDate)
3 | Endevent
  
```

Si quisiéramos también mostrar aquí la cantidad de trips en los que se encuentra la atracción, necesitamos la variable &trips, pero esta vez, ¿dónde la cargamos? ¡En el evento Load! ¡También! Probemos.

Vale lo mismo que si hubiera un grid. Veamos lo que informa el listado de navegación. El evento Load tiene asociada la tabla Attraction, que es la tabla base. Pero en vez de recorrerla toda, solamente se queda con este registro.

```
1 parm(in:&AttractionId);
```

The screenshot shows the 'Event Load' window in GeneXus. On the left, there is a navigation tree with the following structure:

- Attraction (*AttractionId*)
 - Country (*CountryId*)
 - CountryCity (*CountryId, CityId*)
 - Category (*CategoryId*)
 - count(*TripDate*) navigation
 - TripAttraction (*AttractionId*)
 - Trip (*TripId*)

On the right, there is a data grid with the following fields:

Name	AttractionName
Country Id	CountryId
Country Name	CountryName
Category Name	CategoryName
City Name	CityName
Trips	&trips

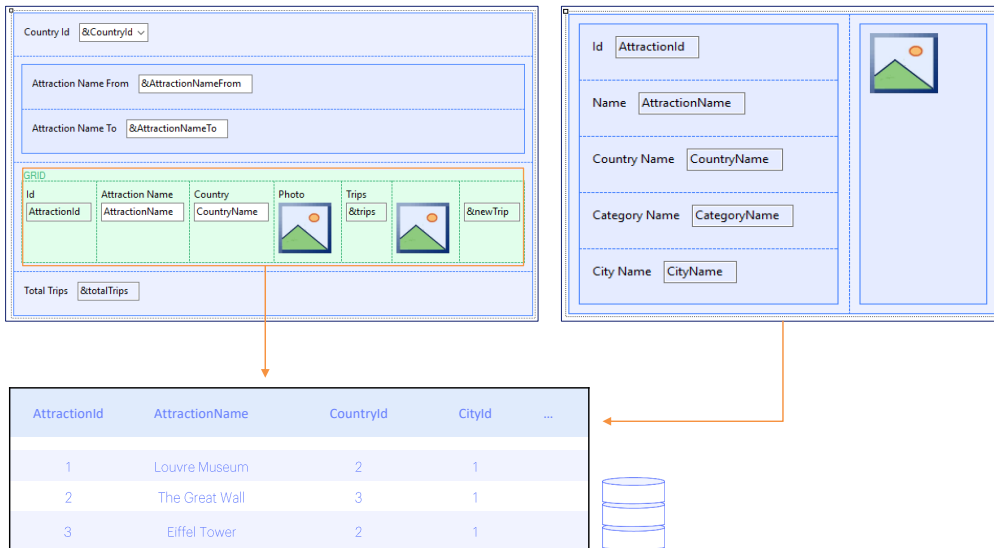
At the top of the grid, there is a small icon of a mountain and a sun.

Por ejemplo, ¿qué hubiera pasado si no hubiéramos recibido en el atributo, sino en variable, pero hubiéramos omitido especificar el filtro en las conditions? ¿Los valores de qué atracción cargará en la pantalla? Veamos el listado de navegación antes de ejecutar. Va a recorrer igual toda la tabla base y a hacer un Load por cada registro, pero como no hay un grid, estará sobrescribiendo por cada registro en el que se posicione, los atributos presentados en pantalla, y lo que vamos a ver finalmente va a ser el último registro de la recorrida, que como es realizada por clave primaria, corresponde a la última atracción ingresada. Veamos.

Forbidden city era la última atracción ingresada.

Si ahora colocamos la condición de filtro, y vemos el listado de navegación vemos que pasará otra vez a filtrar quedándose con una única atracción. Ejecutemos.

Web panels con tabla base



Hemos visto hasta aquí dos web panels que tienen tabla base:

- el primero con un grid. Allí la tabla base del grid es la tabla base del web panel, es decir, la tabla a la que el web panel automáticamente decide ir a navegar para traer la información a cargar en la pantalla.
- el segundo sin grid, pero con atributos. Allí la tabla base del web panel se encuentra a partir de esos atributos.

Pero tenemos también el caso mixto, que es cuando tenemos un web panel con grid con atributos, pero donde también hay atributos sueltos en lo que llamamos parte fija del panel, es decir, fuera de todo grid.

Por ejemplo, supongamos que al hacer click sobre el nombre del país de la atracción turística queremos visualizar la información general del país, y todas sus atracciones turísticas. Es parecido al panel que hicimos, pero donde aquí no estamos permitiendo filtrar por país, sino solo por nombre de atracción.

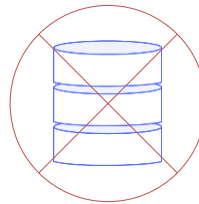
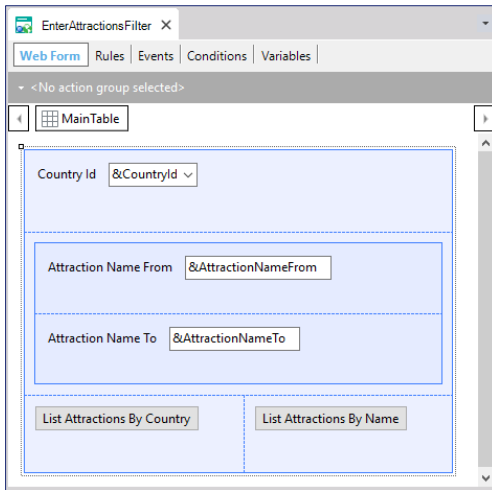
Si observamos cómo lo implementamos, desde el panel del View de la atracción programamos el evento click sobre el nombre de país (observe que tuvimos que colocar el atributo CountryId invisible para poder hacer esta invocación. ¿Por qué?) y allí llamamos a este nuevo Web Panel, que creamos salvando el que teníamos con este otro nombre. Como vemos, este nuevo web panel es muy parecido, salvo que recibe un parámetro, el identificador de país. En lugar de tener la variable CountryId, tiene el atributo CountryName. El orden del grid, que en el primero era condicional, aquí es fijo por CountryId, AttractionName porque el país vendrá instanciado, por parámetro. Y las condicionales para cargar el grid, que antes incluían una para filtrar por país, ya no lo hacen. Es que al recibirlo en el atributo, actuará como un filtro automático.

Si observamos el listado de navegación, vemos que sigue indicando que el evento Load para cargar la información a ser mostrada en la pantalla sigue recorriendo la misma tabla Attraction.

Es que pudimos sacar el atributo del grid y colocarlo en la parte fija, porque el país siempre va a ser el mismo, para todas las atracciones, dado que lo estábamos recibiendo por parámetro en el atributo. Podríamos

quitarlo del grid, donde no tendrá más sentido.

Web panels sin tabla base



Ahora veremos el caso de web panels sin tabla base, es decir, web panels que no tienen programada **en forma automática** una consulta a la base de datos.

El caso más obvio es cuando no se consulta en absoluto a la base de datos, como fue el caso de nuestro web panel de partida, el que únicamente pedía datos al usuario y llamaba a otros objetos.

Web panels sin tabla base

The screenshot shows a web panel layout with the following components:

- Country Id:
- Attraction Name From:
- Attraction Name To:
- GRID:

Attraction Id &AttractionId	Attraction Name &AttractionName	Country &CountryName	Photo	Trips &trips	&newTrip
- Total Trips:

An arrow points from the text "Solo variables!" to the variable labels in the grid.

```

Event Refresh
  &totalTrips = 0
Endevent

Event Start
  &update.FromImage(updateIcon)
  &newTrip = "New Trip"
Endevent

Event &update.Click
  Attraction( TrnMode.Update, &AttractionId )
Endevent

Event &newTrip.Click
  &trips = NewTrip( &AttractionId )
  Refresh
Endevent

Event &AttractionName.Click
  ViewAttractionFromScratch( &AttractionId )
Endevent
  
```

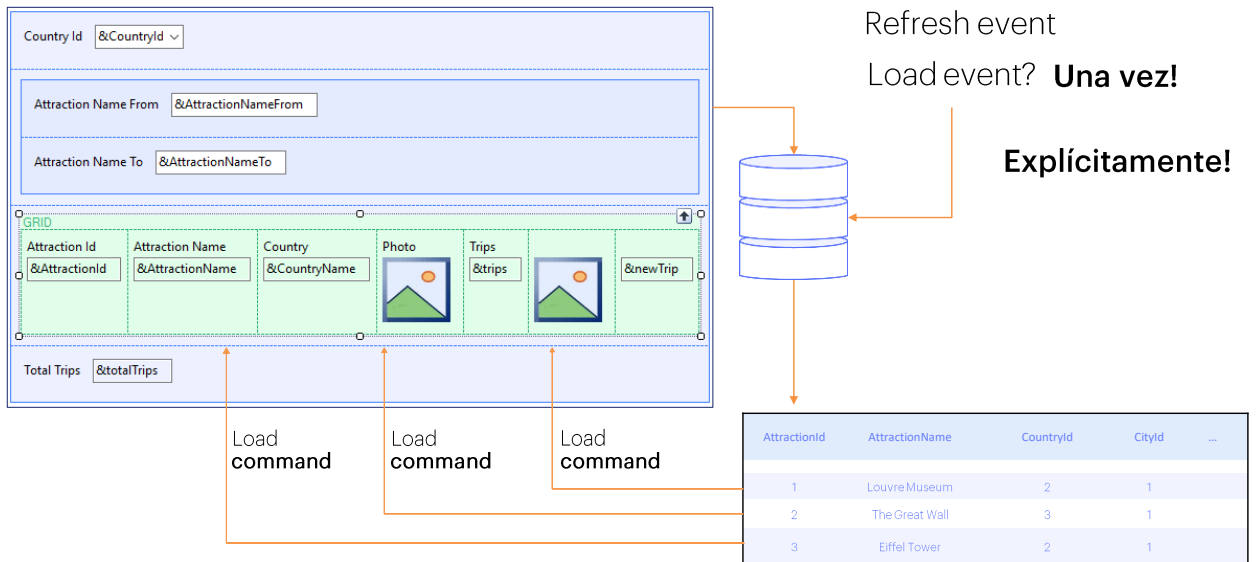
Pero también podemos tener un web panel que sí consulte la base de datos, solo que esa consulta y carga en la pantalla esté completamente en manos del desarrollador.

Los web panels que habíamos implementado con tabla base, bien podrían haberse implementado de este otro modo.

Veamos el caso del web panel que muestra en el grid las atracciones. Pero ahora en vez de tener atributos en el grid, tenemos variables.

Así, en los eventos tenemos que cambiar las invocaciones que teníamos, donde pasábamos el atributo AttracionId, por la variable &AttractionId.

Web panels sin tabla base



Ahora bien, si ahora sólo tenemos variables, ¿cómo sabe GeneXus que tiene que ir a navegar la tabla Attraction y su extendida para cargar una línea del grid por registro?

No lo sabe. La carga de este grid no será automática, como en el caso en que colocamos atributos.

Es decir, el evento Load no se ejecutará cuando ya se haya ido a navegar una tabla, por cada registro en el que se está posicionado en un momento dado. Es que ¡no se está posicionado en lugar alguno!

Sin embargo, el evento Load se va a disparar, sí. Solo que lo hará una única vez, después del Refresh y sin estar en la base de datos en absoluto. Es en ese disparo, en esa ejecución, en la que vamos a tener que programar la carga del grid en forma manual.

Es decir, vamos a tener que pedir explícitamente que se acceda a la base de datos (en nuestro caso, yendo a la tabla Attraction), y decirle cada vez, que cargue cada línea.

Para decirle que inserte una nueva línea en el grid, con los valores que en ese momento tengan las variables correspondientes a las columnas, se cuenta con el **comando Load**. El comando, **no** el evento. Toda vez que dentro del **evento Load**, se encuentre un **comando Load**, único lugar donde este comando está permitido, se insertará una línea en el grid.

Web panels sin tabla base

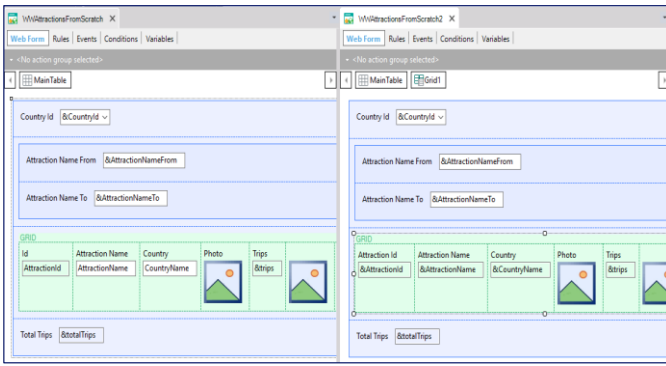
```

Event Load
For each Attraction
  order CountryId, AttractionName when not &CountryId.IsEmpty()
  order AttractionName
  where CountryId = &CountryId when not &CountryId.IsEmpty()
  where AttractionName >= &AttractionNameFrom when not &AttractionNameFrom.IsEmpty()
  where AttractionName <= &AttractionNameTo when not &AttractionNameTo.IsEmpty()
  &AttractionId = AttractionId
  &AttractionName = AttractionName
  &CountryName = CountryName
  &AttractionPhoto = AttractionPhoto
  &trips = count( TripDate )
  Load
  &totalTrips = &totalTrips + &trips
endfor
Endevent
  
```

The screenshot shows a grid control with the following columns: Attraction Id, Attraction Name, Country, Photo, Trips, and a button labeled &newTrip. The grid is currently empty.

AttractionId	AttractionName	CountryId	CityId	...
1	Louvre Museum	2	1	
2	The Great Wall	3	1	
3	Eiffel Tower	2	1	

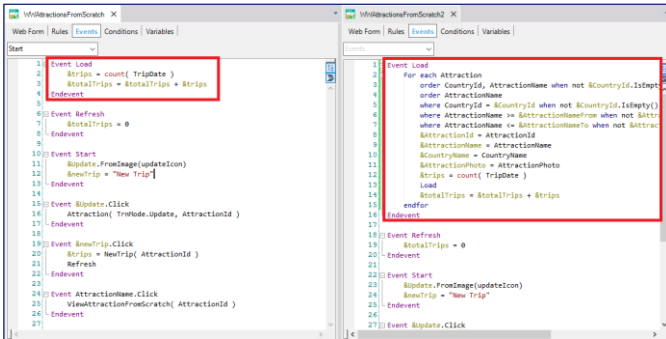
Lo que vamos a tener que hacer es programar, entonces, dentro del **evento Load**, el acceso a la tabla Attraction con un for each. En él especificaremos las cláusulas order y en las cláusulas where las condiciones que deben cumplir los datos. Y por cada registro que cumpla con esos filtros, cargaremos las variables del grid con los valores de esa atracción. Y cuando tenemos todas las variables cargadas, entonces escribimos el comando Load, para que se agregue una línea en el grid con esos valores.



Navigation Report:

```

Event Load
For Each Attraction (Line: 2)
Order:
  CountryId, AttractionName WHEN not &CountryId.isempty()
  ! No index
  AttractionName OTHERWISE
  ! No index
Navigation
Start from: FirstRecord
filters:
Loop while: NotEndOfTable
Constraints:
  CountryId = &CountryId WHEN not &CountryId.isempty()
  AttractionName >= &AttractionNameFrom WHEN not &AttractionNameFrom.isempty()
  AttractionName <= &AttractionNameTo WHEN not &AttractionNameTo.isempty()
Join location: Server
  -Attraction ( AttractionId)
  -Country ( CountryId)
  -count( TripDate) navigation
  -TripAttraction ( AttractionId)
  -Trip ( TripId)
    
```



Control Name	Grid1
Collection	
Base Trm	
Order	
Conditions	

Grid sin tabla base

Aquí tenemos este web panel ya implementado. Hemos hecho un save as del que teníamos con atributos, y lo cambiamos por variables:

Y también cambiamos los eventos, tal como estudiamos recién.

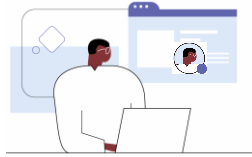
Probemos lo hecho para ver cómo en ejecución no notamos diferencia alguna entre el web panel con tabla base y el web panel sin.

Filtremos por ejemplo por Francia.

Ahora, si observamos el listado de navegación del web panel sin tabla base:

Vemos que aparece el for each en el Load, indicando la navegación.

Una observación: para decir que el grid y el web panel no tienen tabla base, no alcanza con que solo haya variables en la pantalla y ningún atributo. No deberán haber atributos en el Order del grid, ni en las conditions, por supuesto no deberá haber Transacción base indicada, ni atributos fuera de for eachs en los eventos, como aquí.



FRONT-END

GUI LOGIC



BACK-END

LOGIC

En el siguiente video veremos a partir de estos ejemplos un resumen del esquema de ejecución de eventos tanto para el caso en el que hay tabla base como en el que no.

Más de un grid

The screenshot shows a web form titled 'ViewCountryInfo' with a 'MainTable' container. Inside the table, there are input fields for 'Country Name', 'Attraction Name From', and 'Attraction Name To'. Below these is a 'GRID' with columns: 'Attraction Id', 'Attraction Name', 'Attraction Photo', 'Trips', and '&newTrip'. The 'Trips' column contains a '&trips' field. Below the grid is a 'Total Trips' field with '&totalTrips' next to it. Arrows point from the labels 'Grid1' and '+ Grid2' to the grid and the total trips field respectively.

```

Event Grid1.Load
    &trips = Count(TripDate)
    &totalTrips = &totalTrips + &trips
Endevent
  
```

Dijimos un par de veces que tal vez hubiese sido mejor utilizar el evento Load del grid y no el genérico, que solo sirve en el caso de un web panel sin grid o con un grid. Usando el Load del grid nos anticipamos al futuro, a la posibilidad de ingresar otro grid más.

GeneXus™

training.genexus.com
wiki.genexus.com
training.genexus.com/certifications