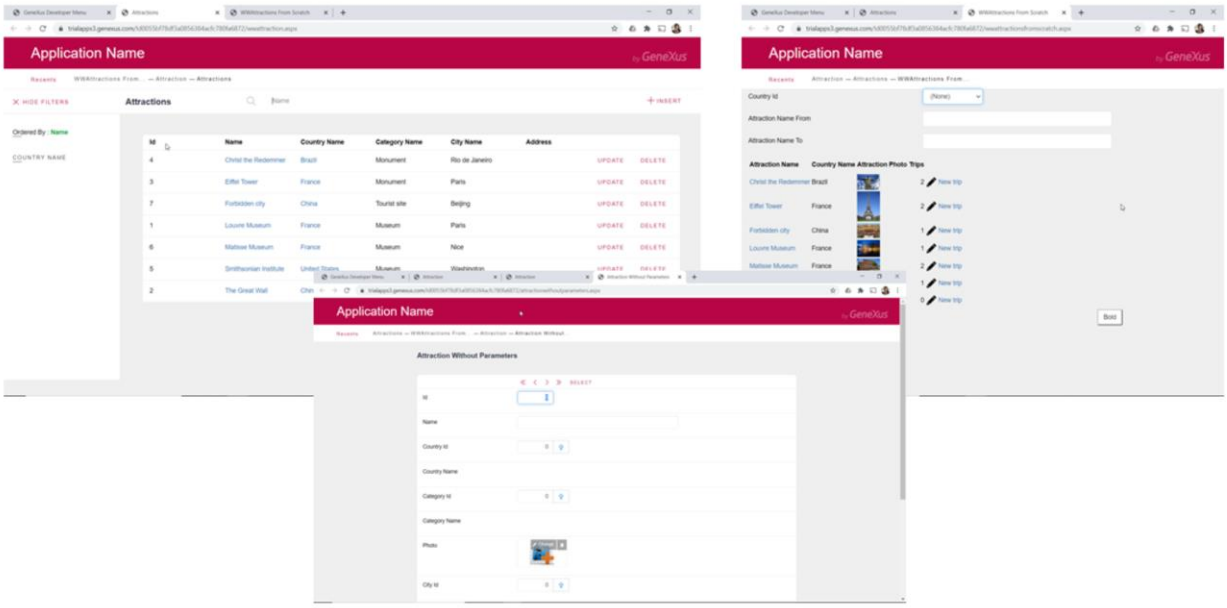
Telas web com foco em Back-Office

Tipos de Web Panels

GeneXus™

What all pages have in common



Nos vídeos anteriores construímos do zero uma solução muito parecida com a que é construída automaticamente, pelo padrão Work With.

Assim, se compararmos em execução ambas as soluções, vemos primeiramente que diferem quanto ao desenho. É que até agora nos concentramos na lógica e deixamos de lado a User Interface, tema no qual entraremos mais tarde. Mas deixando de lado essas diferenças visuais, em ambos web panels podemos ver um grid que mostra informações das atrações turísticas, com filtros, e a possibilidade, por exemplo, de atualizar a informação.

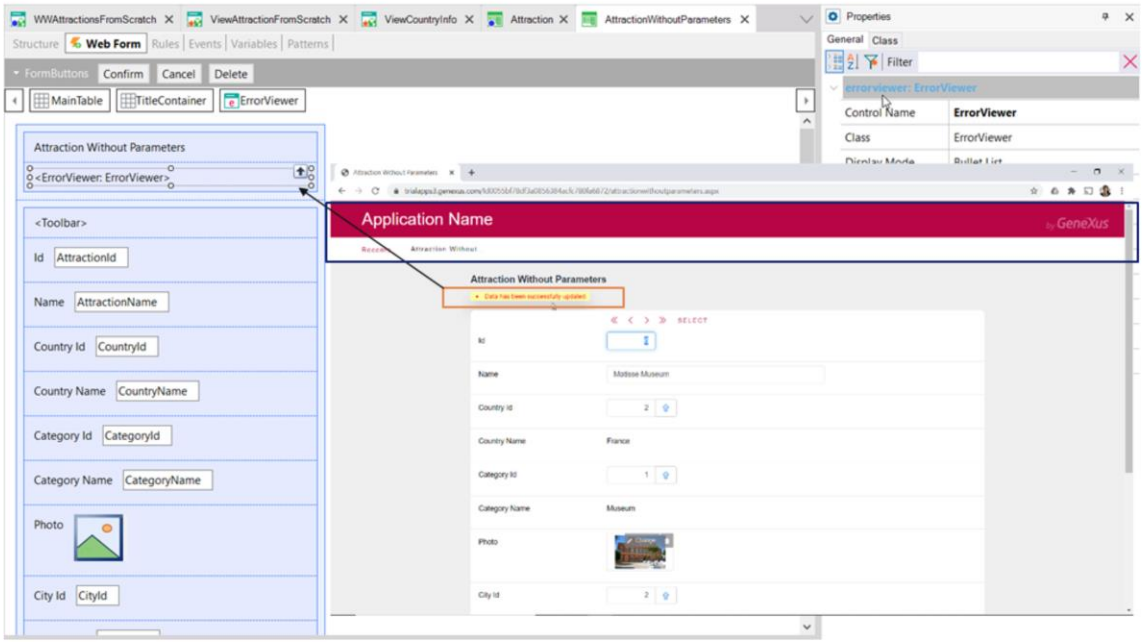
Nas duas soluções é invocada a transação em modo Update.

Algo que já pode começar a nos chamar a atenção é uma constante de todas as telas que temos navegado: esta barra aqui em cima e esta outra com os links navegados recentemente. Vemos que página que abrimos, página que contém essas partes.

Por exemplo, se invocamos diretamente a transação paralela AttractionWithoutParameters, aqui estão.

Ou, se por exemplo, agora vamos ao View de uma atração, o do pattern ou o implementado de zero por nós, vemos que apesar das diferenças, seguem mantendo isto em comum.

Transaction controls



?

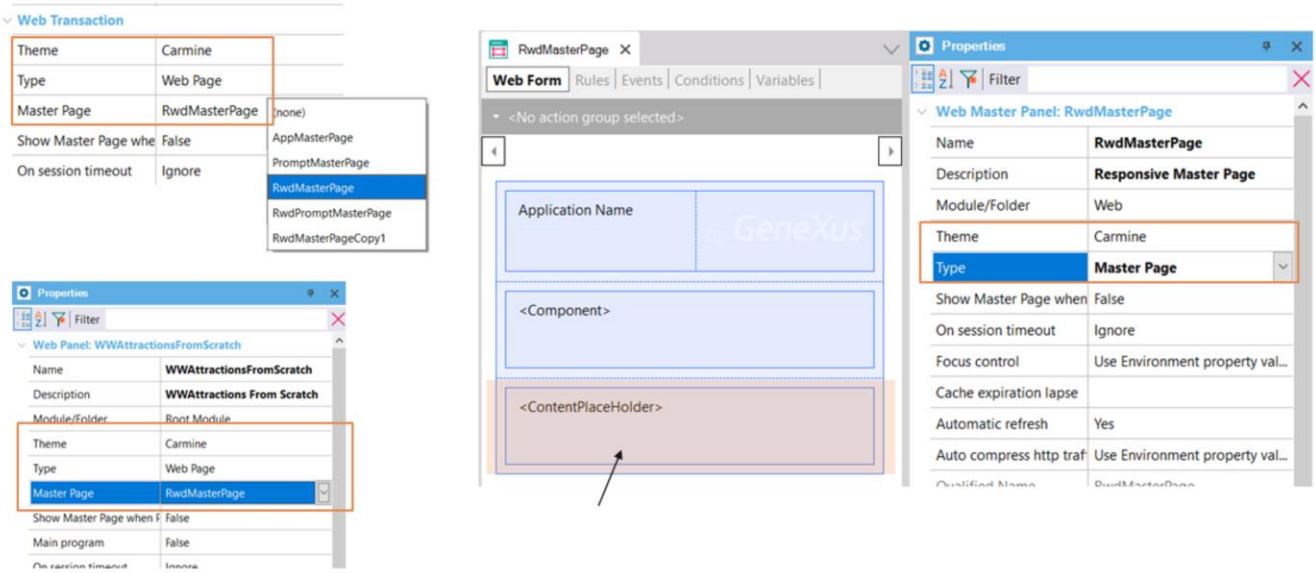
Se agora vamos ao GeneXus ver o form deste View que nós implementamos, onde está essa parte comum?

Ou, se abrirmos a transação Attraction, ou a transação paralela que não recebe parâmetros, e vamos para seu Form, também não a vemos.

Aqui o que estamos vendo é um controle textblock com o nome da transação. Abaixo estamos vendo um controle ErrorViewer que é onde as mensagens aparecem, por exemplo, de sucesso ou fracasso. Então, temos este controle "action group" que é o que apresenta os botões de navegação e o Select. Depois, vêm os atributos e, por último, temos outro "action group" com os botões.

Onde está a área de cima?

Master Page



Se observamos as propriedades da transação, vemos um grupo Web Transaction com três propriedades muito importantes: a primeira, **Theme**, controlará o desenho dos controles, a segunda veremos a seguir, e a terceira é a que veremos agora, a propriedade **Master Page**. Ali se indica qual será a **página mestra**, dentro da qual a página correspondente a este objeto transação será carregada. Vemos que nos oferece algumas possibilidades, que correspondem a todas as páginas mestras definidas, no momento, em nossa KB. Quando se cria uma KB, estas páginas são criadas para já ter algumas por padrão, mas podemos criar outras.
Vemos que, por padrão, a transação foi criada associada a esta página mestra.

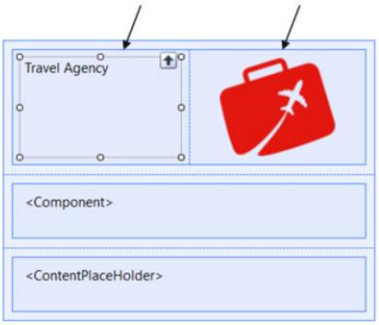
Se agora abrimos os outros objetos que vimos em execução, não ficaremos surpresos em descobrir que eles também têm uma página mestra, e que é esta mesma.
Já de início, vemos que os Web panels também contam com as mesmas três propriedades.

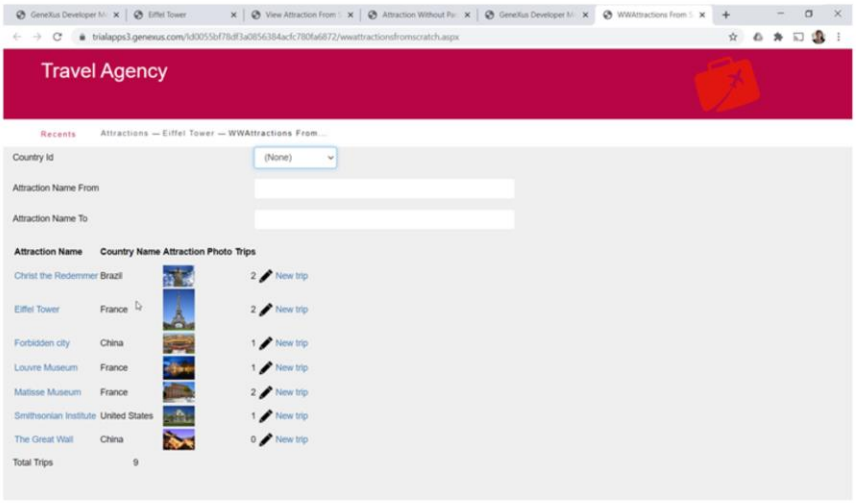
Vamos procurar esta página mestra para ver do que se trata? Se não sabemos onde se encontra, podemos procurá-la por aqui e abri-la. E se queremos localizá-la no KB Explorer, clicando com o botão direito sobre a aba temos a opção para isso. Está dentro de um folder web onde GeneXus coloca objetos que têm a ver com o pattern, por exemplo, e outros.

Se observamos suas propriedades, vemos também a de nome **Theme** e a de nome **Type**, e já não há a **Master Page**. Por quê? Porque se trata nada mais nem menos que de um web panel de um tipo especial, o **tipo Master Page**. Um web panel deste tipo terá um controle especial, **o controle ContentPlaceHolder**. Aqui é onde toda página que tenha a esta como sua página mestra, será carregada.

Então a transação será carregada neste espaço, o view, tudo o que vimos.

Master Page

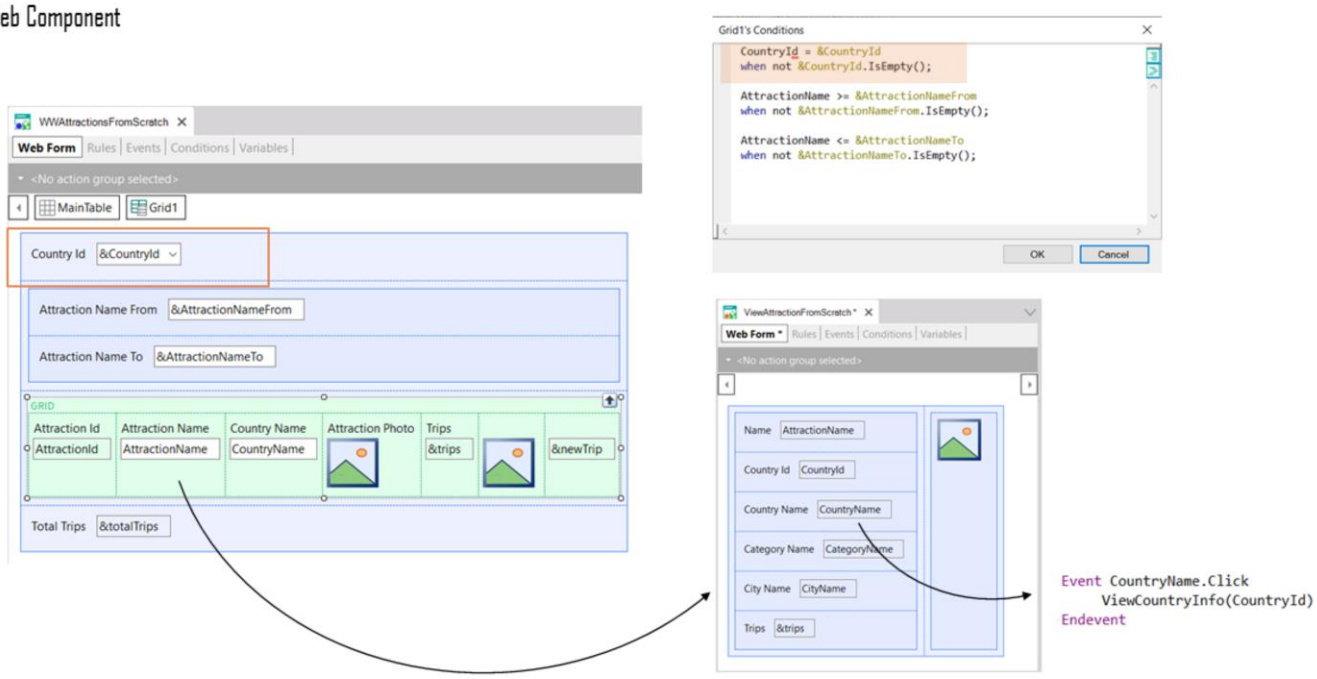




Aqui vemos o controle text block que vemos em execução com o nome Application Name. Por exemplo, vamos tentar mudá-lo para Travel Agency. Aqui temos uma imagem que também podemos mudar para esta outra que previamente inserimos na KB.

Vamos testar o efeito destas mudanças em nossa aplicação. Aqui as vemos.

Web Component

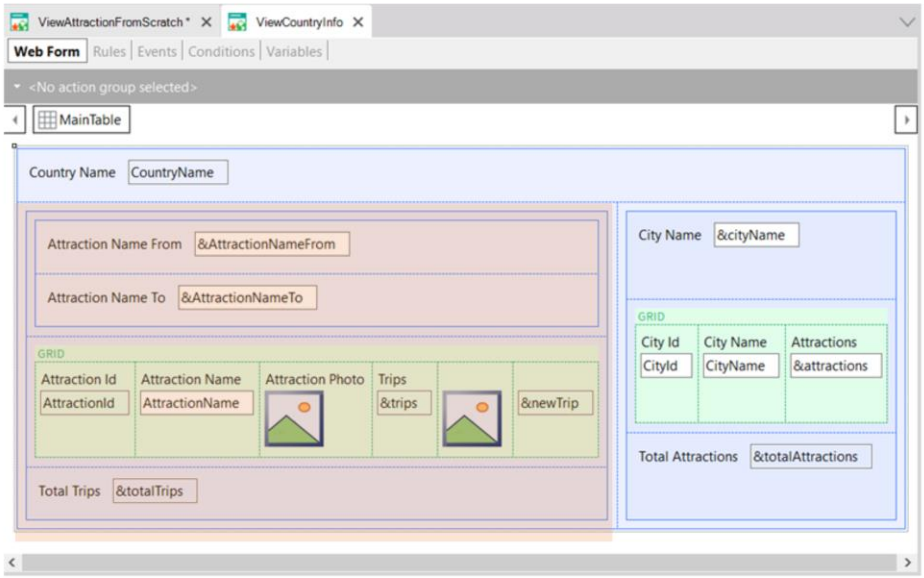


Bem, com isso já vimos dois dos três tipos de telas web: a página mestra e a página comum, que é com a qual trabalhamos até agora, cada vez que criamos um web panel ou uma transação. Nos resta ver somente a **página web** de tipo **componente**.

Para isso, voltemos um pouco sobre nossos passos e recordemos o que tínhamos implementado nos vídeos anteriores. Tínhamos a imitação do Work With de atrações, em que o usuário podia filtrar por país escolhendo um valor nesta variável, que estava sendo utilizada nas conditions do grid para filtrar por esse país, se a variável não estava vazia.

Mas além disso, a partir deste painel se permitia ao usuário ver as informações de uma atração, clicando em seu nome e, por sua vez, a partir dali apresentava-se um link sobre o nome de país, para mostrar toda a informação relevante do país.

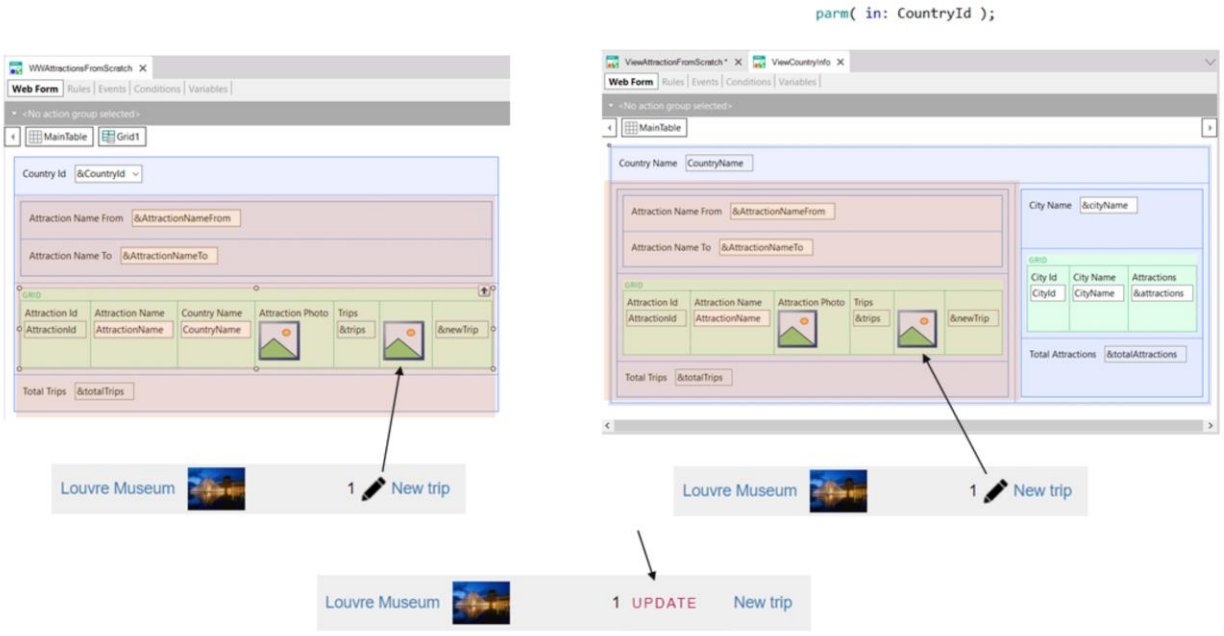
Web Component




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

Para isso chamávamos este web panel que recebia no atributo o identificador de país e mostrava seu nome, e depois, um grid idêntico ao primeiro, com os mesmos filtros por nome de atração, as mesmas colunas e ações e informações em geral e, além disso, informação das cidades.

Web Component

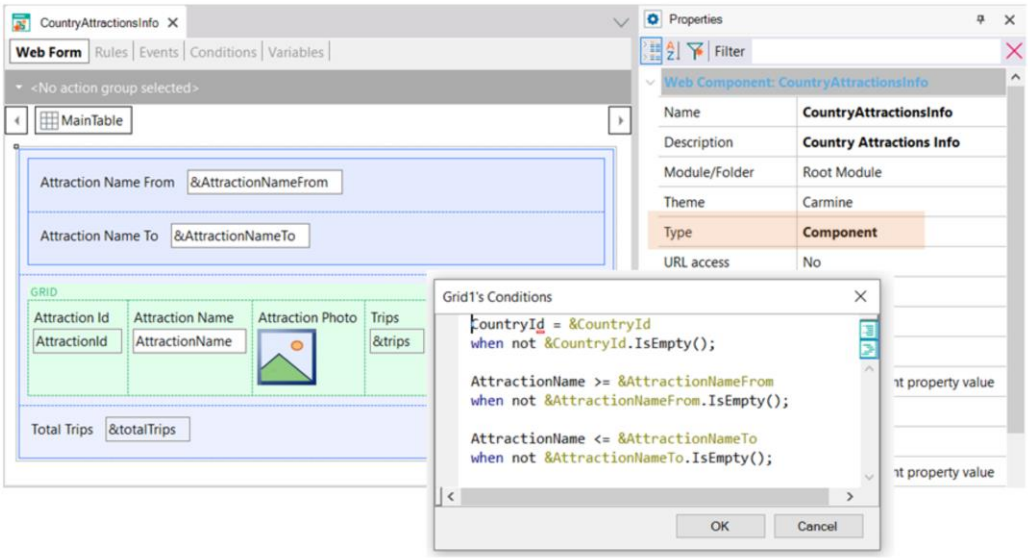


Vemos claramente que há uma parte deste painel que é praticamente idêntica à do outro. Suponhamos que já não queremos que a opção de Update seja oferecida com esta imagem, mas que queremos que seja como é no pattern, com a palavra UPDATE.

Teremos que substituir esta imagem por um text block em ambos os painéis. Para evitar estas duplicações e programar somente uma vez o comportamento e desenho de uma parte do panel, é que temos web panels de tipo componente.

Web Component

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

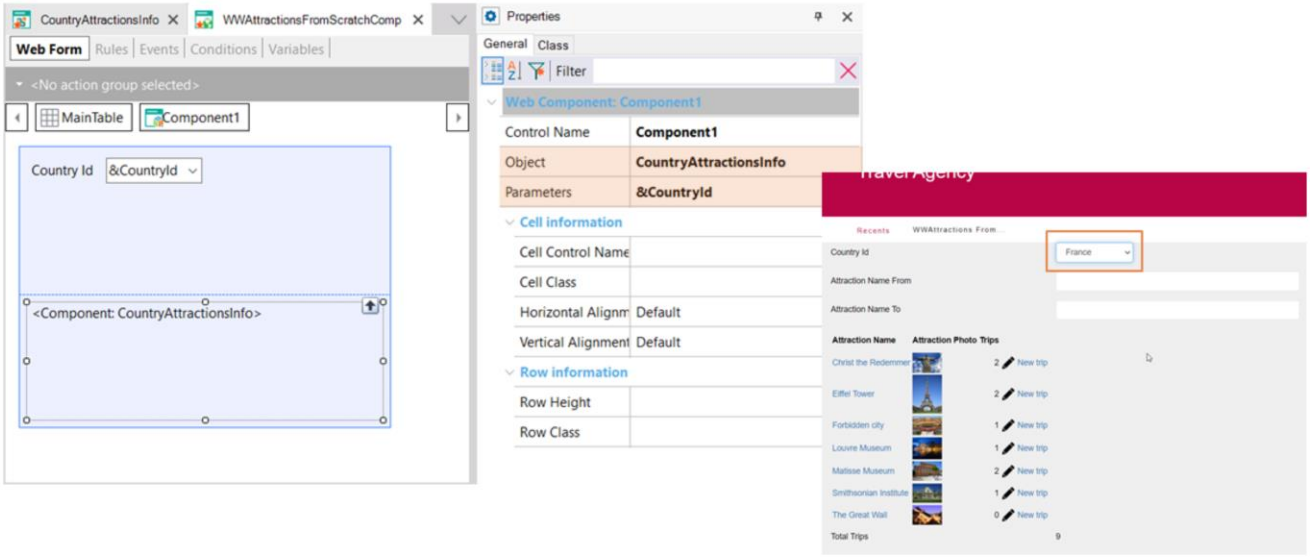


O que no nosso caso se repete? Toda esta seção da tela e seu comportamento.

Então o que fizemos foi um Save as deste web panel, no qual unicamente retiramos a variável CountryId do form e a colocamos, em troca, como parâmetro.
O usuário já não a adicionará diretamente no form desta tela, mas será recebida de quem o chame.

Vemos que as conditions se mantêm idênticas, assim como os eventos e todo o resto. A outra mudança é que modificamos a propriedade Type, passando-a para Component. A partir de agora, este web panel poderá ser um componente de outro.

Component

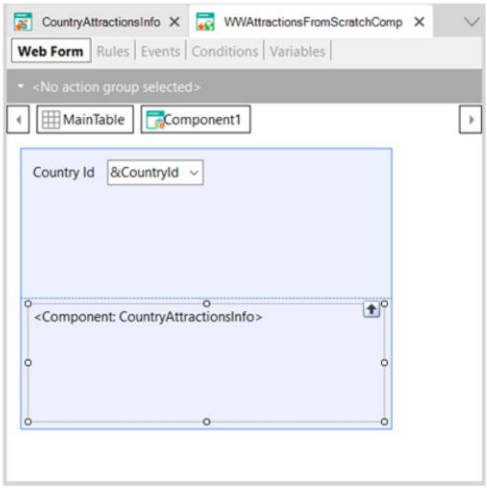


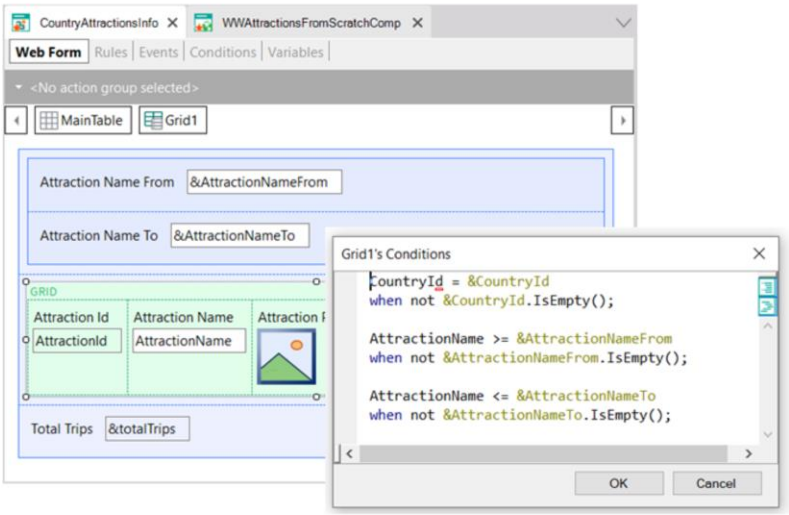
Então, a próxima coisa que fizemos foi um save as do nosso painel original, e substituímos todos esses controles que agora colocamos no web panel componente, por um controle de tipo component. Evidentemente, eliminamos (aqui deixamos comentados) todos os eventos que agora estarão no componente.

Ao colocar um controle deste tipo, estamos dizendo que ali dentro, deverá ser carregado e executado o que especifiquemos. Temos que dizer-lhe que nesse componente seja criada uma instância do web component que acabamos de mostrar, CountryAttractionsInfo. Podemos fazer de forma estática, indicando-o assim nas propriedades, onde aqui indicamos qual será o objeto de tipo componente, e aqui o parâmetro enviado. Vamos testar.

Vemos exatamente o mesmo. Se filtrarmos por nome de atração... está funcionando, perfeito.
Agora, o que acontece se queremos filtrar por país? Nada acontece. Por quê?

Component






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

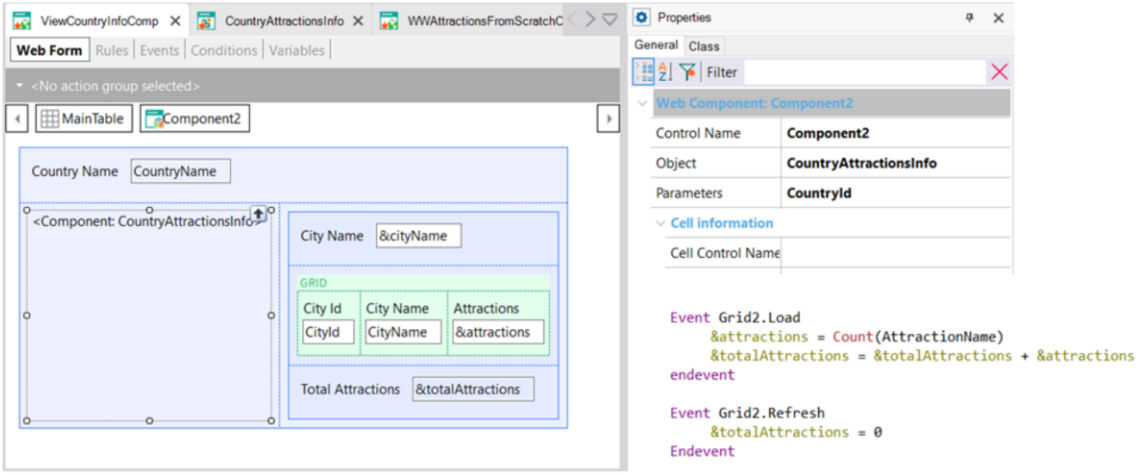
A variável CountryId está em um painel diferente do que o grid de atrações pelo qual se filtra. Aqui o que temos que fazer é utilizar o evento ControlValueChanged, que captura o momento em que se modifica o valor do controle variável &CountryId, e pedir que se crie uma nova instância do componente, passando-lhe agora o novo valor da variável.

Vamos testar

Atualizamos. Tentamos filtrar por França. Perfeito. E, lá dentro, por nome de atração. Perfeito também.

Component

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



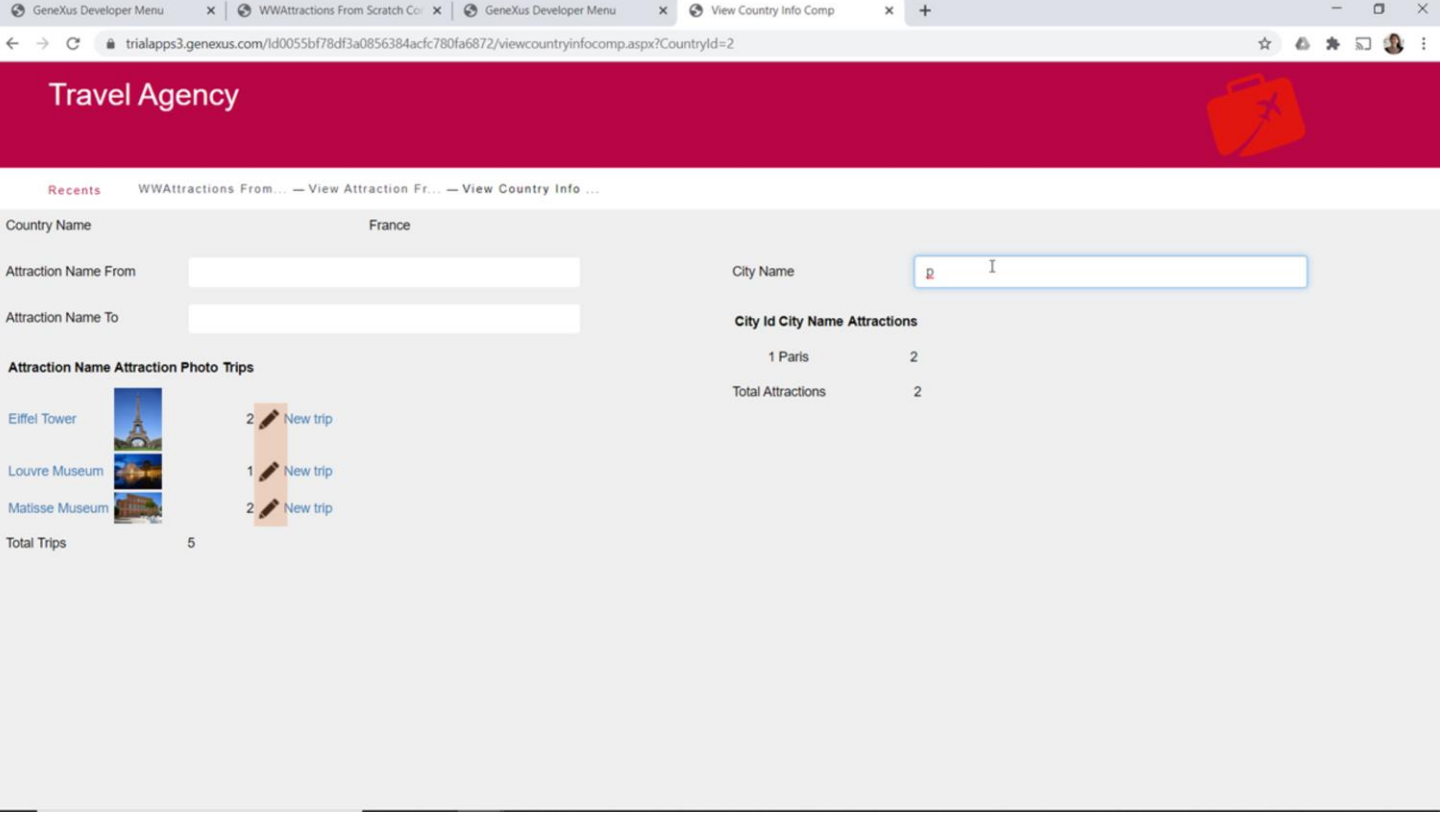
E, claro, fizemos algo similar com o painel que mostrava as informações do país.

Fizemos um Save as deste painel, no qual substituímos toda esta seção por um componente que será carregado com este painel.

Assim, no novo painel o CountryId não é variável, mas é recebido por parâmetro, e por isso podemos criar a instância do componente de forma estática, uma única vez dentro da execução deste web panel, passando-lhe aqui o parâmetro que neste caso vem no atributo recebido na regra parm.

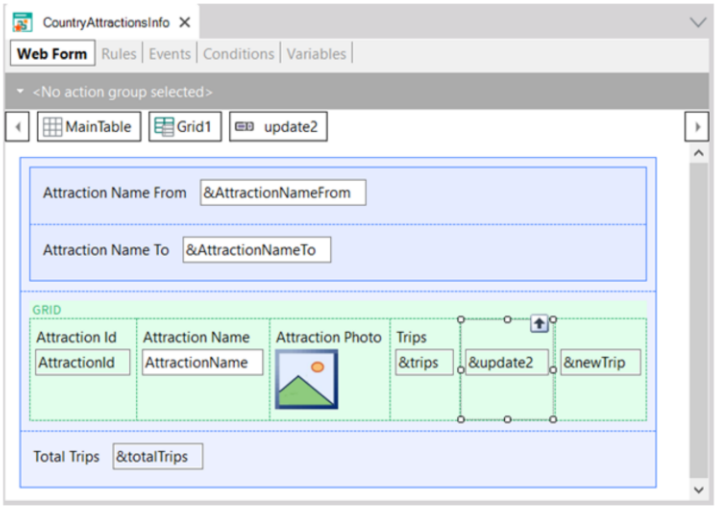
Observemos que do painel original, removemos os controles e a programação de eventos associados às atrações, e só deixamos as das cidades. Testemos em execução.

Para isso, invoquemos este painel e não o anterior.



Agora, modifiquemos a imagem para realizar o UPDATE, e coloquemos em seu lugar o texto UPDATE.

Web Component



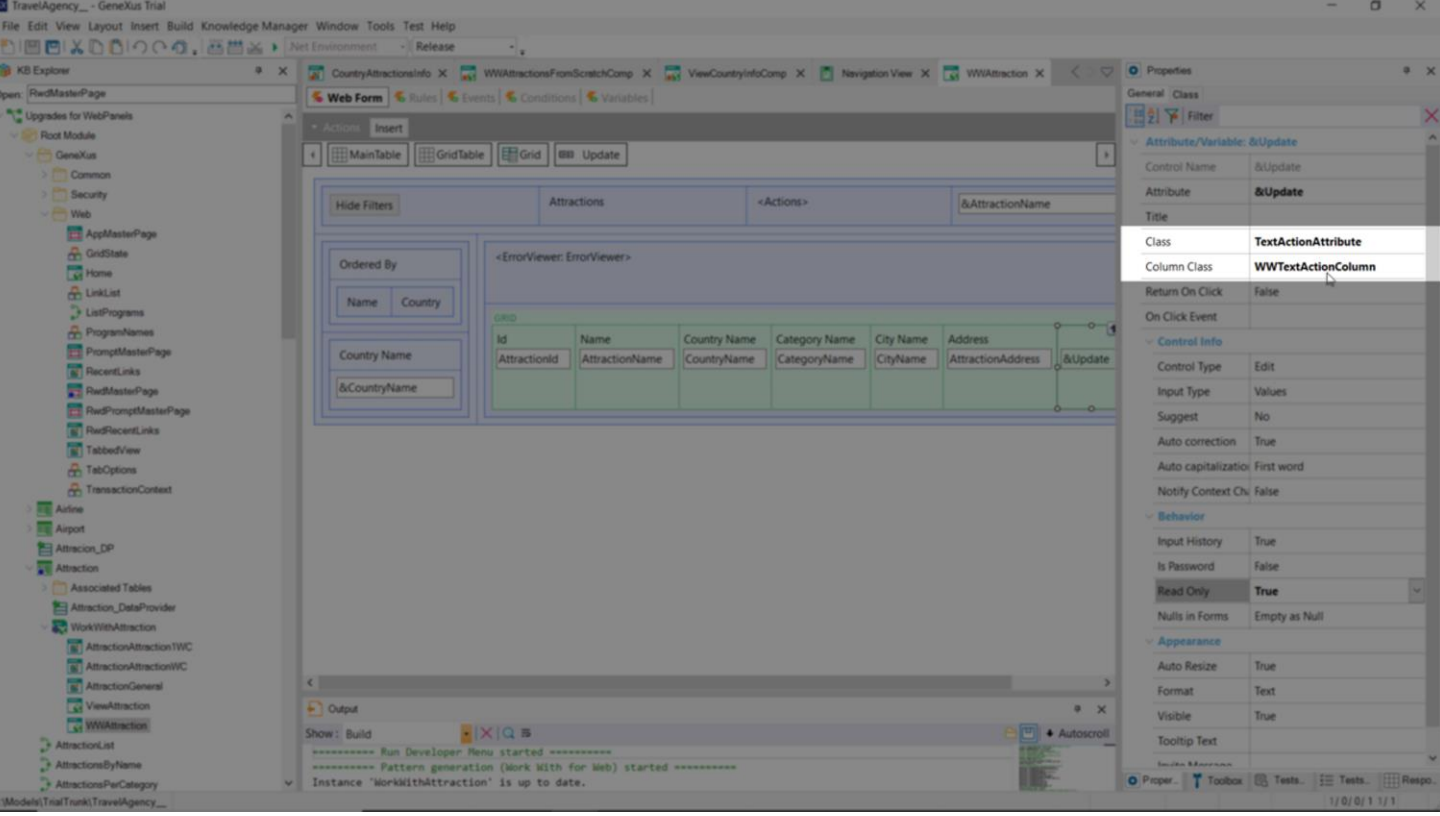

```
Event Grid1.Load
     &trips = Count(TripDate)
     &totalTrips = &totalTrips + &trips
Endevent

Event Grid1.Refresh
     &totalTrips = 0
Endevent

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

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

Para isso, no web panel componente criamos uma variável de tipo Character de 20. A inserimos no grid. No evento Start atribuímos-lhe o valor UPDATE, que é o que queremos que o usuário veja. Eliminamos a atribuição de uma imagem à variável que tínhamos, porque vamos eliminá-la do grid.

E, agora, vamos tirar o título da nova variável e torná-la Read only.

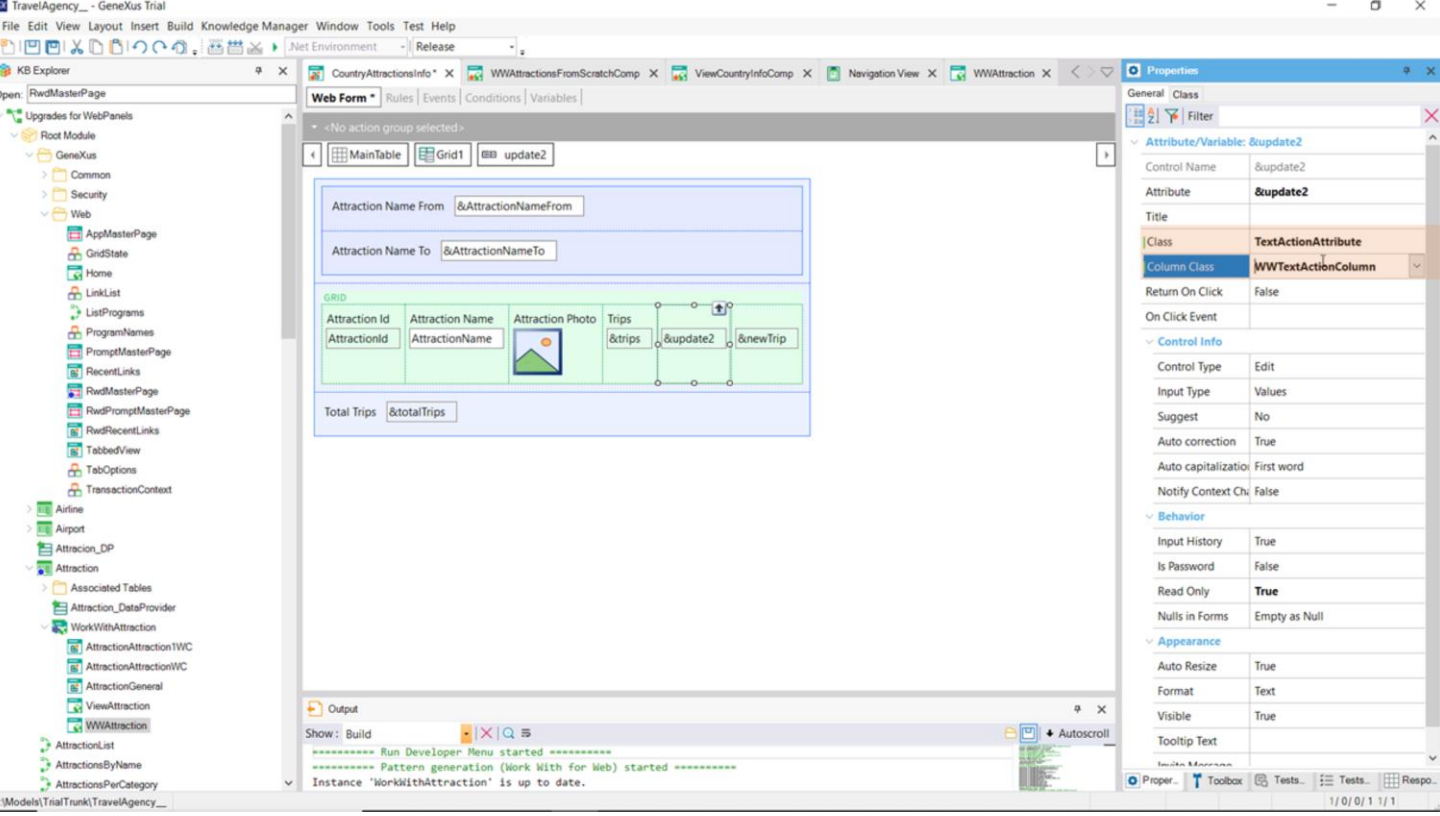
Vamos tentar executar. Nos indica que o click não é um evento válido. É que esquecemos de modificá-lo para que seja o click da nova variável e não da velha.

Agora sim... executemos.

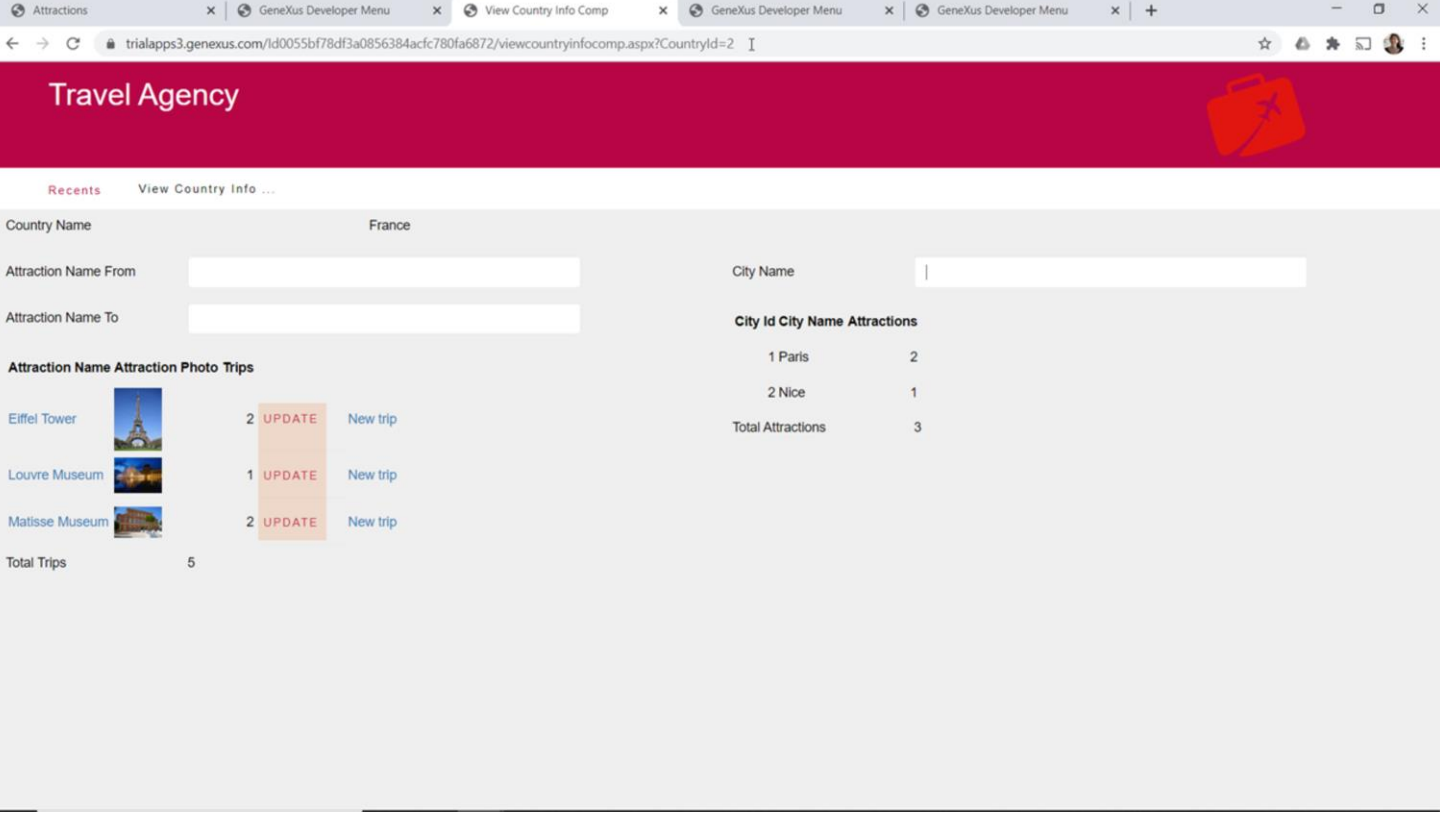


Se vamos observar como ficou no pattern, por que aqui tem este desenho tão mais bonito?

Localizemos as propriedades do controle no web panel gerado pelo pattern. Vejamos estas duas: Class e Column Class.



E vejamos os valores que essas propriedades têm para o nosso controle, o que nós inserimos manualmente em nosso web panel. São diferentes. Vamos atribuir-lhes as mesmas que as do pattern. E testemos.



Aqui o vemos. Com isto já começamos a entender como se manipula o desenho de nossas telas.

Types of Web Panels

Resumindo: Vimos três tipos de Web panels que estão relacionados entre si.

O do tipo **Master Page**, que oferece uma estrutura comum a todas as páginas da aplicação ou de uma parte dela, para não ter que repetir o mesmo cada vez, para cada página. Por exemplo, os menus costumam ir aqui. Este objeto tem a particularidade de conter em seu form um controle especial, o ContentPlaceHolder, onde as páginas web serão carregadas.

O do tipo **Web Page**, que é o que vínhamos estudando, que poderá ter associada uma Master Page determinada e só uma, visto que será carregado no ContentPlaceHolder desta.

E o do tipo **Component**, que serve justamente para reutilizar desenho e programação em diferentes objetos. Para poder fazer com que um objeto definido como Web panel de tipo componente se carregue dentro de outro objeto web, utiliza-se o controle de tipo component, que poderá ser carregado estática ou dinamicamente.

Quanto mais componentes identificarmos e utilizarmos, melhor qualidade terá a aplicação resultante.

Claro, há muito mais para estudar sobre este tema (por exemplo, o que acontece com a execução dos eventos em um panel com componentes, como se atualiza um componente, etc.). Por agora, com isto é mais do que suficiente.

GeneXus™

training.genexus.com

wiki.genexus.com

training.genexus.com/certifications