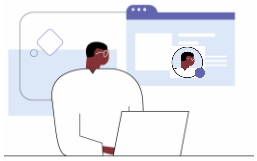
Pantallas web con foco en Back-office

Objeto Web Panel. Esquema de ejecución de eventos

GeneXus

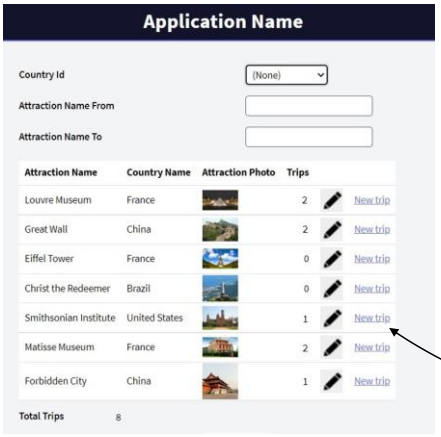
Ejecución del Web Panel por primera vez



FRONT-END

BACK-END

GUI LOGIC → LOGIC

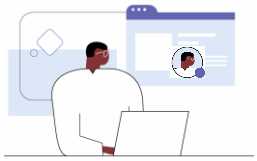



```
Event Start
     &update.FromImage(updateIcon)
     &newTrip = "New trip"
Endevent

Event Refresh
     &totalTrips = 0
Endevent

For each Attraction
     order CountryId, AttractionName when not &CountryId.IsEmpty()
     order AttractionName
     where CountryId = &CountryiId when not &CountryiId.IsEmpty();
     where AttractionName >= &AttractionNameFrom when not &AttractionNameFrom.IsEmpty();
     where AttractionName <= &AttractionNameTo when not &AttractionNameTo.IsEmpty();

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

Si traducimos toda esta lógica que hemos visto a Front-end y Back-end interactuando, entonces...

Cuando se ejecuta por primera vez el web panel, desde la capa de lógica del Front-end se invoca a la lógica del Web panel en el Back-end, donde se ejecuta: el evento Start, el evento Refresh y a continuación, si el grid tiene tabla base, entonces dentro de la lógica GeneXus ha programado de manera transparente una suerte de for each, para acceder a la tabla base, respetando order y conditions especificadas en el grid, y para cada registro encontrado, ejecuta el evento Load, agregando al finalizar una línea a los datos a ser devueltos al Front-end. El agregado de la línea también es automático, el desarrollador no tiene por qué especificar comando Load para ello. Al finalizar se le envía la respuesta al Front-end, que presentará la información en la pantalla.

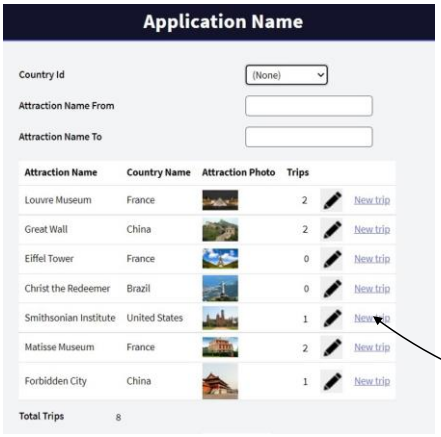
Ejecución del Web Panel por primera vez



FRONT-END

BACK-END

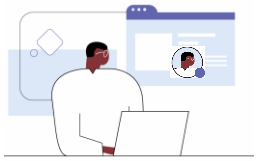
GUI LOGIC → LOGIC




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

Event Refresh
     &totalTrips = 0
Endevent

Event Load
     For each Attraction
          order CountryId, AttractionName when not &CountryId.IsEmpty()
          order AttractionName
          where CountryId = &CountryiId when not &CountryiId.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
```

En cambio, si no hubiera tabla base, entonces esta parte de código no estaría, sino que directamente se dispararía el evento Load, una única vez, donde si queremos acceder a la base de datos debemos programarlo explícitamente, y es por eso que aquí aparece el comando for each programado por el desarrollador. Para que se cargue una línea en el grid, el desarrollador debe indicarlo explícitamente con el comando Load porque GeneXus no puede saber cuál es nuestra intención. Finalizado todo esto, exactamente igual que en el otro caso, se le envía la respuesta al Front-end, que presentará la información en la pantalla.

Ejecución del Web Panel por primera vez

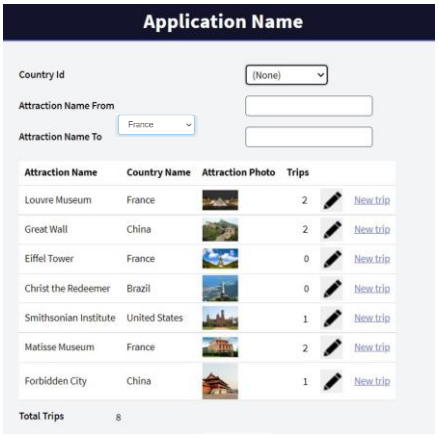


FRONT-END

GUI LOGIC

BACK-END

LOGIC

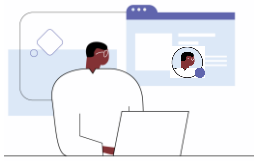



```
Event Refresh
     &totalTrips = 0
Endevent

For each Attraction
     order CountryId, AttractionName when not &CountryId.IsEmpty()
     order AttractionName
     where CountryId = &CountryiId when not &CountryiId.IsEmpty();
     where AttractionName >= &AttractionNameFrom when not &AttractionNameFrom.IsEmpty();
     where AttractionName <= &AttractionNameTo when not &AttractionNameTo.IsEmpty();

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

Ahora el usuario va a interactuar con la pantalla. Dependiendo de cuál sea la interacción, lo que sucederá.

Por ejemplo, si ingresa un valor en una de las variables de filtro de los datos de un grid, desde la capa de lógica del Front-end se invocará al Back-end, pasándole los valores de las variables. Allí se ejecutarán los eventos Refresh y Load para cargar nuevamente la información de ese grid. Otra vez, si el grid tiene tabla base, entonces este será el código que se ejecutará en el server....

Cambios en las variables de los filtros

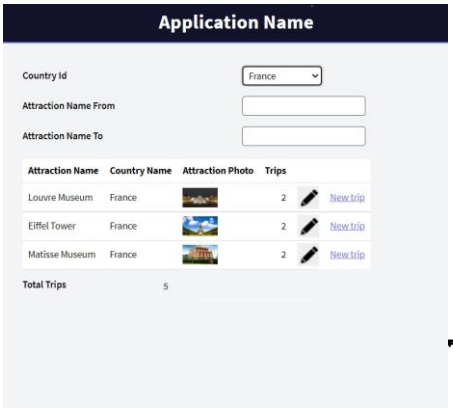


FRONT-END

GUI LOGIC

BACK-END

LOGIC

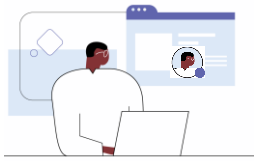



```
Event Refresh
     &totalTrips = 0
Endevent

Event Load
    For each Attraction
         order CountryId, AttractionName when not &CountryId.IsEmpty()
         order AttractionName
         where CountryId = &CountryiId when not &CountryiId.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
```

...y si no, será este otro.

En cualquier caso, al volverse a consultar la base de datos, ahora algunos registros no pasarán el filtro. Otra vez, se responderá al Front-end que ahora presentará el grid con los datos resultantes, entre los cuales está la variable &TotalTrips.

Cambios en las variables de los filtros

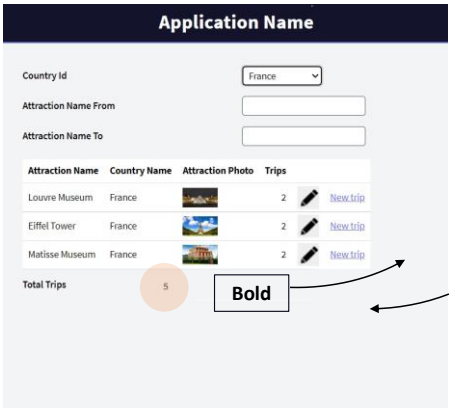


FRONT-END

GUI LOGIC

BACK-END

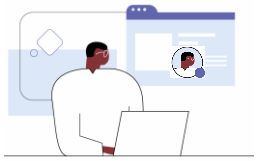
LOGIC




```
Event 'Bold'
     &totalTrips.FontBold = True
Endevent
```

Si tuviéramos un evento cuyo código puede resolverse en el cliente, como por ejemplo, cambiar la fuente a negrita para un control del form, ese código se ejecutará en el propio Front-end directamente, sin necesidad de ningún viaje al servidor.

Cambios en las variables de los filtros

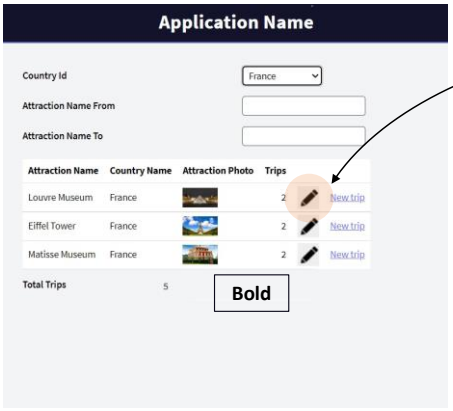


FRONT-END

GUI LOGIC

BACK-END

LOGIC

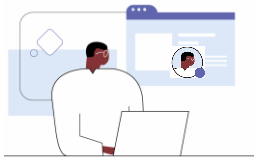



```
Event &newTrip.Click
     &trips = NewTrip(AttractionId)
endevent

parm(in:&AttractionId, out:&trips);
new
     TripDate = Today()
     TripDescription = "Created automatically"
endnew
&tripId = TripId
new
     TripId = &tripId
     AttractionId = &AttractionId
endnew
&trips = Count(TripDate, AttractionId = &AttractionId)
```

En cambio, si necesita ejecutarse en el servidor, como era nuestro ejemplo donde invocábamos a un procedimiento para crear un trip con la atracción, entonces desde el Front-end se invoca al Back-end donde se encuentra programado el evento asociado, que invoca al procedimiento que allí se ejecuta, inserta los registros, y devuelve la ejecución al evento que lo llamó.

En este caso, como se le asigna el resultado del procedimiento a una variable del grid, entonces se le envía al Front-end la información de que debe modificar el valor de esa línea, que es lo que allí se hace.

Cambios en las variables de los filtros

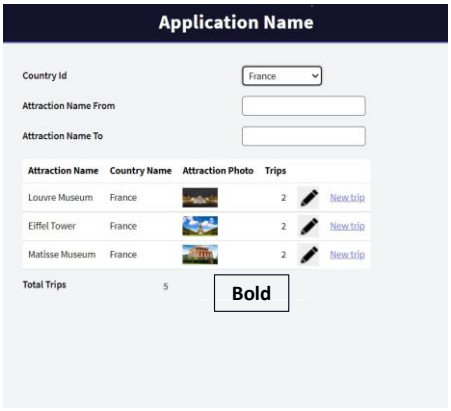


FRONT-END

GUI LOGIC

BACK-END

LOGIC

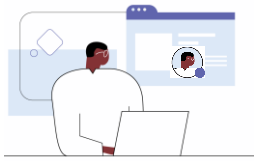



```
Event &newTrip.Click
     &trips = NewTrip(AttractionId)
     Refresh
endevent
```

```
new
     TripDate = Today()
     TripDescription = "Created automatically"
endnew
&tripId = TripId
new
     TripId = &tripId
     AttractionId = &AttractionId
endnew
&trips = Count(TripDate, AttractionId = &AttractionId)
```

Pero si luego de la invocación encuentra un comando Refresh,

Cambios en las variables de los filtros

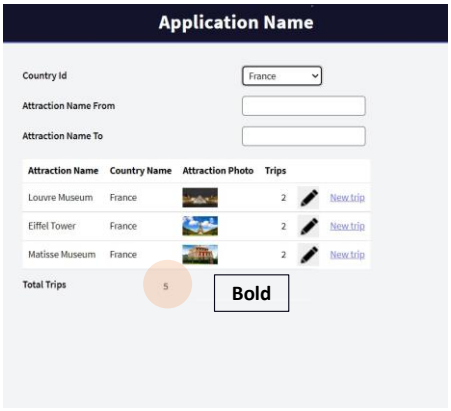


FRONT-END

GUI LOGIC

BACK-END

LOGIC




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

Event Refresh
    &totalTrips = 0
Endevent
For each Attraction
    order CountryId, AttractionName when not &CountryId.IsEmpty()
    order AttractionName
    where CountryId = &CountryiId when not &CountryiId.IsEmpty();
    where AttractionName >= &AttractionNameFrom when not &AttractionNameFrom.IsEmpty();
    where AttractionName <= &AttractionNameTo when not &AttractionNameTo.IsEmpty();
        Event Load
            &trips = Count(TripDate)
            &totalTrips = &totalTrips + &trips
        Endevent
endfor
```

entonces antes de enviar una respuesta al cliente, ejecuta los eventos Refresh y Load, (aquí vemos el código para el grid con tabla base) tras lo que devuelve al Front-end todas las líneas del grid que cumplen las condiciones, nuevamente y la variable &totalTrips, que el usuario ve con los valores esperados.

GeneXus™

training.genexus.com
wiki.genexus.com