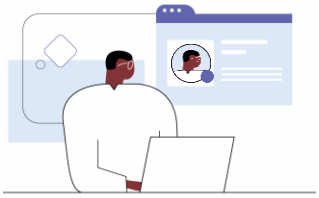
Telas web com foco em Customer-facing

Esquema de execução de eventos, gramática de eventos do cliente e determinação de tabelas base.

GeneXus

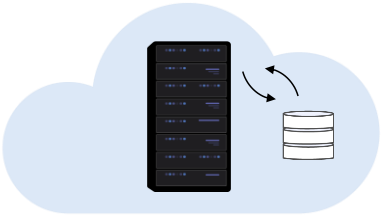
Neste vídeo começaremos a ver os eventos que podemos definir nas telas com foco em aplicações Customer-Facing, particularmente nas telas web e também focaremos em algumas características próprias do objeto panel.

Eventos do cliente e do servidor



CLIENTE WEB
com foco em UX

- ClientStart
- Back
- User Events
- Control Events



SERVIDOR

- Start
- Refresh
- Load

Este tipo de aplicações incluem dois tipos de eventos, os eventos que são disparados do lado do cliente e os eventos que são disparados no servidor.

Os eventos do servidor são os mesmos que já vimos quando estudamos webpanels, os eventos Start, Refresh e Load, que nos permitem interagir com a base de dados.
Os eventos do lado do cliente são o ClientStart, o Back, eventos definidos pelo usuário e eventos associados aos controles da tela.

No caso de que os objetos panel, os usemos para gerar uma aplicação para dispositivos móveis, teremos também outros eventos associados à plataforma móvel, por exemplo, relacionados ao tipo de dispositivo e sua orientação ao ser executada a aplicação.

Eventos do lado do servidor

SERVIDOR

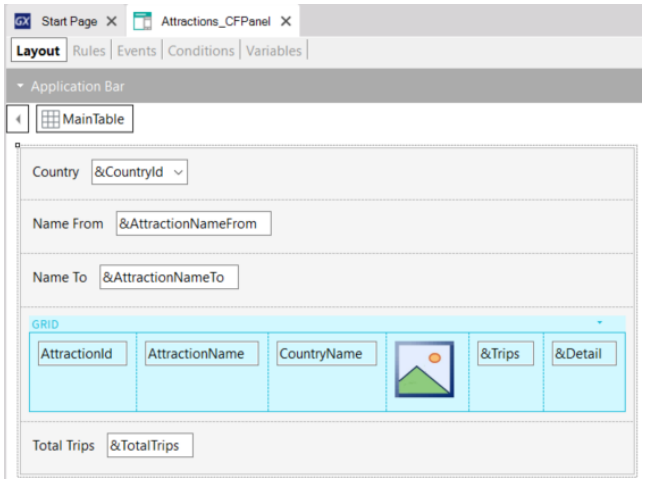
- Start
- Refresh
- Load

- Start Event execute only once.
- Refresh Event is executed after Start Event
- Refresh Event trigger Load Event
- Load Event is the last of system Events executed (only if a grid exist)
  - Grid with Base Table: Load is executed as many times as records in base Table.
  - Grids without Base Table: Load is executed only once.
  - SDT based Grids: Load is not triggered.

Veremos agora os eventos do lado do servidor.

- O primeiro que é executado é o evento Start. É executado apenas uma vez quando o panel é aberto, e não será executado novamente a menos que saiamos do objeto e voltemos a entrar nele.
- O evento Refresh é executado depois do Evento Start, normalmente uma única vez, mas pode ser invocado novamente por meio do comando Refresh, nesse caso será executado mais de uma vez e será o primeiro evento, já que o Start já não será executado novamente.
- Quando o evento Refresh for invocado, no final o evento Load será disparado. Isto é somente se houver pelo menos um grid no panel e o grid não é uma variável SDT coleção.
- O evento load é o último dos eventos do sistema a serem executados e
  - Se tem Tabela Base será executado tantas vezes quanto registros existam na tabela base.
  - Se o grid não tem tabela base, será executado apenas uma vez
  - E se o grid está baseado em um SDT, o evento load não será executado.

Exemplo de eventos do lado do servidor




```
Event Load
    &Trips = Count(TripDate)
Endevent

Event Refresh
    &TotalTrips = 0
    For each Trip.Attraction
        &TotalTrips += 1
    Endfor
Endevent

Event &CountryId.ControlValueChanged
    Grid1.Refresh()
Endevent

Event &AttractionNameFrom.ControlValueChanged
    Grid1.Refresh()
Endevent


Event &AttractionNameTo.ControlValueChanged
    Grid1.Refresh()
Endevent

Event Start
    &Details = "Details"
Endevent

Event &Details.Tap
    AttractionDetail_CFPanel.Call(AttractionId)
Endevent
```

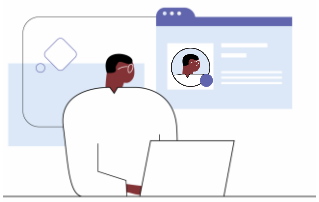
No exemplo do vídeo anterior, quando implementamos o objeto panel Attractions_CFPanel, programamos os eventos Start, Refresh e Load.

No evento Load calculamos o total de viagens de uma atração através de uma fórmula que acessava a tabela TripAttraction.

No evento Refresh, programamos um For Each para que acesse a tabela TripAttraction para contar o total global de viagens das atrações.

E também programamos o evento Start para inicializar a variável &Details com o texto que queríamos que aparecesse na tela.

Eventos do lado do cliente



CLIENTE WEB
com foco em
customer-facing

- ClientStart
- Back
- User Events
- Control Events

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

Os eventos do lado do Cliente são a resposta da aplicação à interação do usuário.

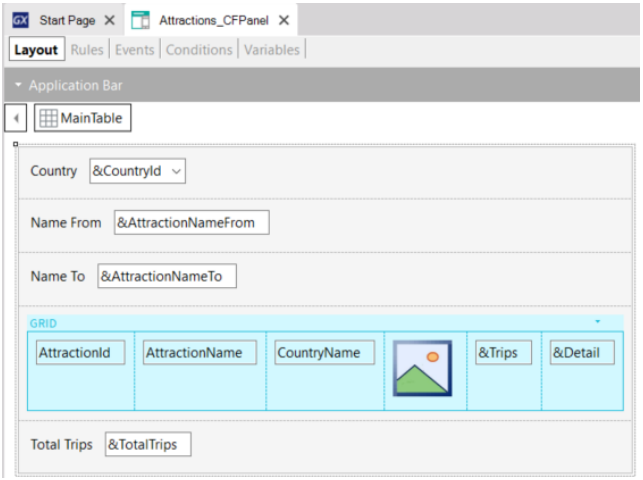
• Há três tipos de eventos de cliente: do sistema, de usuário e eventos de controles. Os eventos do sistema são o ClientStart e o Back, que não veremos neste curso. Os eventos de usuário são os criados pelo usuário e os eventos dos controles são os associados aos controles da tela, como os eventos Tap (que corresponde ao clique), Double Tap (que é um duplo clique) ou ControlValueChanged, entre outros

• O código associado a estes eventos é executado no cliente, a menos que seja requerido acessar um serviço do servidor, por exemplo, se quiser acessar a base de dados

• Durante a execução de um evento do lado do cliente, os eventos do lado do servidor não são executados, a menos que sejam requeridos explicitamente através do comando Refresh.

• Estes eventos do lado do cliente, terão uma gramática particular diferente dos eventos do lado do servidor.

Exemplos de eventos do lado do cliente




```
1  Event Load
2      &Trips = Count(TripDate)
3  Endevent
4
5  Event Refresh
6      &TotalTrips = 0
7      For each Trip.Attraction
8          &TotalTrips += 1
9      Endfor
10 Endevent
11
12 Event &CountryId.ControlValueChanged
13     Grid1.Refresh()
14 Endevent
15
16 Event &AttractionNameFrom.ControlValueChanged
17     Grid1.Refresh()
18 Endevent
19
20
21 Event &AttractionNameTo.ControlValueChanged
22     Grid1.Refresh()
23 Endevent
24
25 Event Start
26     &Details = "Details"
27 Endevent
28
29 Event &Details.Tap
30     AttractionDetail_CFPanel.Call(AttractionId)
31 Endevent
```

No mesmo objeto que vimos antes, programamos somente eventos associados a controles, não programamos nenhum evento de usuário.
A cada variável dos filtros em tela, programamos seu evento ControlValueChanged, para disparar o método Refresh do Grid, o que provocará que se disparem os eventos Refresh e Load do servidor para atualizar o conteúdo do grid com os filtros inseridos.

Também programamos o evento Tap da variável &Details para invocar o panel AttactionDetail_CFPanel, para ver o detalhe da atração selecionada, cujo identificador AttractionId passamos por parâmetro.

Gramática de eventos do lado do cliente

COMMANDS

Composite

<*Control*>.<*Property*> = <*value*>

If **<Bool_expr>**
Do case... endcase
Do while <Bool_expr>
Do-sub (except Menu for Smart Devices)

For each selected line

Simple variable assignation: &var = ***<expr>***

SDT or BC elements assignation:
- &SDT.A = <value>
- &BC.A = <value>

Return
Refresh

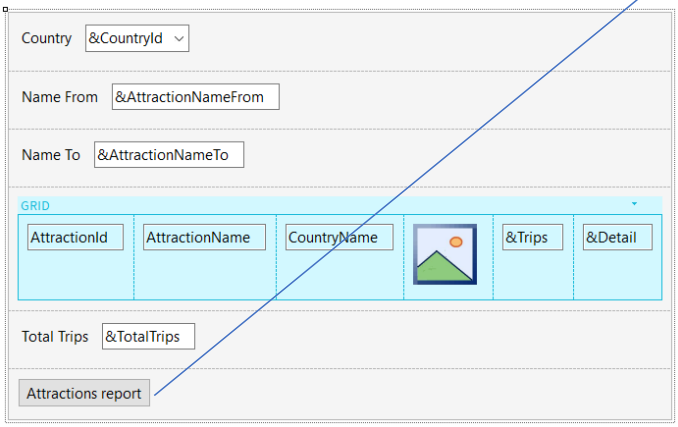
Inside an **expression**:
- Variables
- Attributes
- Constants
- Methods
- Functions
- Control properties
- Operators (+, -,, /, ^)

Vejamos agora a gramática de eventos do lado do cliente.

Nos eventos do lado do cliente, há restrições quanto aos comandos que podem ser utilizados, não é assim para os eventos do lado do servidor.

Os comandos aceitos no momento são os que são mostrados. Não entraremos em mais detalhes neste curso,

Comando Composite




```
Event "Attractions report"
     Composite
          AttractionsByName("C", "M")
          Return
     EndComposite
Endevent
```

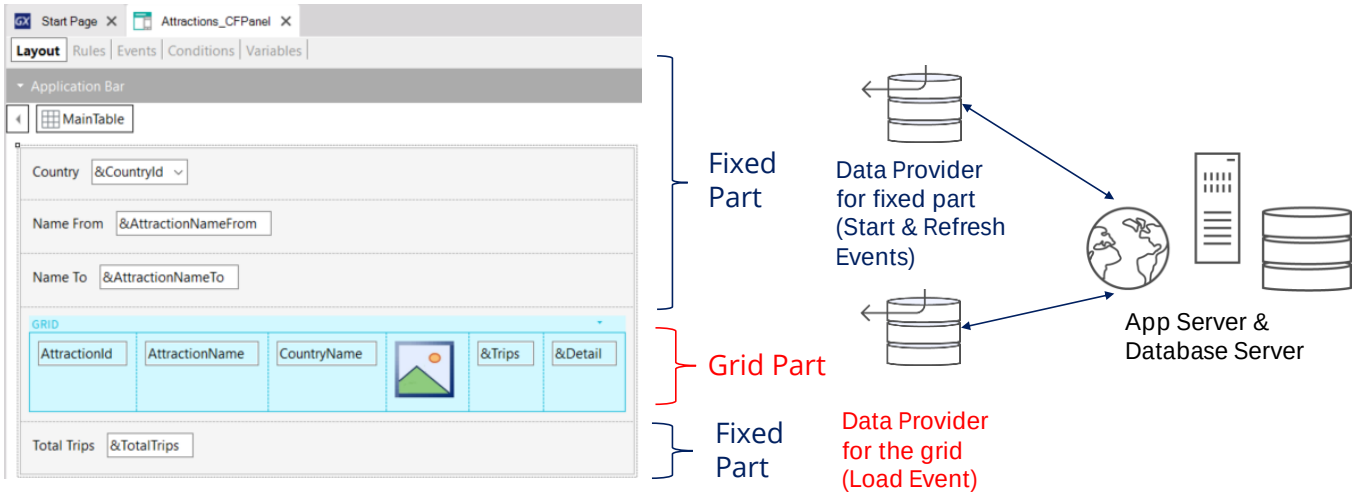
- Only for Client Side Events with more than 1 line of Code.
- Stop execution on Error.
- Automatic Error Handling.

O comando Composite se utiliza em eventos do lado do cliente em objetos panel, quando deve-se escrever duas ou mais linhas em um evento, e neste caso é obrigatório agrupar o código completo do evento dentro deste comando.

A importância deste comando é que, quando ocorre um erro na sequência de chamadas, a execução se interrompe e os erros são tratados automaticamente, enviando-os para a tela sem ter que implementar nenhuma programação.

Esta é uma grande diferença com os webpanels, dado que nestes quando dentro de um evento um objeto chamado produz um erro, não se interrompe a execução, segue na sentença seguinte e é o desenvolvedor quem deve encarregar-se de tratar os erros e programar as ações a tomar.

Partes de um objeto panel



Vejamos agora algumas características do objeto panel.

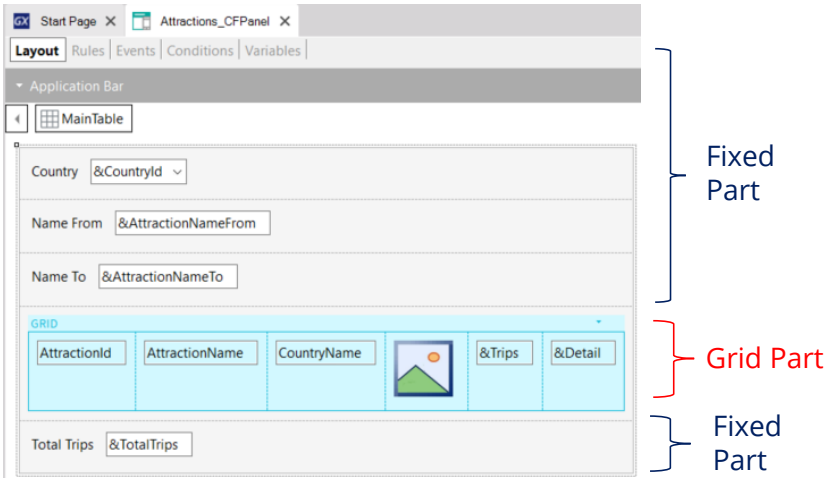
Como mencionamos antes, nos objetos panel podemos identificar duas partes distintas, a primeira parte que denominamos a parte fixa ou plana que contém tudo o que está no formulário e que não esteja incluído em nenhum grid. No exemplo que vemos em tela, a parte fixa é composta pelas variáveis dos filtros: &CountryId, &AttractionNameFrom, &AttractionNameTo e o total de viagens &TotalTrips.

A segunda parte denominaremos parte do grid ou variável, e será composta, neste caso, pelo grid que contém os dados das atrações.

Sempre teremos uma parte fixa, e podemos ter uma parte variável para cada um dos grids que tenha o panel.

Para cada parte (fixa e grid) GeneXus vai gerar, automaticamente, Data Providers que serão publicados como serviços no servidor e resolverão o acesso aos dados. Estes data providers, não os veremos na Base de Conhecimentos já que GeneXus será responsável por gerá-los e mantê-los, mas poderemos ver que dados cada um acessa, se vemos sua lista de navegação, como fizemos antes.

Ordem de execução dos eventos



- ClientStart
- *Call to Start & Refresh Data Provider*
- Start
- Refresh

- ***Fixed Part is Drawn***

- *Call to Grid Data Provider*
- Load

- ***Grid Part is Drawn***

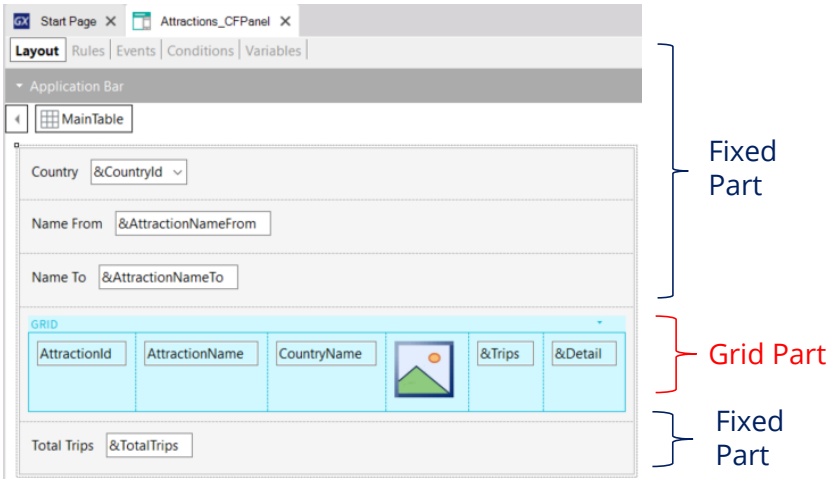
Agora vejamos o que acontece quando executamos um objeto panel.

Primeiro, executa-se o evento ClientStart, executa-se só uma vez e roda, como já vimos no cliente, . Em seguida, executa-se um data provider que retornará os dados necessários para carregar a parte fixa do panel. Este data provider integra a execução do código dos eventos Start e Refresh que serão executados no servidor e retorna em um único resultado, as informações para carregar a parte fixa. Então a parte fixa do panel será desenhada.

Em seguida, executa-se um segundo Data Provider que resolverá a recuperação dos dados requeridos pelo grid. Dentro da execução deste data provider, é executado o código do evento Load no servidor. Este evento Load será executado N vezes quando o grid possuir tabela base, uma vez para cada registro e apenas uma vez se o grid não tem tabela base.
Ao finalizar a execução do data provider, o mesmo retornará as informações geradas por todas estas execuções do evento Load em um único resultado, com o qual será carregado grid e depois, terminará de desenhar o grid.

A particularidade de que a tela seja desenhada em dois momentos distintos tem suas consequências como veremos a seguir.

Determinação de tabelas base da parte fixa e do grid



Attributes involved in determining of **Fixed Part base table**:

- Attbs in fixed part of panel (form)
- Attbs outside For Each in events: Refresh, Buttons or controls in fixed part and Application Bar
- Attbs in Conditions Tab

Attributes involved in determining of **Grid Part base table:**

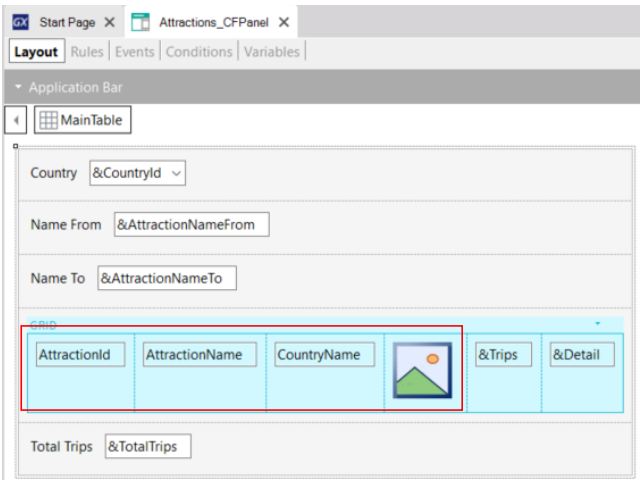
- Attbs in grid columns
- Attbs in Order, Search, Advanced Search and Conditions
- Attbs outside For Each in events: Load, Buttons or controls inside the grid
- Attbs in Conditions Tab

Em um objeto panel a parte fixa e o grid determinam navegações independentes e cada parte terá sua tabela base, como se houvesse dois for eachs paralelos. Isto estabelece uma diferença importante com os webpanels.

Para determinar a tabela base da parte fixa, serão considerados os atributos que pertencem à parte fixa do form, os atributos que pertençam aos eventos associados à parte fixa contanto que estejam fora de um comando For Each (no evento Refresh e eventos associados a botões ou controles da parte fixa, incluídos os da Application Bar), e atributos da Tab Conditions do objeto Panel.

Para determinar a tabela base do grid, serão levados em conta os atributos incluídos nas colunas do grid, tanto visíveis como não visíveis, os atributos referenciados no Order, Search, Advanced Search e Conditions do grid e os atributos fora de cláusulas For Each que estejam no evento Load e nos eventos de botões ou controles dentro do grid, e atributos da Tab Conditions do objeto Panel.

Determinação de tabelas base da parte fixa e do grid



Fixed Part: No base table

Fixed Part

Grid Part

Fixed Part

Grid Part base table: Attraction

```
1  Event Load
2      &Trips = Count(TripDate)
3  Endevent
4
5  Event Refresh
6      &TotalTrips = 0
7      For each Trip.Attraction
8          &TotalTrips += 1
9      Endfor
10 Endevent
11
12 Event &CountryId.ControlValueChanged
13     Grid1.Refresh()
14 Endevent
15
16 Event &AttractionNameFrom.ControlValueChanged
17     Grid1.Refresh()
18 Endevent
19
20
21 Event &AttractionNameTo.ControlValueChanged
22     Grid1.Refresh()
23 Endevent
24
25 Event Start
26     &Details = "Details"
27 Endevent
28
29 Event &Details.Tap
30     AttractionDetail_CFPanel.Call(AttractionId)
31 Endevent
```

Portanto, no exemplo que vimos, como no formulário não há atributos na parte fixa (apenas variáveis), não há botões, não há atributos na Tab Conditions do panel, também não há atributos nos eventos associados às variáveis e no evento Refresh não há atributos fora do For Each, a parte fixa do panel não tem tabela base.

No grid temos presentes os atributos AttractionId, AttractionName, CountryName e AttractionPhoto, pelo que a tabela base será Attraction, já que no evento Load o único atributo presente está na fórmula Count.

Mais informações....

https://wiki.genexus.com/commwiki/servlet/wiki?24829

Se quer saber mais sobre os temas tratados neste vídeo, você pode consultar a documentação relacionada à programação para dispositivos móveis nativos (Smart Devices), já que, como dissemos antes, o objeto panel também é utilizado em tais aplicações.

Você pode encontrar mais informações na página da wiki que aparece na tela.

training.genexus.com
wiki.genexus.com