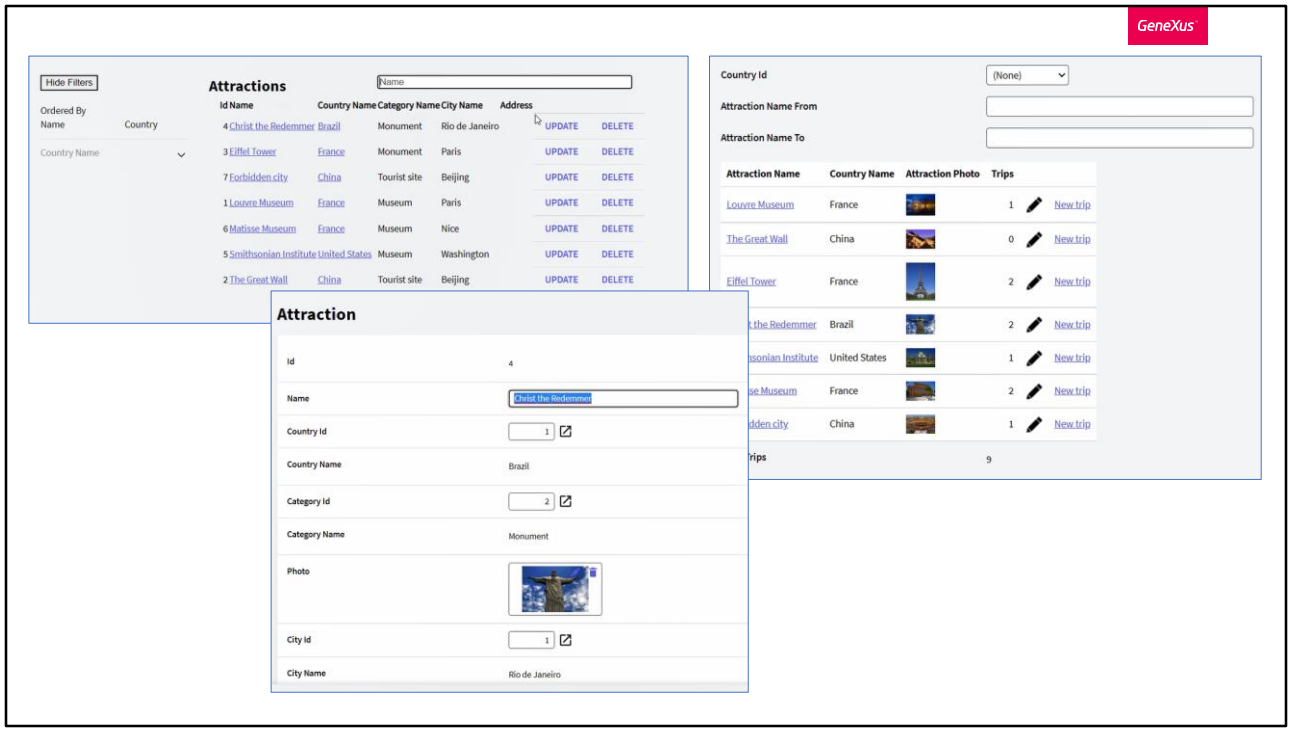
Tipos de Web Panels

GeneXus™



En los videos anteriores estuvimos construyendo desde cero una solución muy parecida a la que construye automáticamente por nosotros el patrón Work With.

Así, si comparamos en ejecución ambas soluciones, vemos primeramente que difieren en cuanto al diseño. Es que hasta ahora nos concentramos en la lógica y dejamos de lado la User Interface, tema en el que entraremos luego. Pero dejando de lado esas diferencias visuales, en ambos web panels podemos ver un grid que muestra información de las atracciones turísticas, con filtros, y la posibilidad, por ejemplo, de actualizar la información.

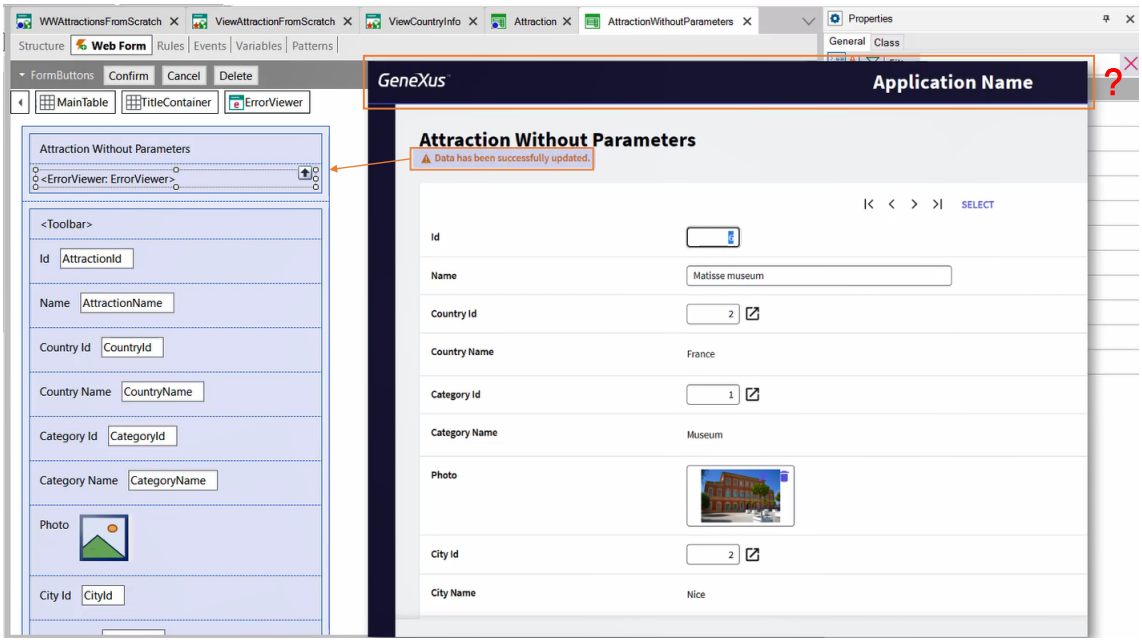
En las dos soluciones se invoca a la transacción en modo Update.

Algo que ya puede empezar a llamarnos la atención es una constante de todas las pantallas que venimos navegando. Vemos que página que abrimos, página que tiene esta parte.

Por ejemplo, si invocamos directamente a la transacción paralela AttractionWithoutParameters, aquí está.

O, si por ejemplo ahora vamos al View de una atracción, el del pattern o el

implementado de cero por nosotros, vemos que más allá de las diferencias, siguen manteniendo esto en común.



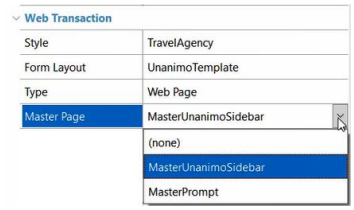
Si ahora vamos a GeneXus a ver el form de este View que implementamos nosotros, ¿dónde está esa parte común?

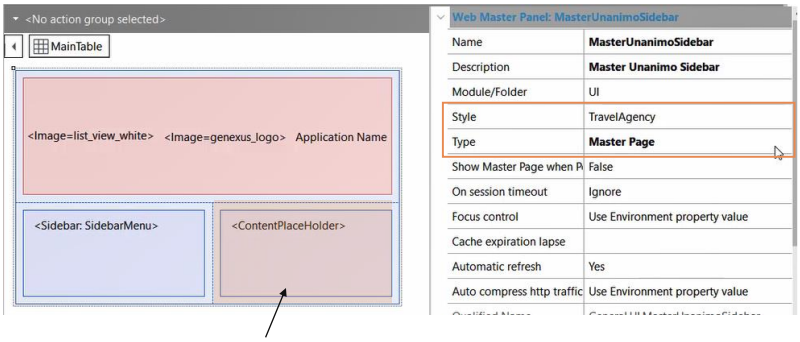
O, si abrimos la transacción Attraction, o la transacción paralela que no recibe parámetros, y vamos a su Form, tampoco la vemos.

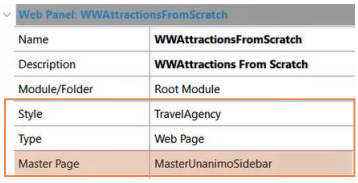
Aquí lo que estamos viendo es un control textblock con el nombre de la transacción. Debajo estamos viendo un control ErrorViewer que es donde aparecen los mensajes, por ejemplo de éxito o fracaso. Luego tenemos este control “action group” que es el que presenta los botones de navegación y el Select. Luego vienen los atributos, y por último tenemos otro “action group” con los botones.

¿Dónde está el área de arriba?

Master Page







Si observamos las propiedades de la transacción vemos un grupo Web Transaction con cuatro propiedades importantes: en la primera se asigna el objeto con el que se aplicará el estilo, en la segunda se especifica el templete que se utilizará para generar el form, la tercera la veremos a continuación, y la cuarta es la que veremos ahora, la propiedad Master page. Allí se indica cuál será la página maestra dentro de la cual la página correspondiente a este objeto transacción se cargará Vemos que nos ofrece algunas posibilidades, que corresponden a todas las páginas maestras definidas por ahora en nuestra KB. Cuando se crea una KB se crean estas páginas para ya contar con algunas por defecto, pero podremos crear otras.
Vemos que por defecto la transacción fue creada asociada a esta página maestra.

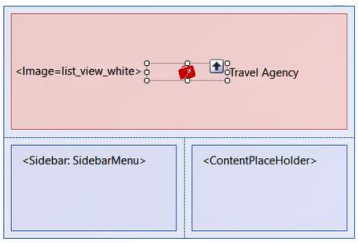
Si ahora abrimos los demás objetos que estuvimos viendo en ejecución no nos sorprenderá encontrar que también tienen página maestra, y que es esta misma.
Ya de paso vemos que los Web panels también cuentan con las mismas cuatro propiedades.

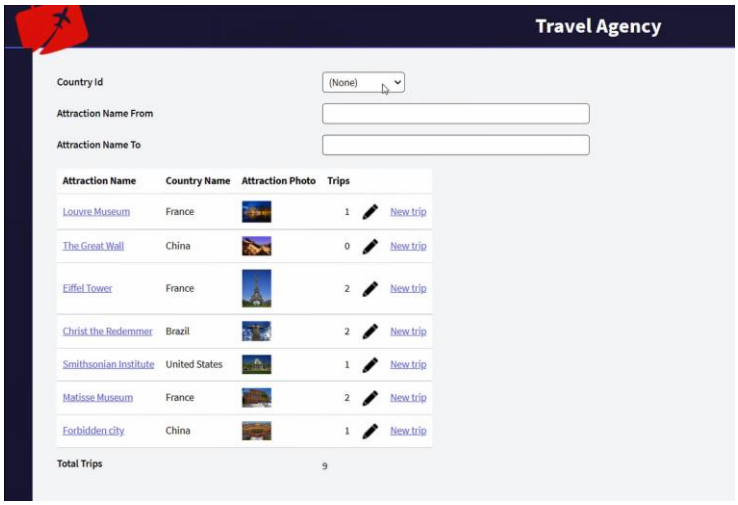
¿Vamos a buscar esta página maestra para ver de qué se trata? Si no sabemos dónde se encuentra, podemos buscarla por aquí y abrirla.

Si observamos sus propiedades, vemos la de nombre Style y la de nombre Type, y ya no está la Master Page. ¿Por qué? Porque se trata ni más ni menos que de un web panel de un tipo especial, el tipo Master Page. Un web panel de este tipo tendrá un control especial, el control ContentPlaceHolder. Aquí es donde toda página que tenga a ésta como su página maestra va a cargarse.

Entonces la transacción se cargará en este espacio, el view, todo lo que vimos.

Master Page



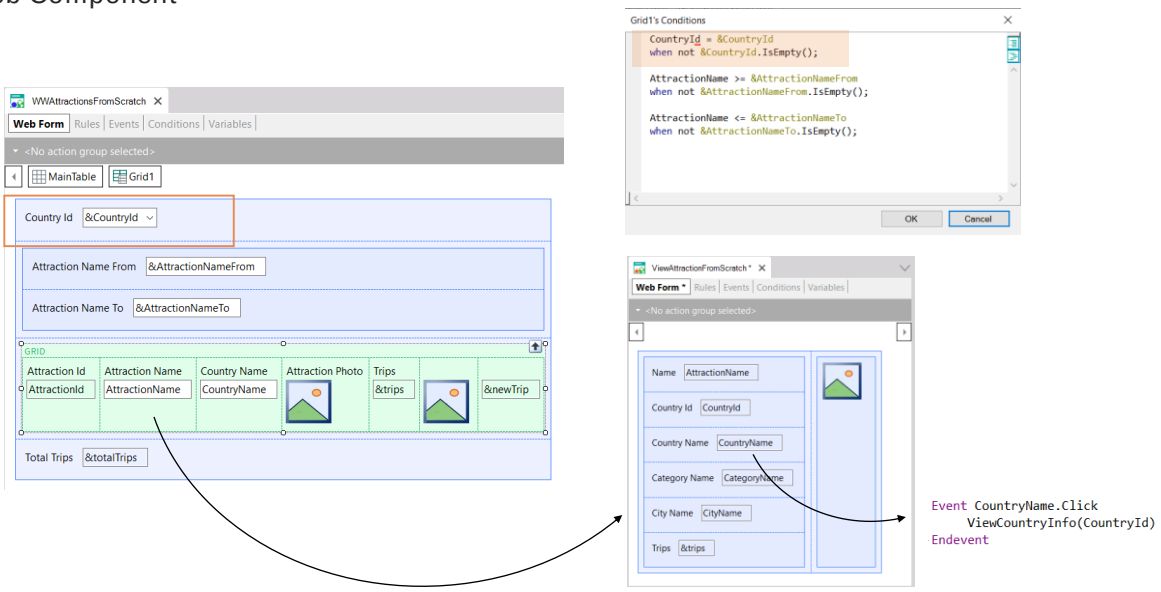


Aquí vemos el control text block que vemos en ejecución con el nombre Application Name. Por ejemplo, probemos cambiarlo por Travel Agency. Aquí tenemos una imagen que también podemos cambiar por esta otra que previamente insertamos en la KB.

Probemos el efecto de estos cambios en nuestra aplicación. Aquí los vemos.

Otro control que vemos en este Master Panel es un User Control, propio del design system Unanimo, de nombre Sidebar. Este control viene por defecto y mostrará una barra lateral desplegable en la que podremos tener accesos a diferentes objetos.

Web Component



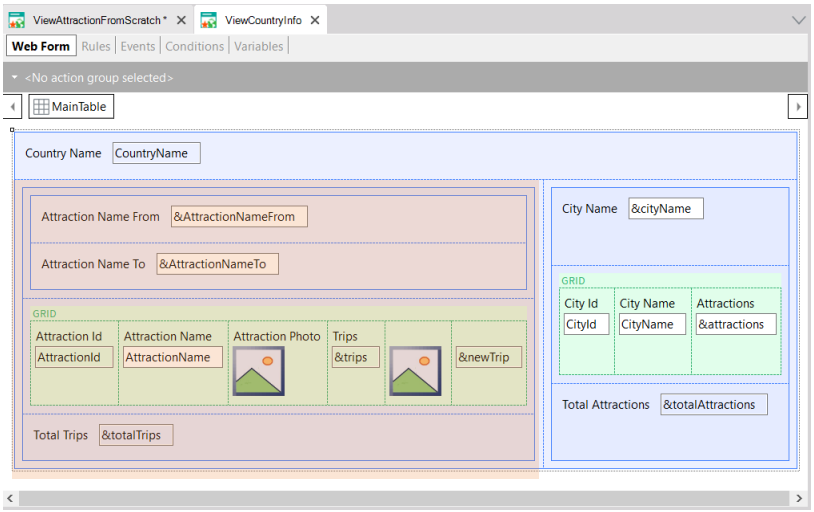
Bien, con esto ya vimos dos de los tres tipos de pantallas web: la página maestra, y la página web común, que es con la que trabajamos hasta ahora cada vez que creamos un web panel o una transacción. Nos queda ver solamente la página web de tipo componente.

Para ello, volvamos un poco sobre nuestros pasos y recordemos qué habíamos implementado en los videos anteriores. Teníamos la imitación del Work With de atracciones, en el cual el usuario podía filtrar por país eligiendo un valor en esta variable, que estaba siendo utilizada en las conditions del grid para filtrar por ese país si la variable no estaba vacía.

Pero además, desde este panel se permitía al usuario ver la información de una atracción, haciendo clic sobre su nombre, y a su vez, desde allí, se presentaba un link sobre el nombre de país para mostrar toda la información relevante del país.

GeneXus

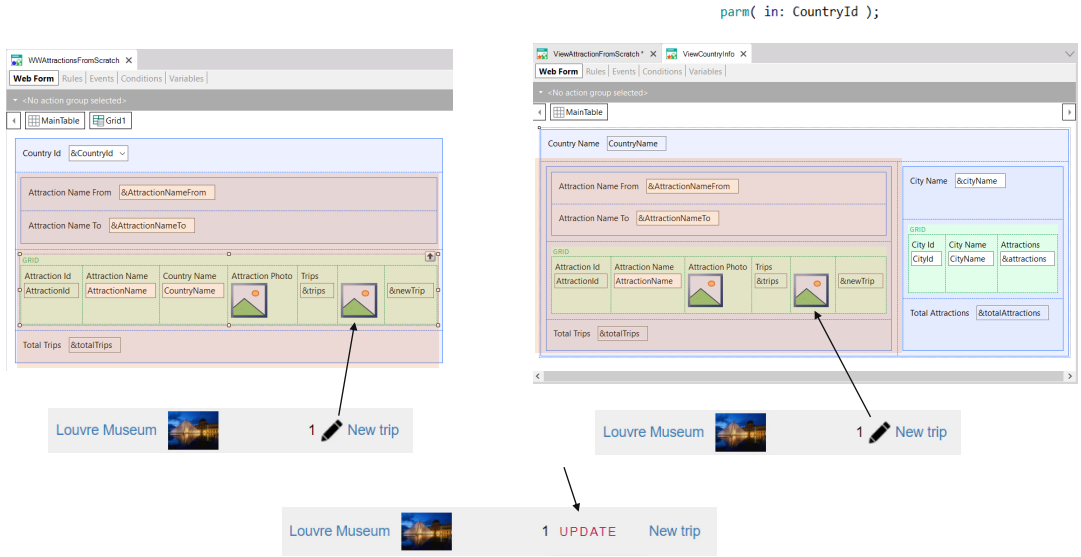
Web Component




```
parm( in: CountryId );
```

Para eso llamábamos a este web panel que recibía en el atributo el identificador de país, y mostraba su nombre, y luego un grid idéntico al primero, con los mismos filtros por nombre de atracción, las mismas columnas y acciones e info en general, y además, información de las ciudades.

Web Component



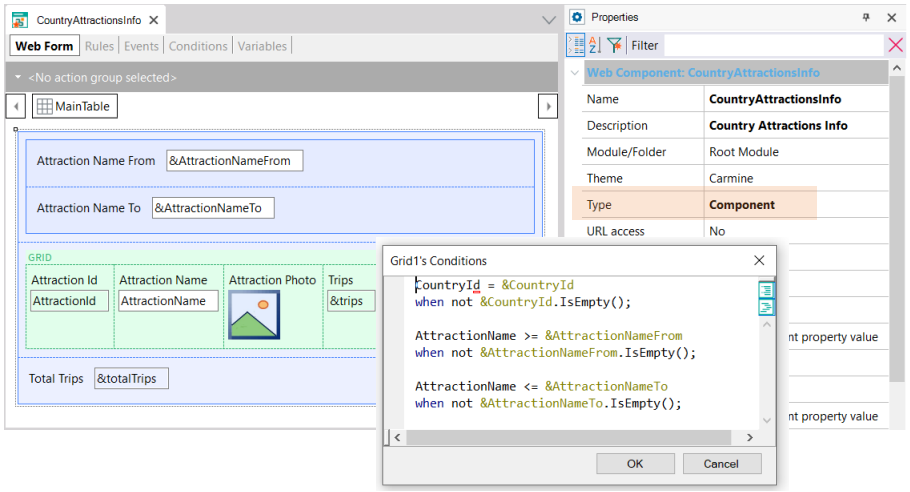
Vemos claramente que hay una parte de este panel que es prácticamente idéntica a la del otro. Supongamos que ya no quisiéramos que la opción de Update se ofrezca con esta imagen, sino que queremos que sea como es en el pattern, con la palabra UPDATE.

Vamos a tener que sustituir esta imagen por un text block en ambos paneles. Para evitar estas duplicaciones y programar solamente una vez el comportamiento y diseño de una porción del panel, es que contamos con los web panels de tipo componente.

GeneXus

Web Component

```
parm( in: &CountryId );
```

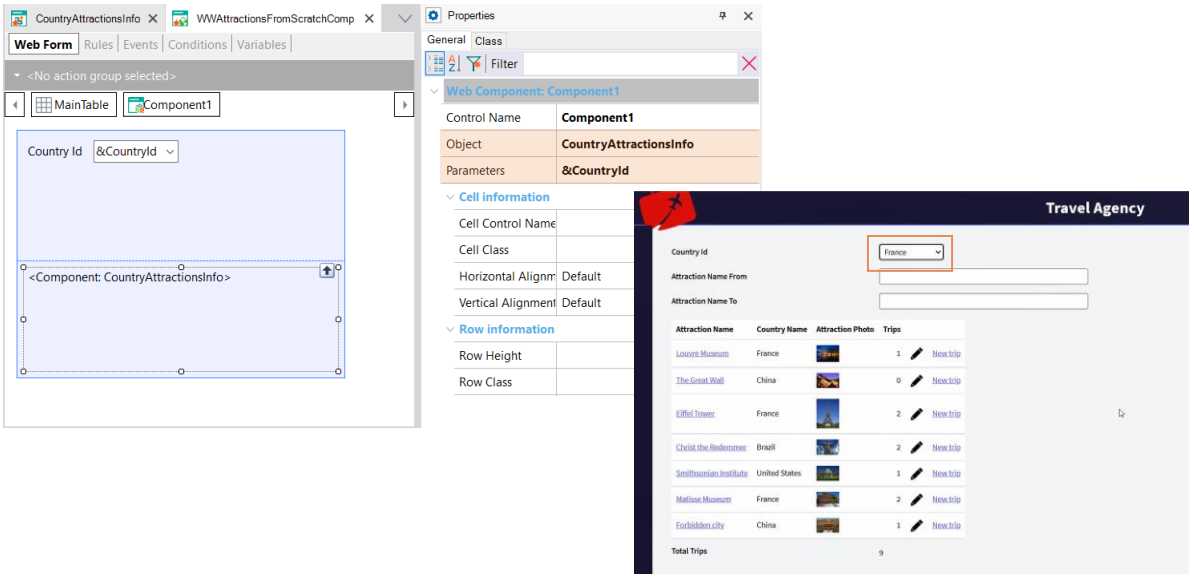


¿Qué es lo que en nuestro caso se repite? Toda esta sección de la pantalla y su comportamiento.

Entonces lo que hicimos fue un Save as de este web panel, al que únicamente le quitamos la variable CountryId del form y la colocamos, en cambio, como parámetro.
Ya no lo ingresará el usuario directamente en el form de esta pantalla, sino que será recibido de quien lo llame.

Vemos que las conditions se mantienen idénticas, así como los eventos y todo lo demás. El otro cambio es que modificamos la propiedad Type, pasándola a Component. A partir de ahora, este web panel podrá ser un componente de otro.

Component

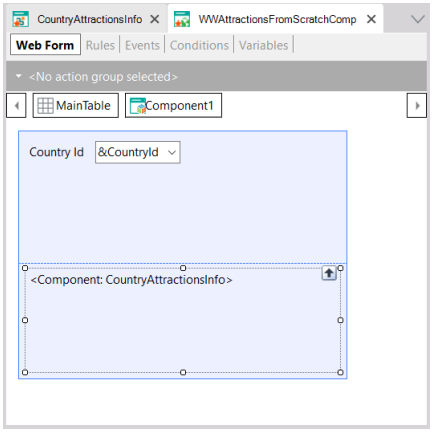


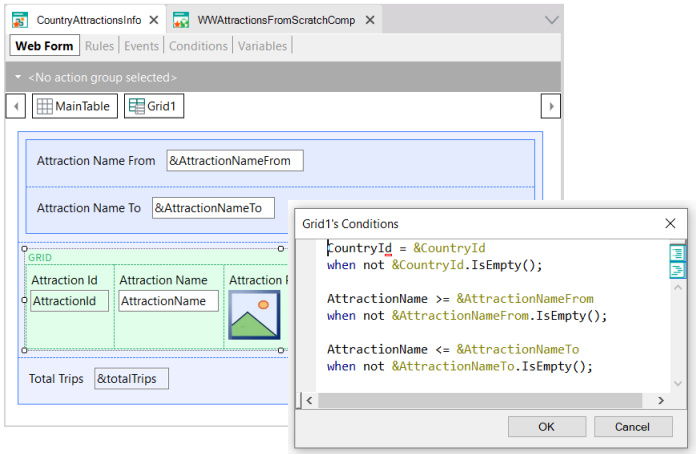
Así, lo siguiente que hicimos fue un save as de nuestro panel original, y sustituimos todos esos controles que ahora colocamos en el web panel componente, por un control de tipo component. Por supuesto eliminamos (aquí dejamos comentados) todos los eventos que ahora estarán en el componente.

Al colocar un control de este tipo, estamos diciendo que allí dentro deberá cargarse y ejecutarse lo que especifiquemos. Tenemos que decirle que en ese componente se cree una instancia del web component que acabamos de mostrar, CountryAttractionsInfo. Lo podemos hacer de manera estática, indicándolo así en las propiedades, donde aquí indicamos cuál será el objeto de tipo componente y aquí el parámetro enviado. Probémoslo.

Vemos exactamente lo mismo. Si filtramos por nombre de atracción... está funcionando perfecto.
Ahora, qué pasa si queremos filtrar por país? No pasa nada. ¿Por qué?

Component






```
Event &CountryId.ControlValueChanged
     Component1.Object = CountryAttractionsInfo.Create(&CountryId)
Endevent
```

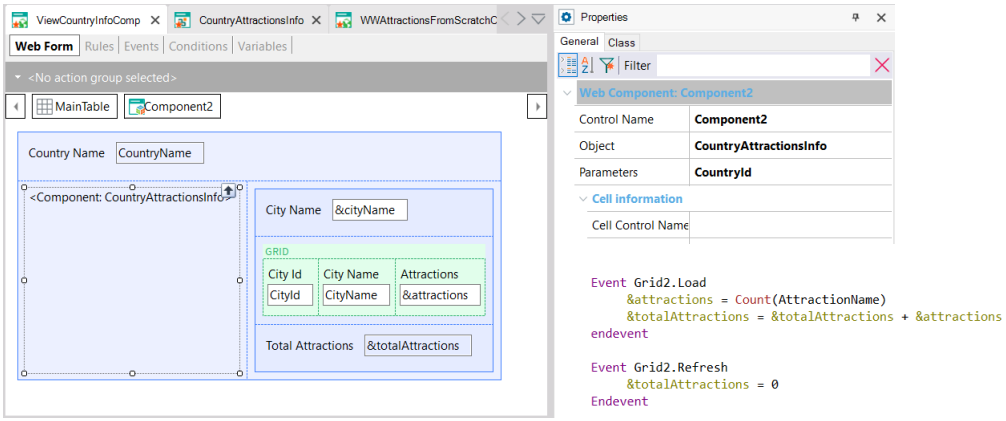
La variable CountryId está en un panel distinto que el grid de atracciones por el que se filtra. Aquí lo que tenemos que hacer es utilizar el evento ControlValueChanged, que captura el momento en que se modifica el valor del control variable &CountryId, y pedir que se cree una nueva instancia del componente, pasándole ahora el nuevo valor de la variable.

Probémoslo

Refrescamos. Intentamos filtrar por Francia. Perfecto. Y allí dentro por nombre de atracción. Perfecto también.

Component

```
parm( in: CountryId );
```




```
Event Grid2.Load
     &attractions = Count(AttractionName)
     &totalAttractions = &totalAttractions + &attractions
endevent

Event Grid2.Refresh
     &totalAttractions = 0
Endevent
```

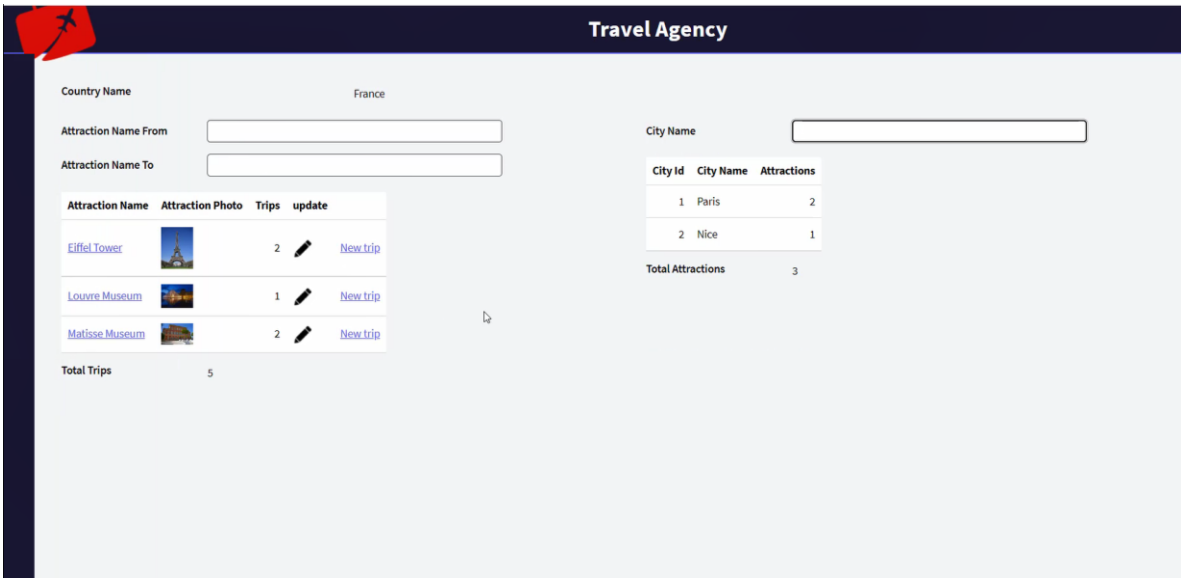
Y, por supuesto, hicimos algo similar con el panel que mostraba la info del país.

Hicimos un Save as de este panel, al que le reemplazamos toda esta sección por un componente que se cargará con este panel.

Así, en el nuevo panel el CountryId no es variable, sino que se recibe por parámetro, y por ello podemos crear la instancia del componente de manera estática, una única vez dentro de la ejecución de este web panel, pasándole aquí el parámetro que en este caso viene en el atributo recibido en la regla parm.

Observemos que del panel original quitamos los controles y la programación de eventos asociados a las atracciones, y solo dejamos las de las ciudades. Probémoslo en ejecución.

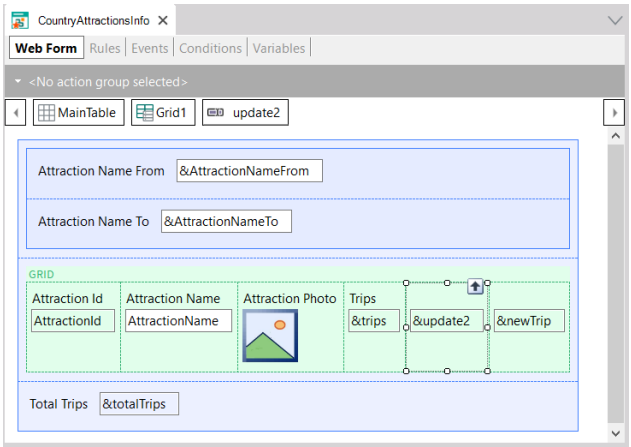
Para ello invoquemos a este panel y no al anterior.



Ahora modifiquemos la imagen para realizar el UPDATE, y coloquemos en su lugar el texto UPDATE.

GeneXus

Web Component



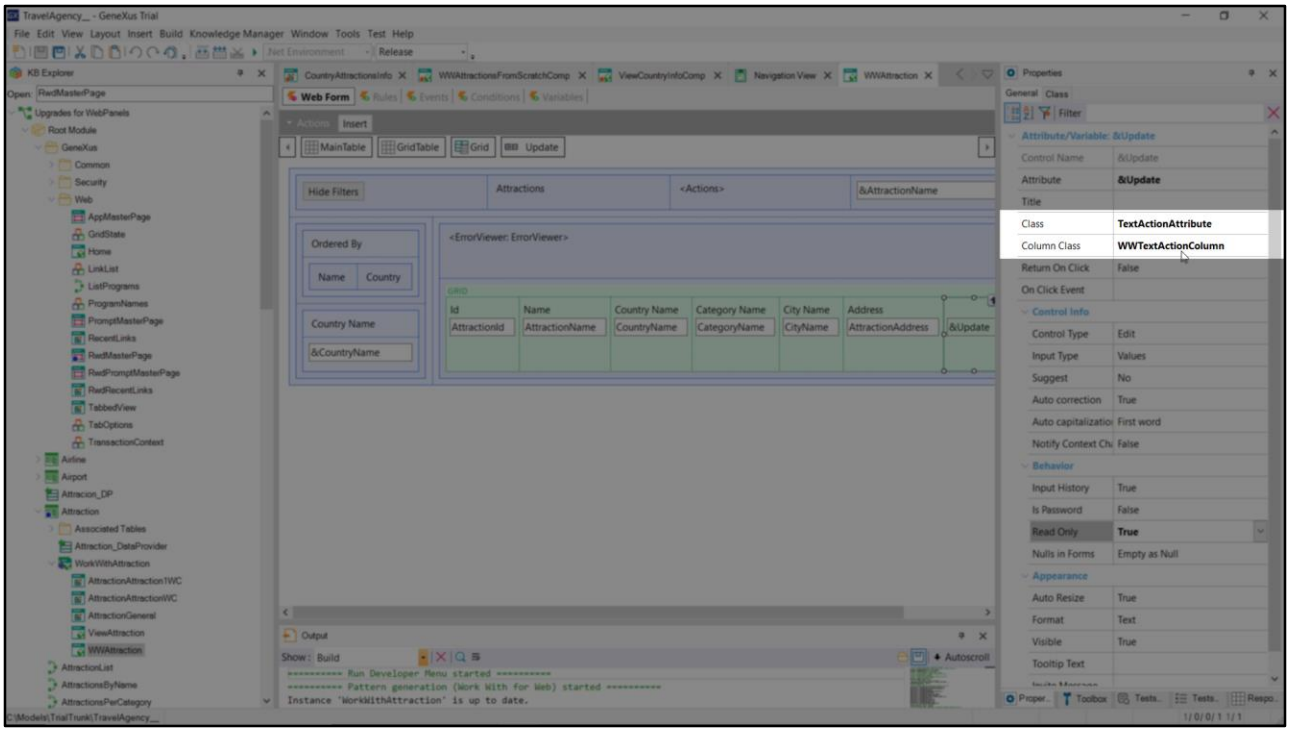

```
Event Grid1.Load
     &trips = Count(TripDate)
     &totalTrips = &totalTrips + &trips
Endevent

Event Grid1.Refresh
     &totalTrips = 0
Endevent

Event Start
     &update2 = "UPDATE"
     &newTrip = "New trip"
Endevent

Event &update2.Click
     Attraction(trnMode.Update, AttractionId)
Endevent
```

Para ello, en el web panel componente creamos una variable de tipo Character de 20. La insertamos en el grid. En el evento Start le asignamos el valor UPDATE, que es lo que queremos que el usuario vea. Eliminamos la asignación de una imagen a la variable que teníamos, porque la vamos a eliminar del grid.

Y ahora quitémosle el título a la nueva variable y hagámosla Read only.

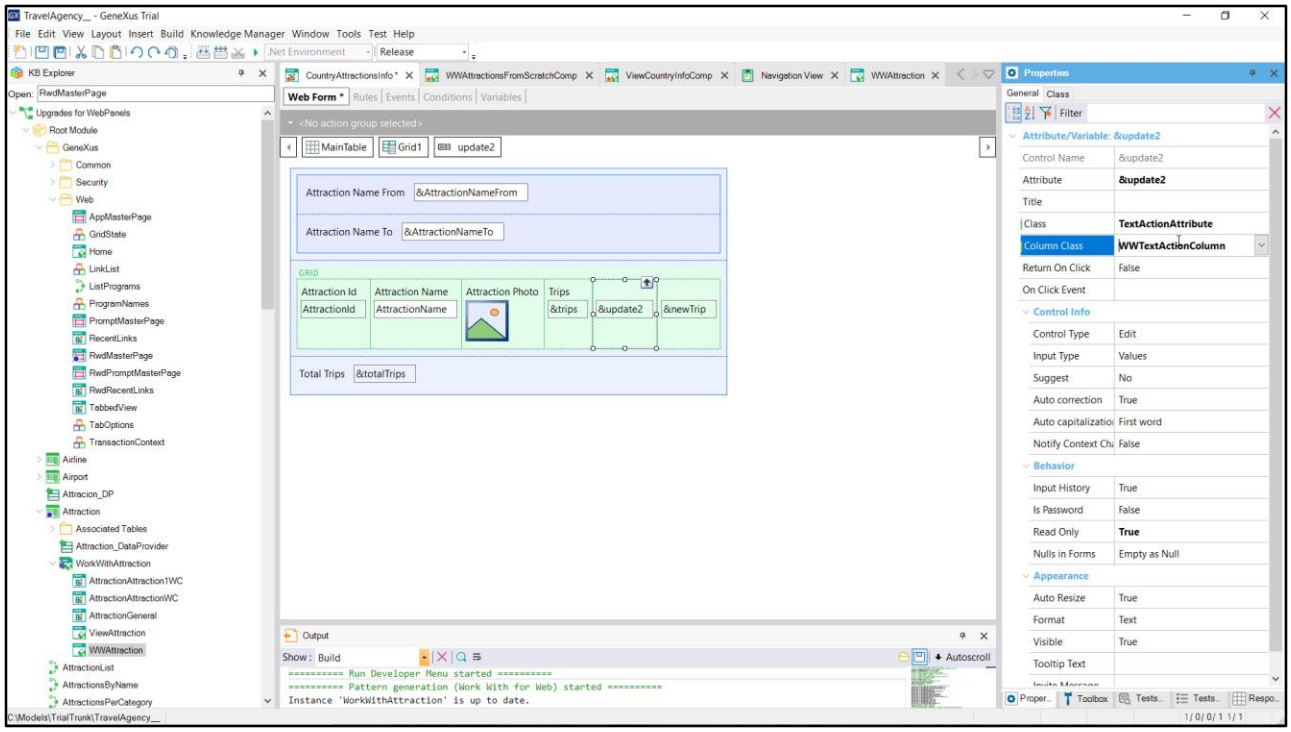
Intentemos ejecutar. Nos indica que el click no es un evento válido. Es que olvidámos modificarlo para que sea el click de la nueva variable y no la vieja.

Ahora sí... ejecutemos.

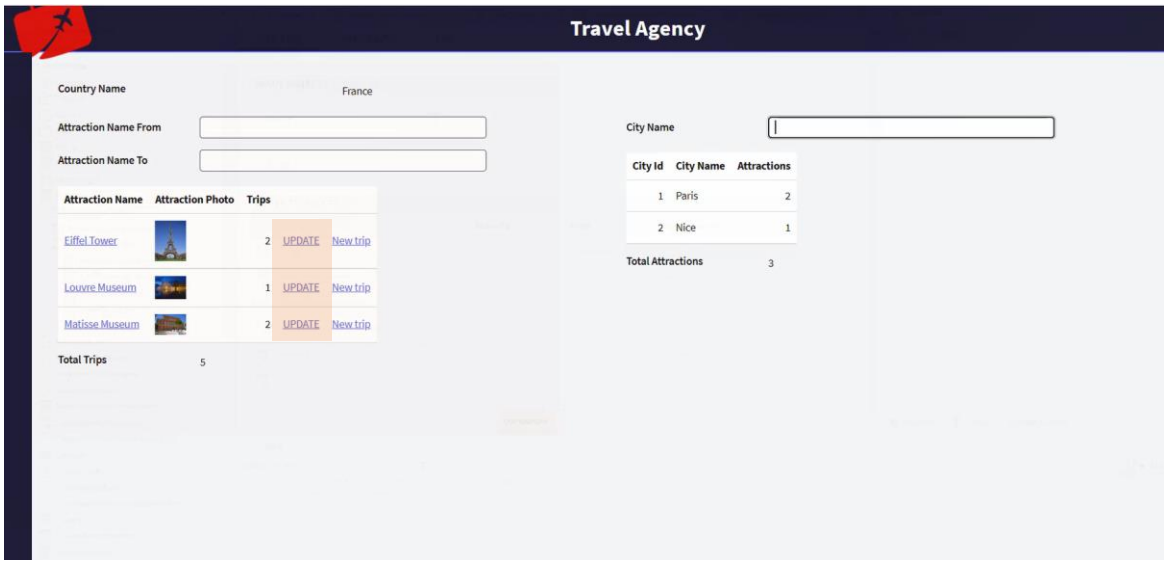


Si vamos a observar cómo quedó en el pattern, ¿por qué aquí tiene este diseño tanto más lindo?

Ubiquemos las propiedades del control en el web panel generado por el pattern. Veamos estas dos: Class y Column Class.



Y veamos los valores que tienen esas propiedades para nuestro control, el que insertamos nosotros a mano en nuestro web panel. Son distintas. Asignémosles las mismas que las del pattern. Y probemos.

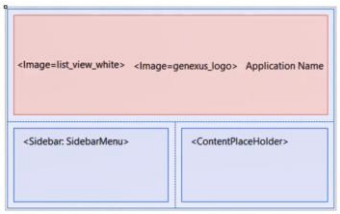


Ahí lo vemos. Con esto ya empezamos a entender cómo se manipula el diseño de nuestras pantallas.

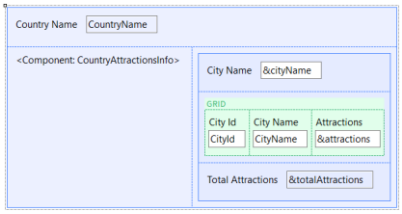
GeneXus

Types of Web Panels

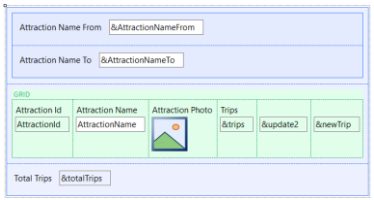
Master Page



Web Page



Component



Resumiendo: Vimos tres tipos de Web panels que están relacionados entre sí.

El de tipo Master Page, que ofrece un marco común a todas las páginas de la aplicación o de una parte de ella, para no tener que repetir lo mismo cada vez, para cada página. Por ejemplo, los menús suelen ir allí. Este objeto tiene la particularidad de contener en su form un control especial, el ContentPlaceHolder, donde se cargarán las páginas web.

El de tipo Web Page, que es el que veníamos estudiando, que podrá tener asociada una Master Page determinada y solo una, puesto que se cargará en el ContentPlaceHolder de ésta.

Y el de tipo Component, que sirve justamente para reutilizar diseño y programación en distintos objetos. Para poder hacer que un objeto definido como Web panel de tipo componente se cargue dentro de otro objeto web, se utiliza el control de tipo component, que podrá ser cargado estática o dinámicamente.

Cuantos más componentes identifiquemos, y utilicemos, mejor calidad tendrá la aplicación resultante.

Por supuesto, hay mucho más para estudiar sobre este tema (por ejemplo, qué pasa con la ejecución de los eventos en un panel con componentes, cómo se refresca un componente, etc.). Por ahora con esto es más que suficiente.

GeneXus™

training.genexus.com
wiki.genexus.com