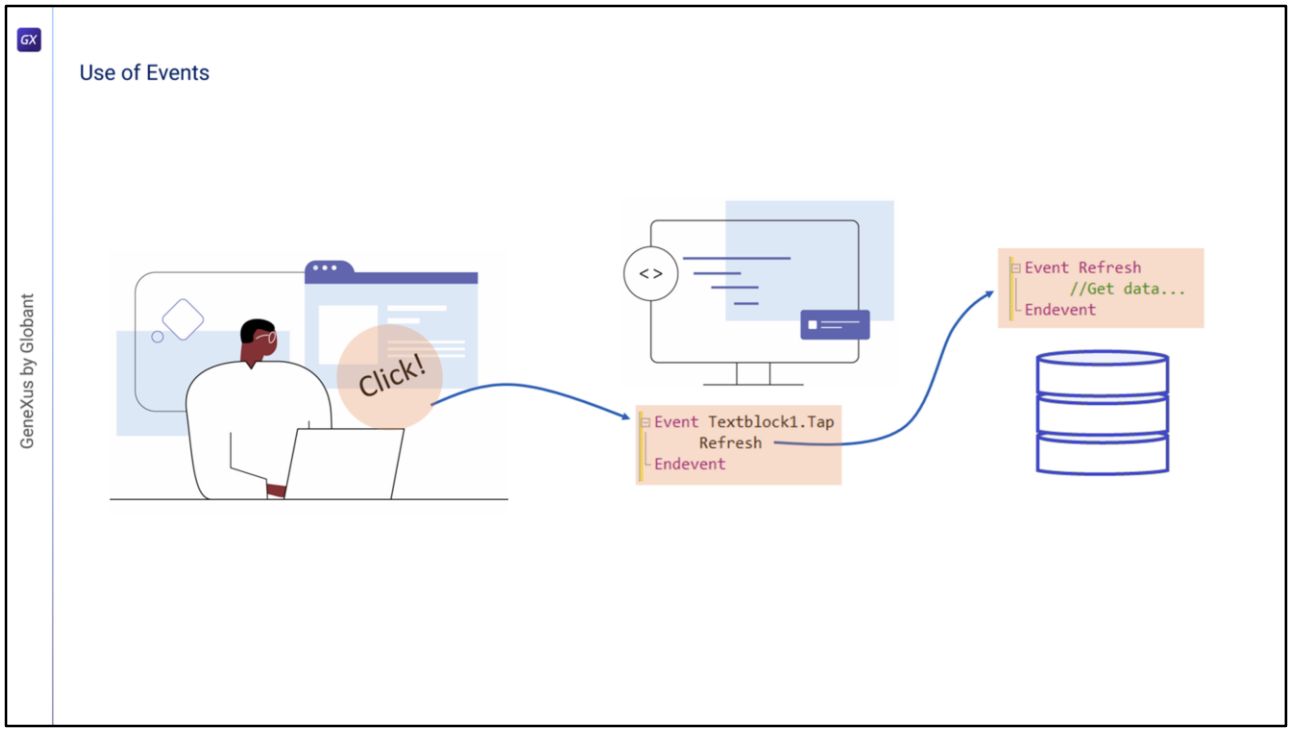
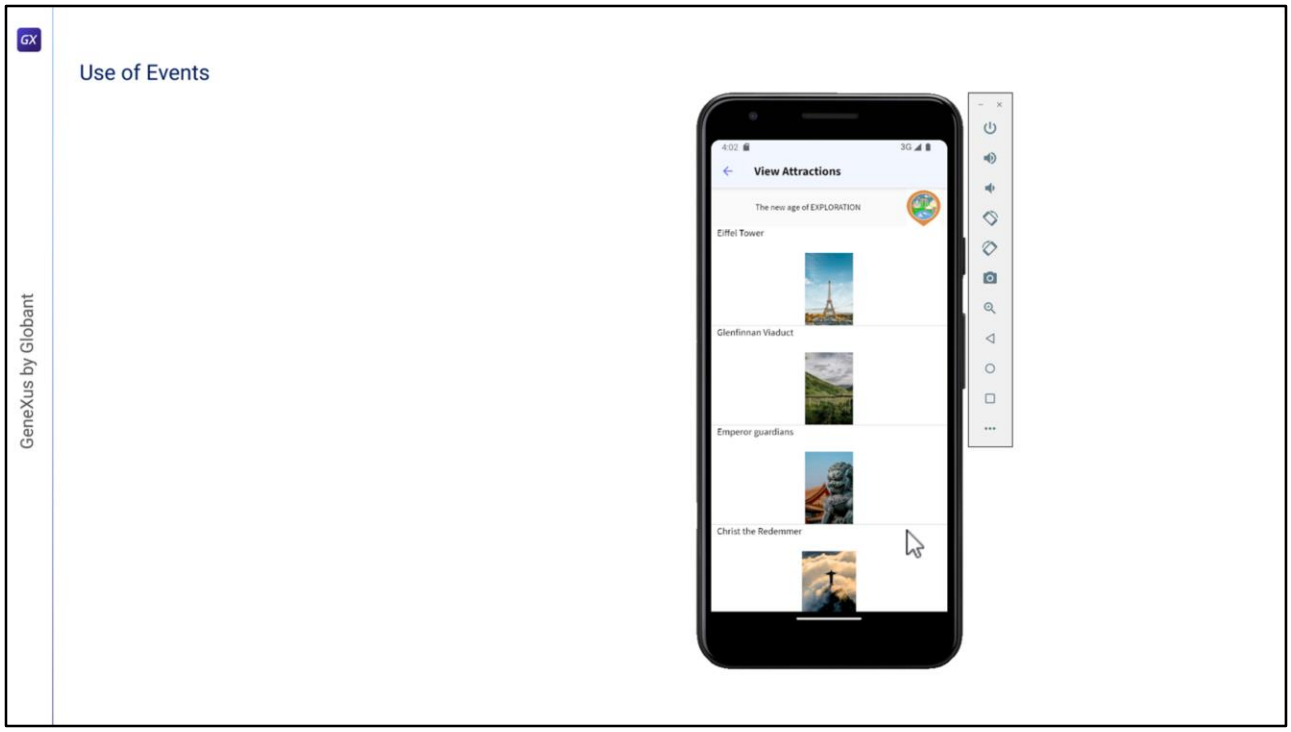
GeneXus by Globant

Events in a Panel

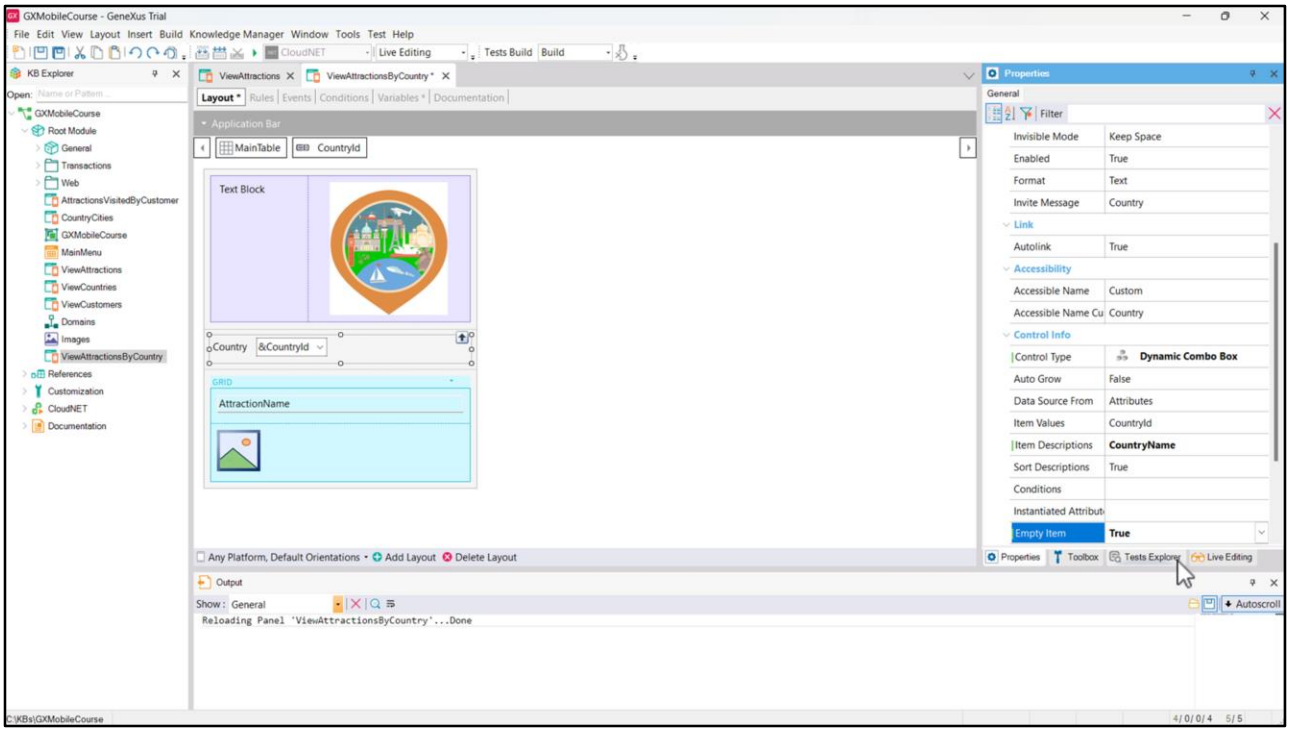
Vanesa Fernández



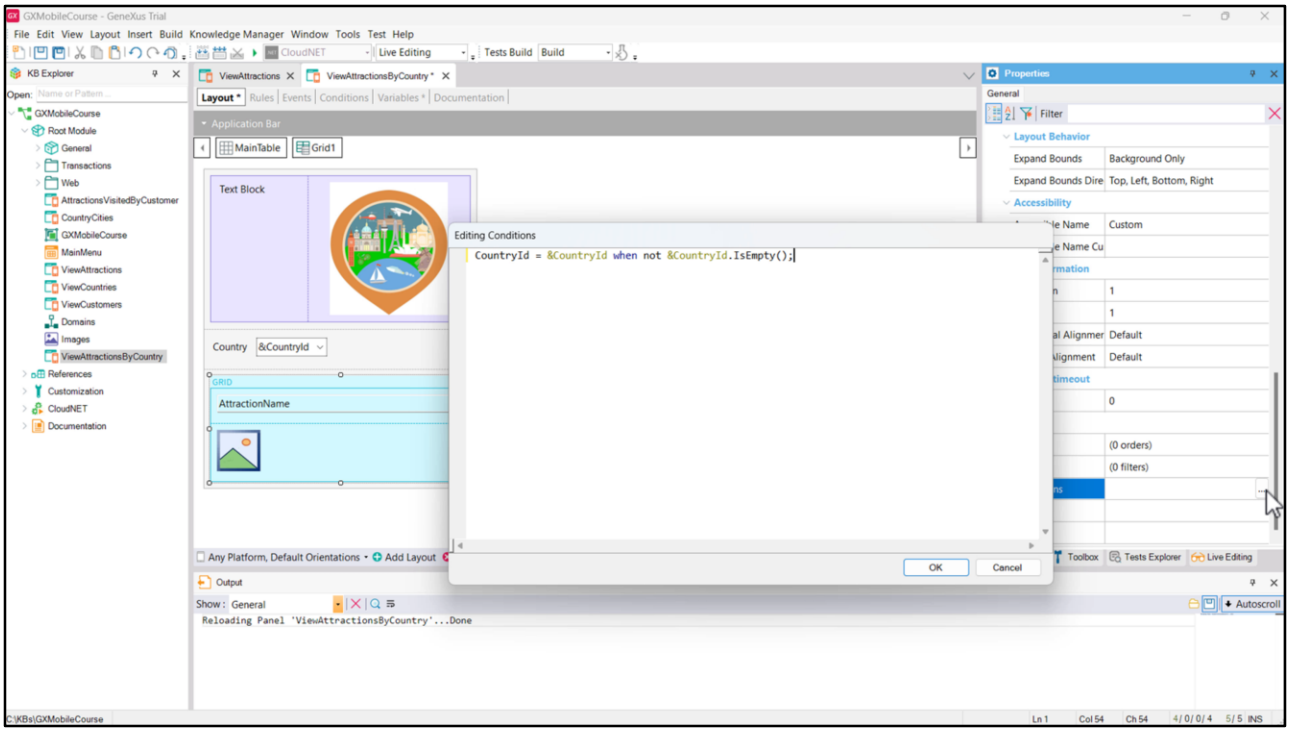
Como já sabemos, os eventos são mensagens que o software ou o sistema nos envia em determinados momentos da execução da nossa aplicação e isso nos permite programar ações a serem executadas nesses momentos. Por exemplo, se o usuário clica em um botão, ou se escreve algo em um controle de texto e sai dele, ou se é necessário que o servidor envie informação atualizada, podemos programar uma resposta após o disparo do evento.



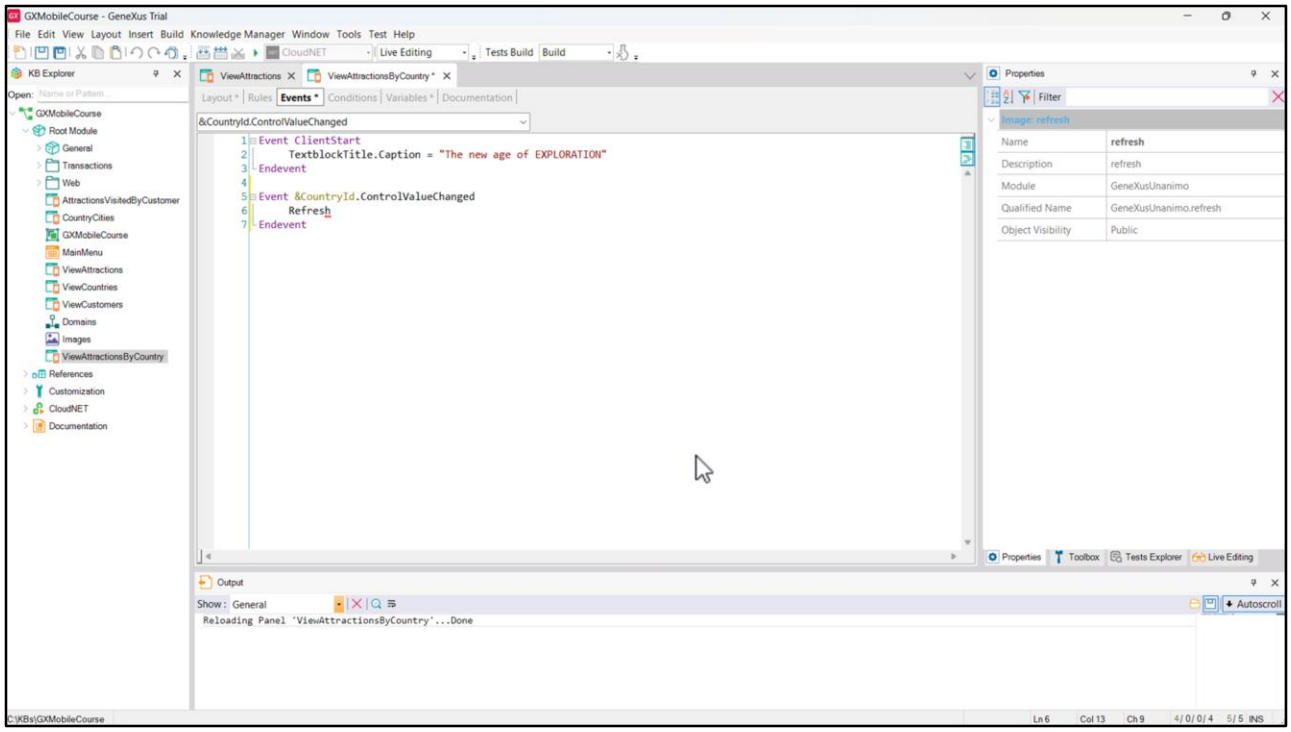
Suponhamos que quando vemos os dados das atrações, queremos ter filtros para que os dados que são exibidos estejam de acordo com a(s) condição(ões) escolhida(s).



Para implementar isso, abrimos o Panel ViewAttractions e o salvamos com o nome ViewAttractionsByCountry. Agora vamos para a seção de variáveis e adicionamos uma variável chamada &CountryId que automaticamente fica baseada no atributo CountryId. A arrastamos para o form após o textblock do título e antes do grid, e em suas propriedades alteramos o rótulo de CountryId para Country, alteramos a propriedade ControlType para Dynamic Combo e colocamos Item Descriptions em CountryName. Atribuímos também para a propriedade EmptyItem o valor True, para que no início apareça o combo sem nenhum valor padrão.

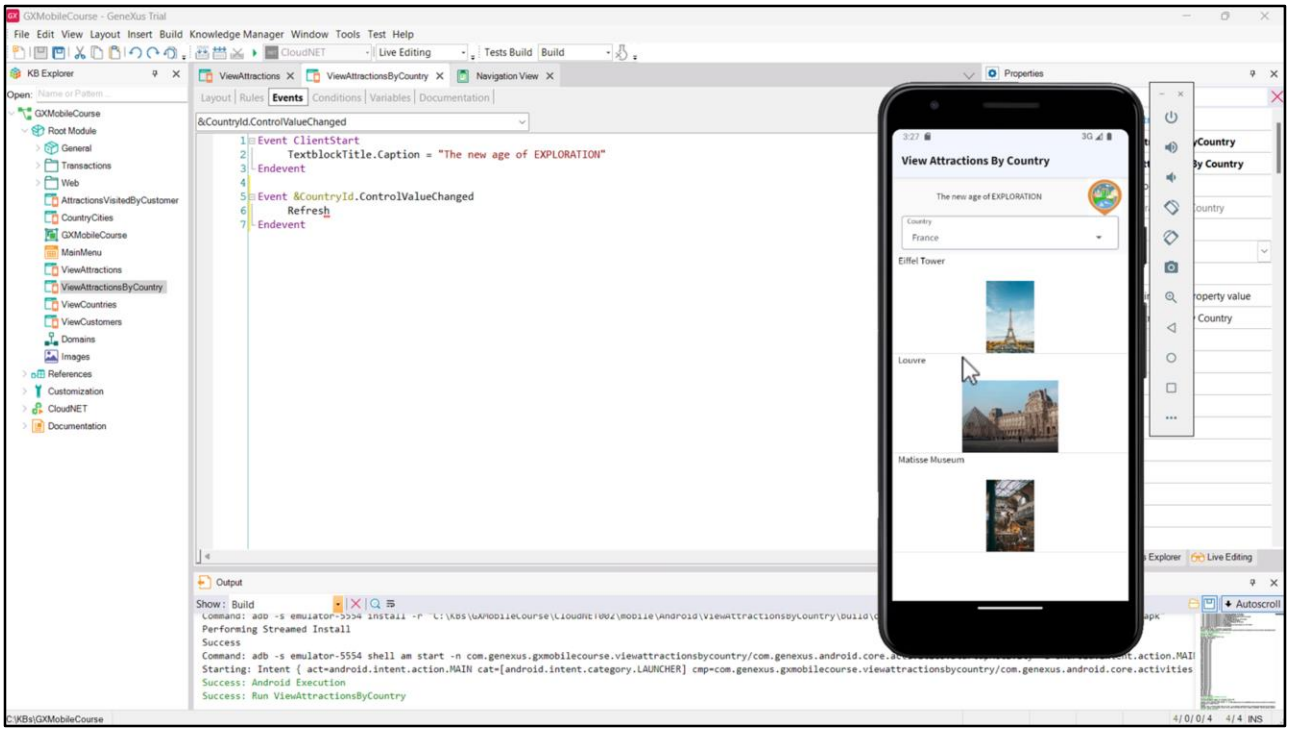


Agora, para que o filtro tenha efeito nas atrações que vemos no grid, editamos suas propriedades e na propriedade Conditions escrevemos: CountryId = &CountryId when not &CountryId.IsEmpty() e fechamos com ponto e vírgula. Assim que escolhermos um país, o Panel deve ser atualizado para que o grid carregue apenas aquelas atrações que sejam desse país.



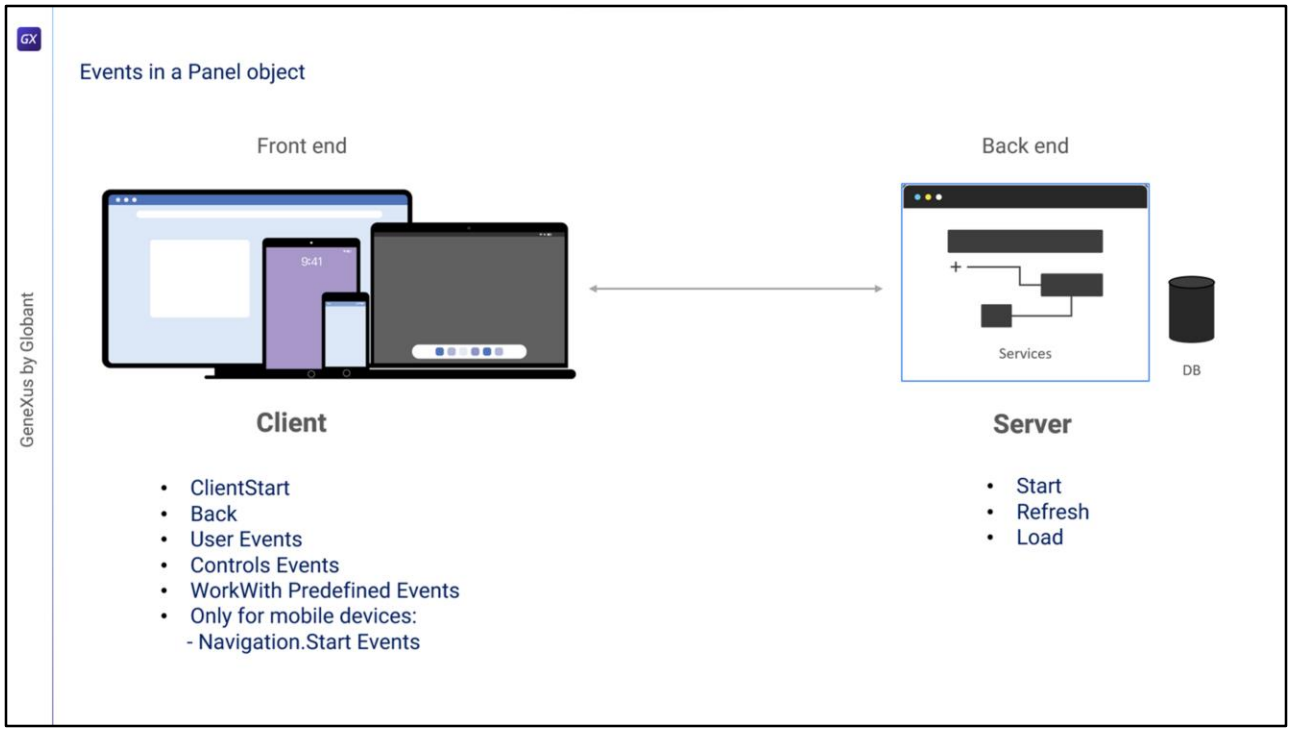
Para isso usamos o evento ControlValueChanged do combo dinâmico, então vamos para a aba de Events e no menu escolhemos Insert event, clicamos sobre a variável &CountryId e selecionamos o evento que dissemos.

Vemos que é aberto o código do evento para que escrevamos o que queremos que seja executado quando for disparado este evento. Escrevemos Refresh. Este comando fará com que o grid obtenha novamente os dados, desta vez filtrados, pelo país escolhido.



Vamos definir a propriedade Main program do Panel como True, para clicar com o botão direito e depois Run. Vemos que, como colocamos a propriedade Empty Item como True, aparece o combo sem um país selecionado e são exibidas todas as atrações. Se escolhermos a França, vemos que o grid é recarregado e agora mostra apenas as atrações turísticas da França.

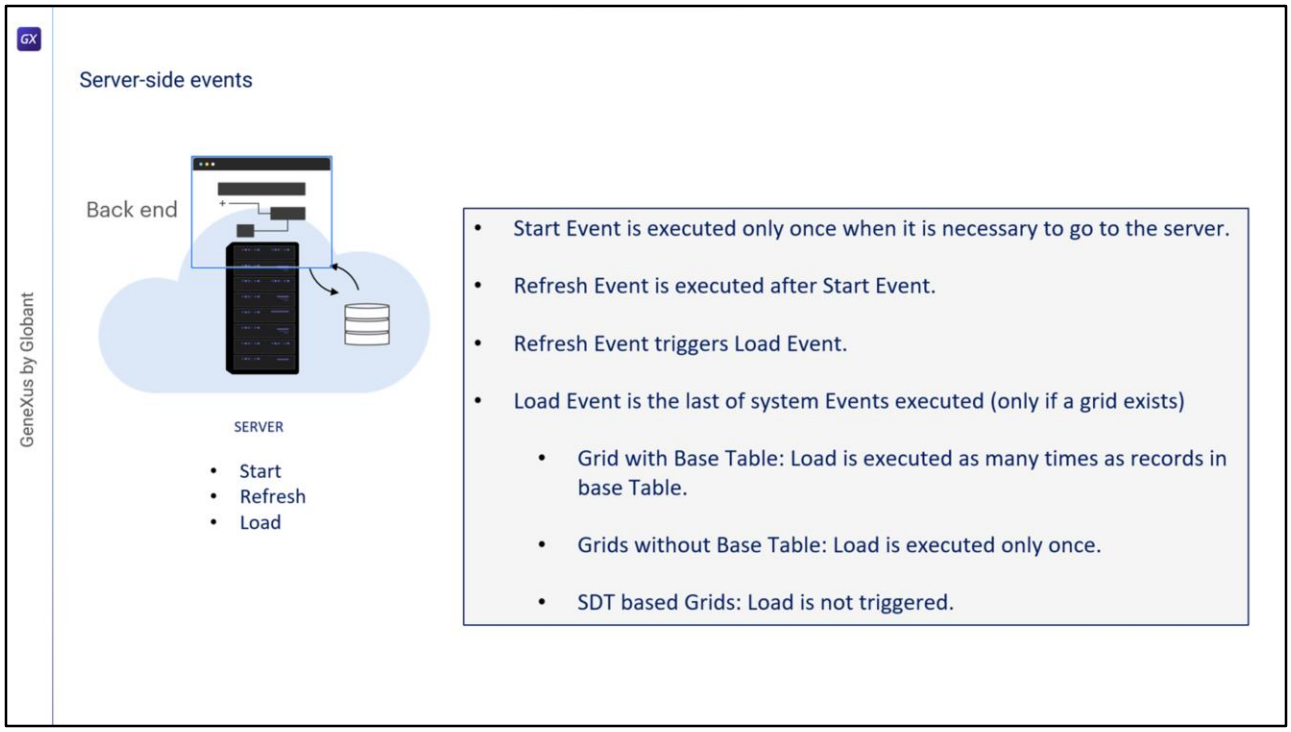
Vamos nos aprofundar um pouco mais no tema dos eventos.



Em um objeto Panel temos dois tipos de eventos: aqueles que são disparados no lado do cliente e aqueles que são disparados no servidor.

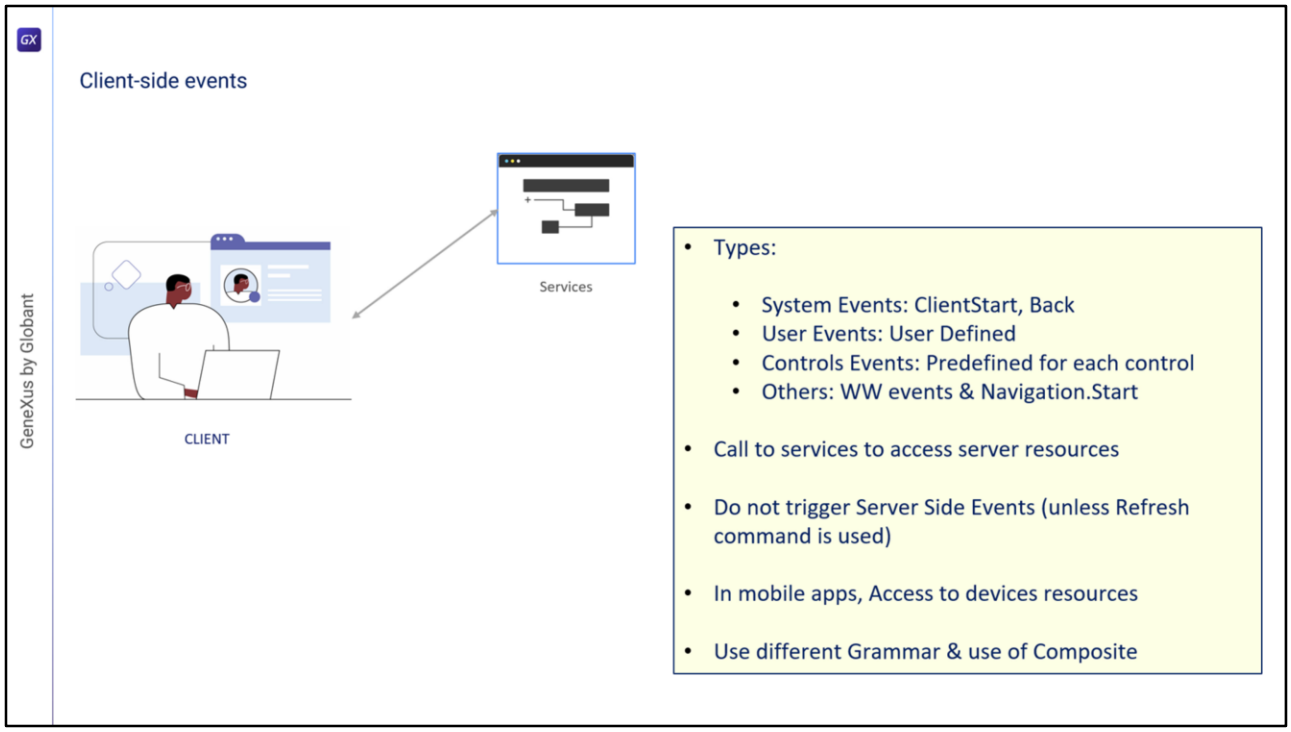
Os eventos do servidor são os mesmos de quando trabalhamos com WebPanels: os eventos Start, Refresh e Load. Estes eventos sempre serão disparados no servidor no caso de aplicações móveis nativas online. Caso estejamos implementando aplicações móveis nativas desconectadas, serão disparados internamente no dispositivo.

Os eventos do lado do cliente são: o ClientStart, o Back, eventos definidos pelo usuário, eventos associados aos controles da tela e também os eventos predefinidos pelo padrão WorkWith. Também teremos os eventos associados aos estilos de navegação da informação, que dependem, por exemplo, do tipo de dispositivo e da sua orientação quando é iniciada a aplicação.



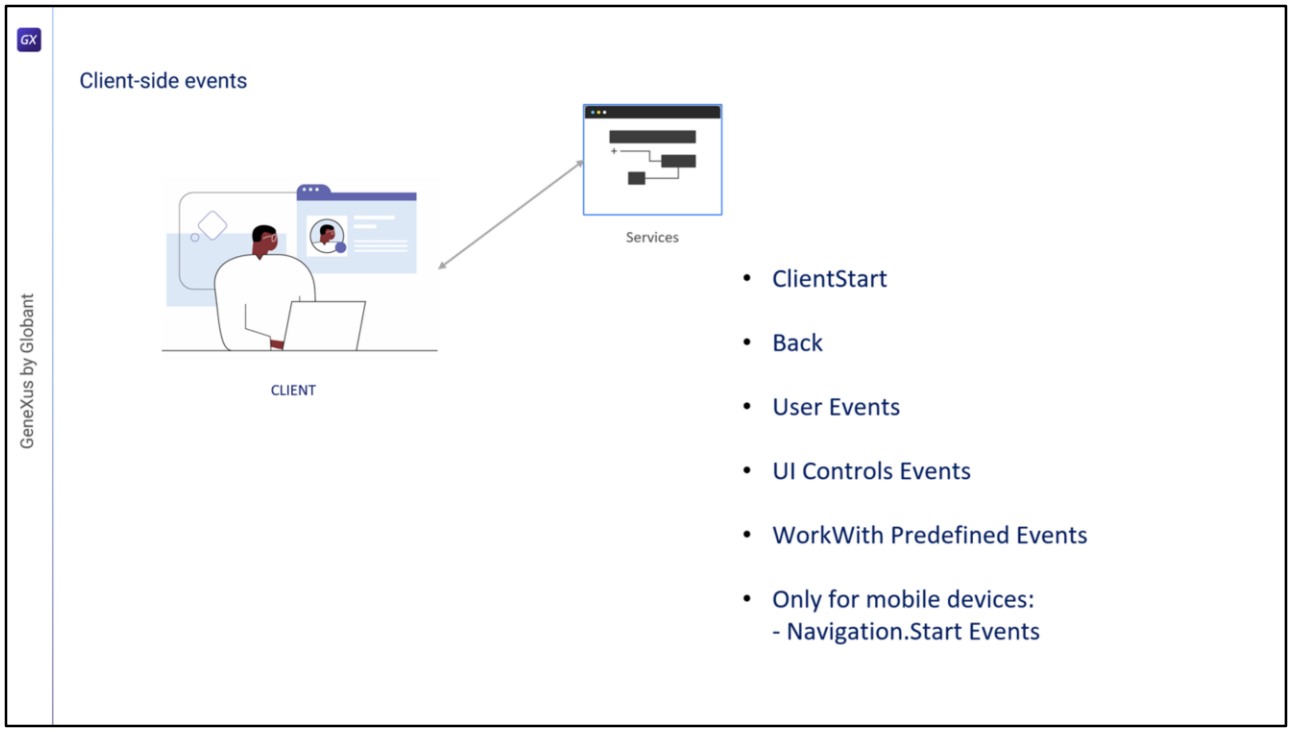
Vejamos com mais detalhes os eventos do lado do servidor.

• O primeiro que é executado é o evento Start. É executado apenas uma vez quando é aberto o Panel, desde que seja necessário ir ao servidor, e não será executado novamente a menos que saiamos do objeto e entremos nele novamente.
• O evento Refresh é executado após o evento Start, normalmente apenas uma vez, mas pode ser invocado novamente através do comando Refresh. Neste caso, será executado mais de uma vez e será o primeiro evento, pois o Start não será executado novamente. O evento Refresh também será disparado se fizermos um refresh do navegador.
• Quando o evento Refresh é invocado, após a conclusão será disparado o evento Load. Isso ocorre somente se houver pelo menos um grid no Panel e o grid não for uma variável SDT coleção.
• O evento Load é o último dos eventos do sistema a ser executado e
• se possuir Tabela Base, será executado tantas vezes quanto houver registros na tabela base;
• caso o grid não tenha tabela base será executado apenas uma vez
• e se o grid for baseado em um SDT, não será executado o evento Load.
É importante levar em consideração que quando trabalhamos em uma aplicação para dispositivos móveis, no código dos eventos Start, Refresh e Load não há acesso aos recursos do dispositivo, por exemplo a câmera, o GPS, etc.



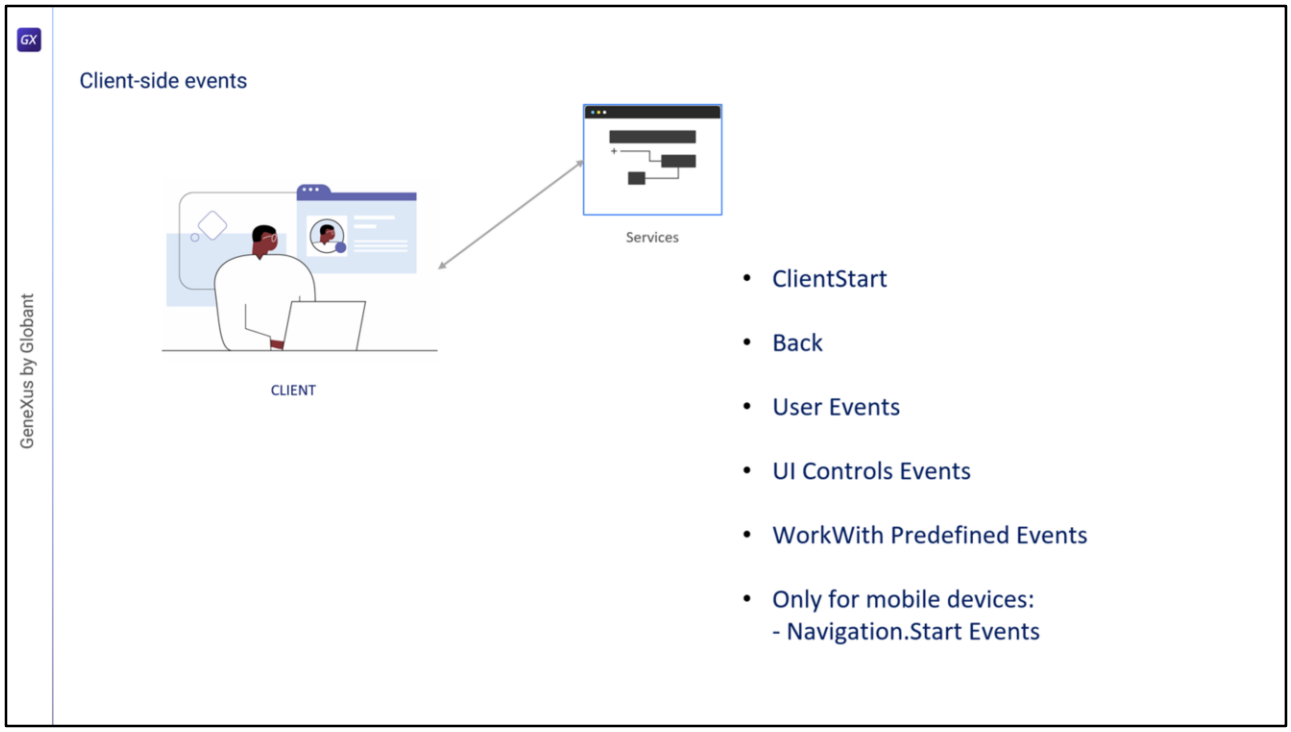
Vejamos agora os eventos do lado do Cliente. Esses eventos são a resposta da aplicação à interação do usuário.

• Existem vários tipos de eventos no cliente: os eventos de sistema como o ClientStart e o Back, os de usuário, eventos de controles em tela e outros que veremos a seguir.
• O código associado a esses eventos é executado no cliente, a menos que seja necessário acessar um recurso do servidor, por exemplo, se você quiser acessar a base de dados. Neste caso o cliente deve invocar um serviço do servidor.
• Durante a execução de um evento do lado do cliente, os eventos do lado do servidor não são executados, a menos que sejam solicitados explicitamente por meio do comando Refresh, que faz com que seja disparado o evento Refresh do servidor, seguido do evento Load (se houver pelo menos um grid, como vimos antes).
• Em aplicações nativas para dispositivos móveis, os eventos do lado do cliente têm acesso a todos os recursos de hardware e software do dispositivo, como a câmera, GPS, microfone, calendário, contatos, etc.
• Esses eventos do lado do cliente terão uma gramática particular diferente dos eventos do lado do servidor.



Vejamos brevemente cada evento do lado do cliente.

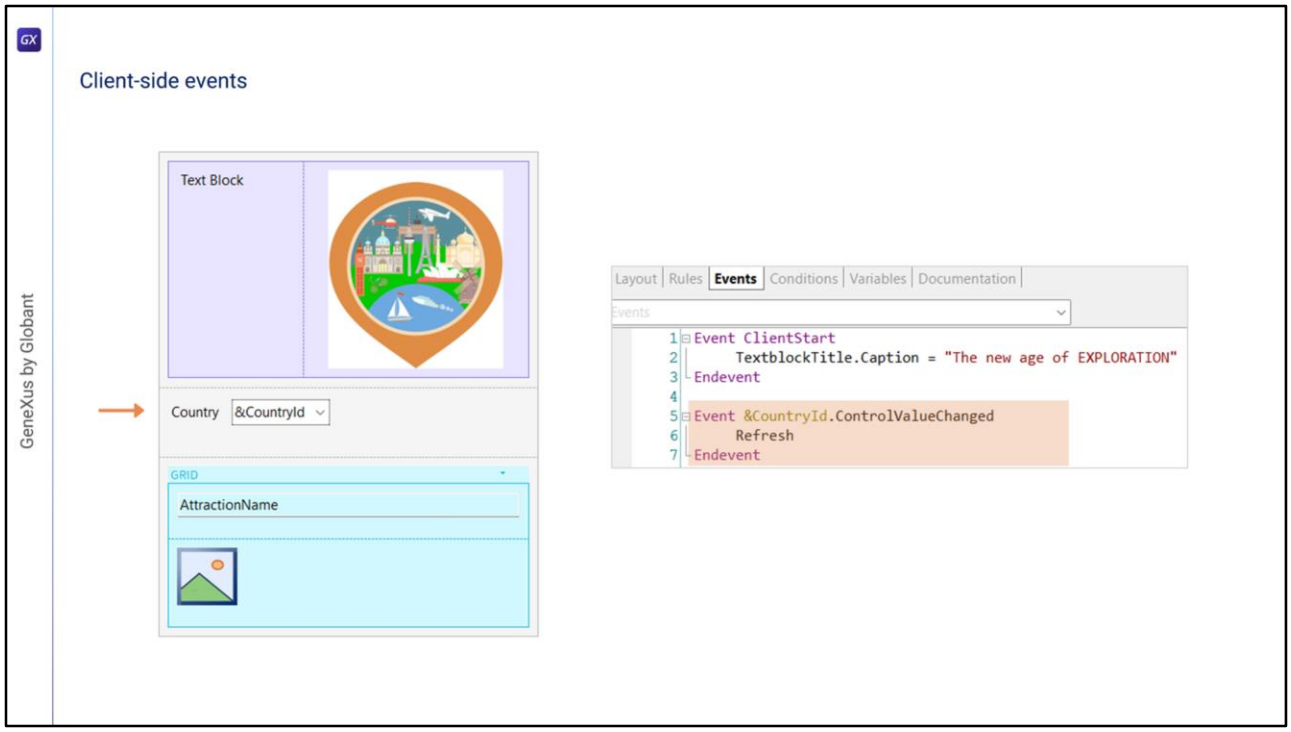
O evento ClientStart é o primeiro evento disparado, mesmo antes do evento Start que é disparado no servidor e sem necessidade de qualquer interação do usuário com a aplicação. Ele é utilizado para inicializar a tela inicial e aspectos relacionados à UI.
O evento Back é utilizado apenas em plataformas móveis e permite capturar que o usuário pressionou o botão Voltar em dispositivos Android ou que foi realizado o gesto de voltar em dispositivos iOS e programar alguma ação apropriada.
Os eventos de Usuário têm um nome dado pelo usuário e permitem que o desenvolvedor associe um determinado código que é executado quando é ativado um determinado controle em tela, ao clicar (ou tocar) sobre ele.



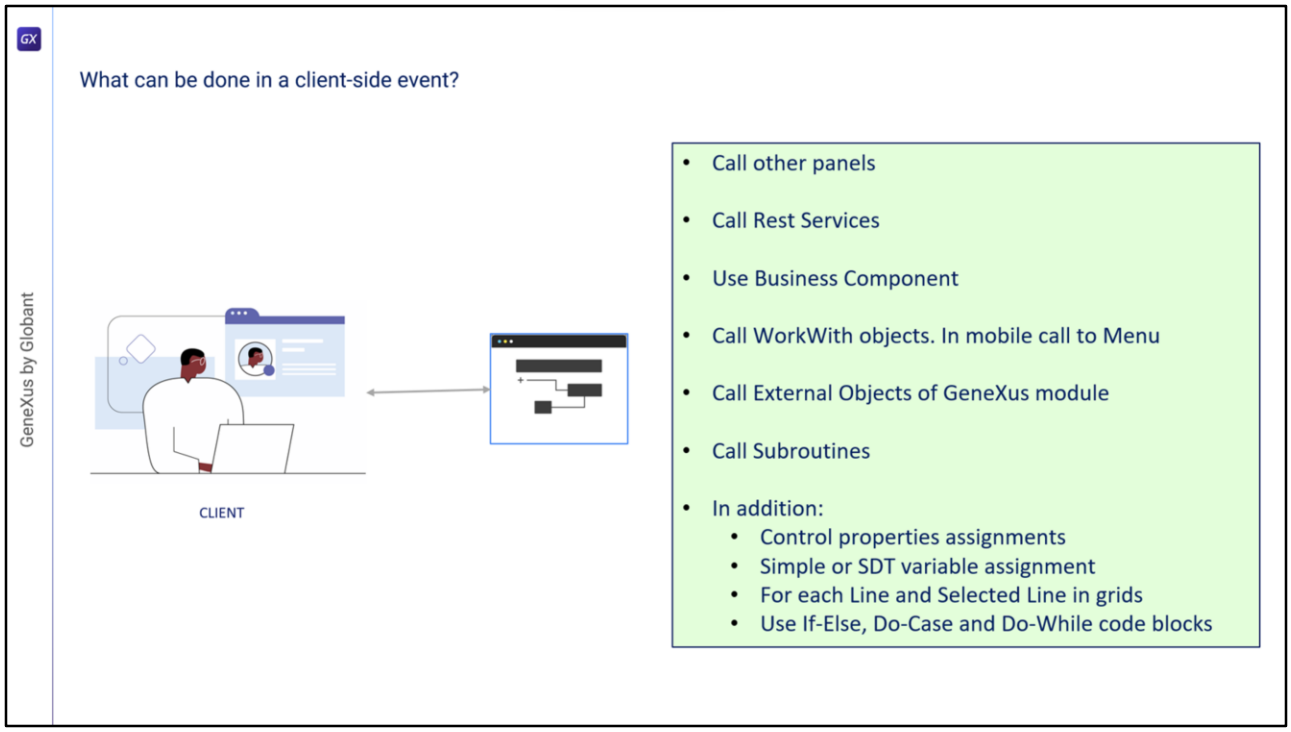
Os controles em tela possuem eventos próprios, dependendo do controle em questão, como: clicar (ou tocar), clicar duas vezes (ou tocar duas vezes), drag, swipe, ControlValueChanged, etc. Esses eventos são disparados quando ocorre alguma das ações mencionadas e o desenvolvedor pode programar uma resposta da aplicação à interação do usuário.

O padrão WorkWith possui eventos predefinidos que são disparados dependendo da ação que realizamos sobre os dados da entidade à qual aplicamos o padrão. Entre eles estão o evento Insert, Update, Delete, Save, Cancel, entre outros. Dependendo da parte do objeto WorkWith que estivermos executando (lista, detalhe, etc.), serão os eventos que estarão disponíveis.

Nos dispositivos móveis, dependendo do tipo de dispositivo e a orientação, ao iniciarmos a aplicação será disparado um evento Start que depende da navegação. Por exemplo, se estivermos usando um Tablet, por padrão a aplicação inicia em modo Split, o que significa que a tela é exibida dividida em duas seções, com uma lista à esquerda e o detalhe do elemento selecionado à direita. Nesse caso, ao iniciar a aplicação é disparado o evento Split.Start. No caso dos telefones, a tela mostrará, por exemplo, apenas a lista e para ver o detalhe será aberta outra tela independente. Este modo é denominado Flip e é o comportamento padrão nesses dispositivos, portanto ao iniciar a aplicação será disparado o evento Flip.Start. Embora dependendo do dispositivo exista uma navegação padrão, os objetos main de aplicações móveis possuem uma propriedade que nos permite alterar a forma como é iniciada a aplicação. Esses eventos Start associados ao tipo de navegação são disparados no início da aplicação mobile e imediatamente após o evento ClientStart.

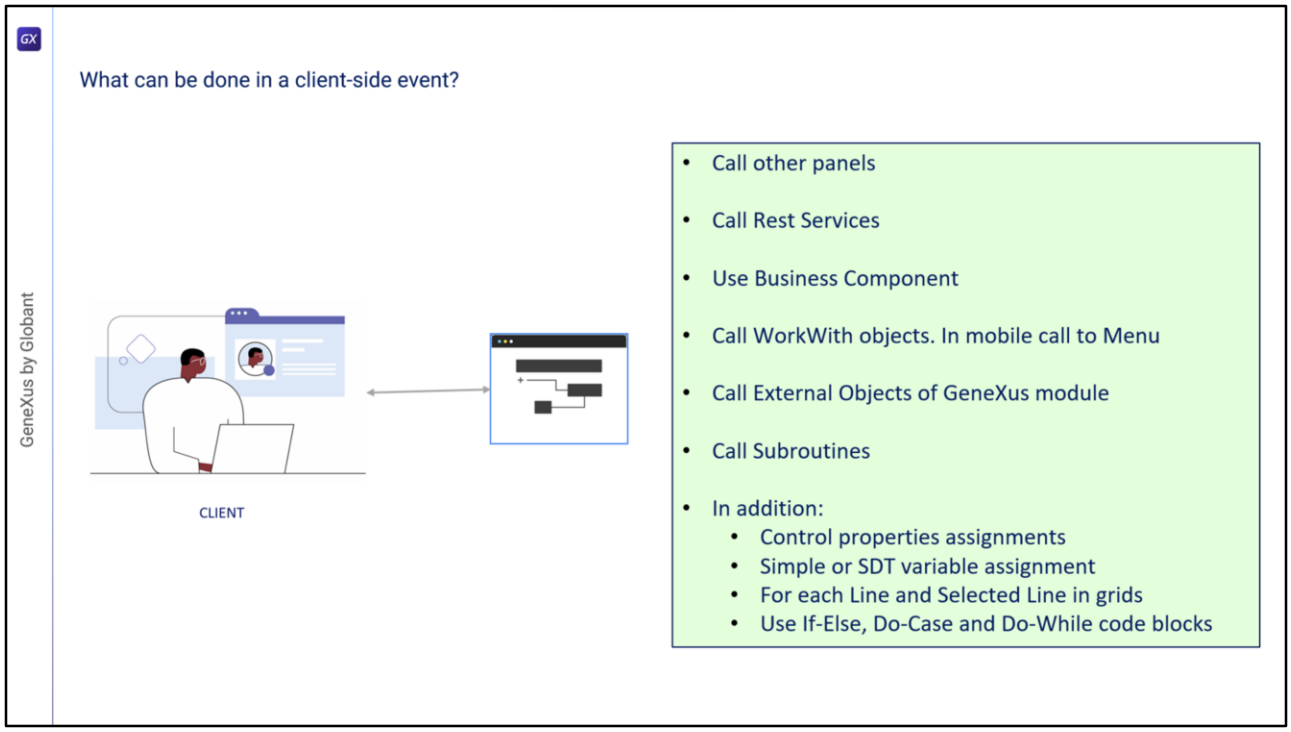


No objeto que vimos antes, programamos apenas um evento associado a um controle, não programamos nenhum evento de usuário. Para a variável de filtro em tela, programamos seu evento ControlValueChanged para disparar o método Refresh do Grid, o que fará com que sejam disparados os eventos Refresh e Load do servidor para atualizar o conteúdo do grid com o filtro inserido.

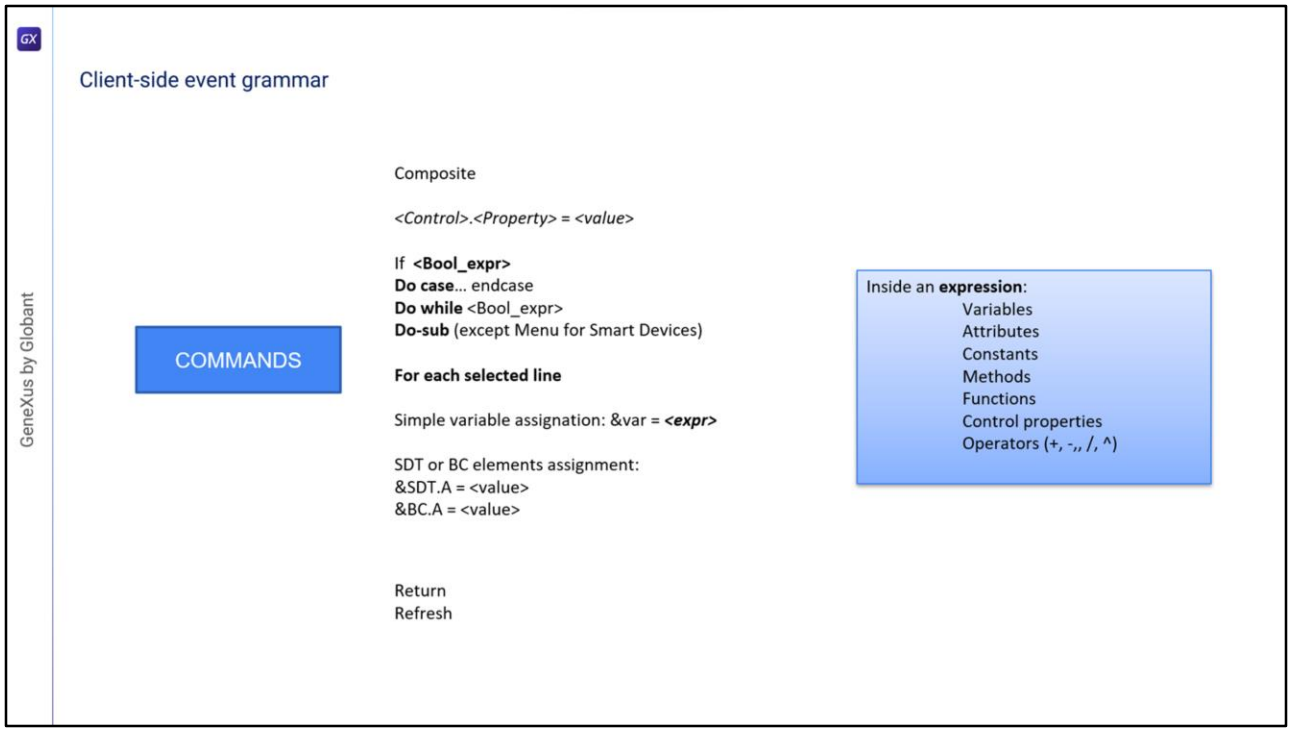


Vejamos em resumo o que podemos fazer em um evento do lado do cliente.

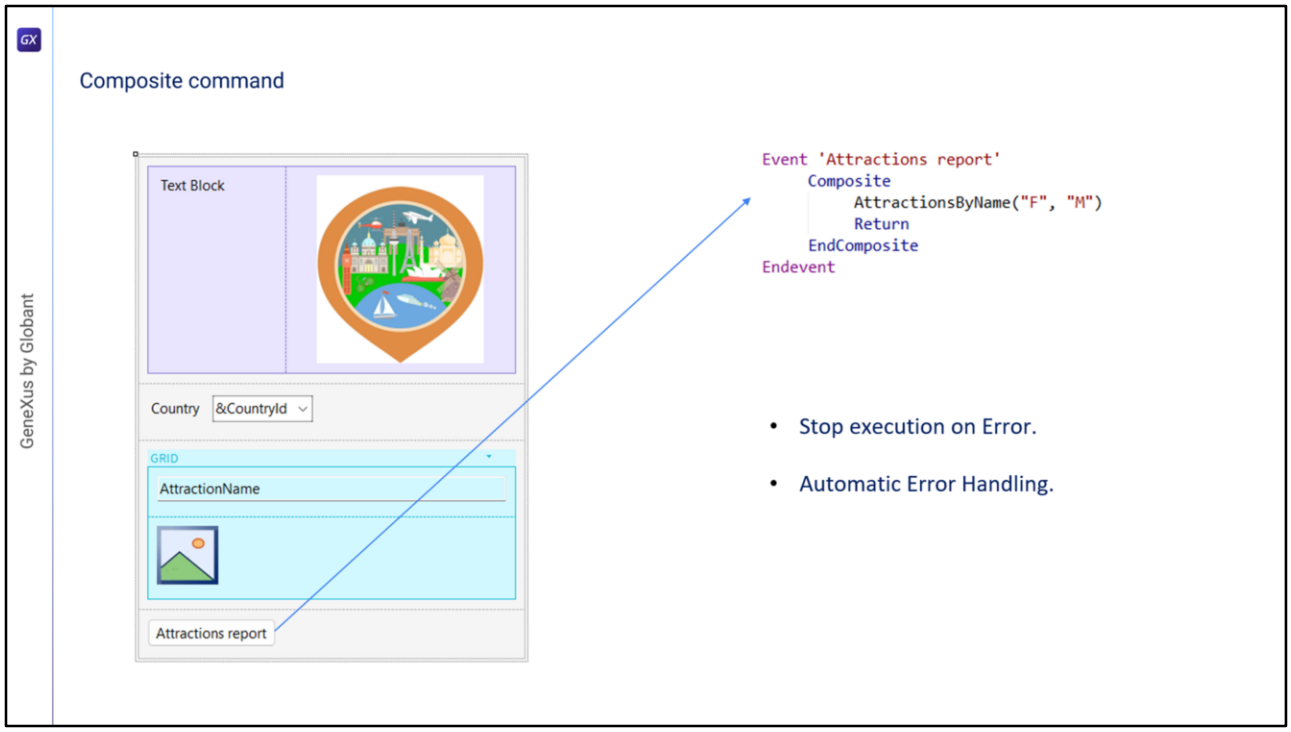
• Podemos invocar outro objeto Panel
• Também podemos invocar Serviços Rest e quando invocamos Procedimentos ou Data Providers, estes serão automaticamente expostos como serviços Rest no servidor. Se esses procedimentos ou DPs tivessem sido invocados somente a partir de um evento do lado do servidor (dentro de eventos Start, Refresh ou Load), eles não seriam expostos como serviços REST porque não seria necessário.
• Podemos usar Business Components para recuperar ou atualizar informação, neste caso também esses Business Components serão expostos como serviços Rest de forma automática.
• É possível invocar diretamente qualquer nó do padrão Work With. No caso de aplicações nativas, podemos invocar um objeto Menu.



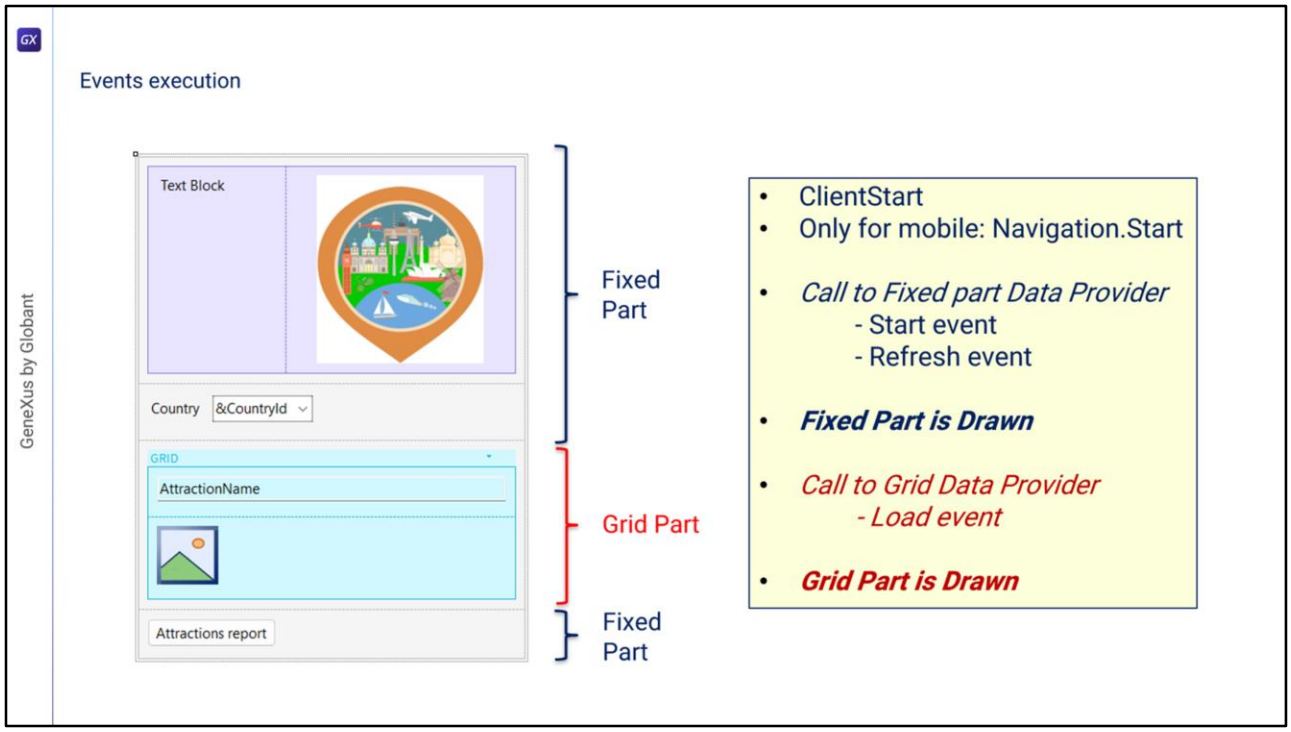
• Podemos invocar os objetos externos definidos no módulo GeneXus. Algumas dessas API's fazem sentido para uma plataforma particular, por exemplo, apenas para dispositivos móveis, outras apenas em aplicações web e outras podem ser utilizadas em ambas as plataformas
• A partir de um evento no cliente podemos chamar sub-rotinas
• E também podemos fazer:
• Atribuição de propriedades a controles
• Atribuição de variáveis simples ou de variáveis SDT
• Execução de For Each Line e For each Selected Line em grids
• Uso dos blocos IF-Else, Do-Case e Do-While
No caso de tentarmos utilizar comandos não permitidos em um evento do lado do cliente, veremos um erro na tela de saída, para aquelas linhas que não atendem às restrições da gramática, por exemplo, se tentarmos usar um comando For Each, um New ou qualquer tentativa de acessar ou modificar informação que é acessada apenas a partir do servidor.



Aqui vemos um resumo dos comandos que podemos usar no código dos eventos do lado do cliente. Estas restrições são apenas para os eventos do lado cliente, não para os eventos do lado servidor (Start, Refresh e Load) onde podemos usar todos os comandos e funções disponíveis no GeneXus. Os comandos aceitos no momento são os mostrados.

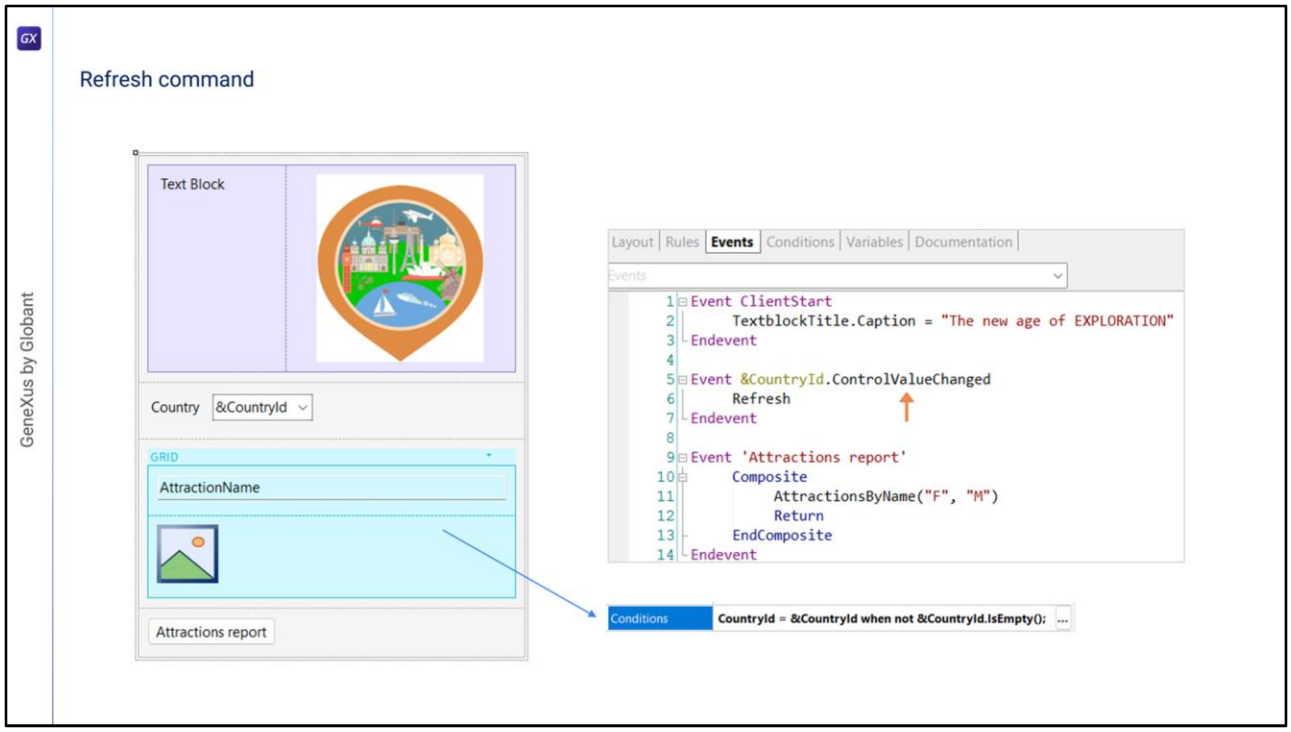


Vejamos agora o comando Composite que mencionamos anteriormente. Este comando é utilizado em eventos do lado do cliente em objetos Panel. A importância deste comando é que, quando ocorre um erro na sequência de chamadas, a execução é interrompida e são tratados os erros automaticamente, exibindo-os na tela sem ter que implementar nenhuma programação. Esta é uma grande diferença dos WebPanels, pois nestes, quando dentro de um evento um objeto chamado produz um erro, não é interrompida a execução, continua na instrução seguinte e é o desenvolvedor quem deve se encarregar de tratar os erros e programar as ações a serem tomadas. Este comando Composite é opcional: se não o utilizarmos, o funcionamento será idêntico ao dos WebPanels.

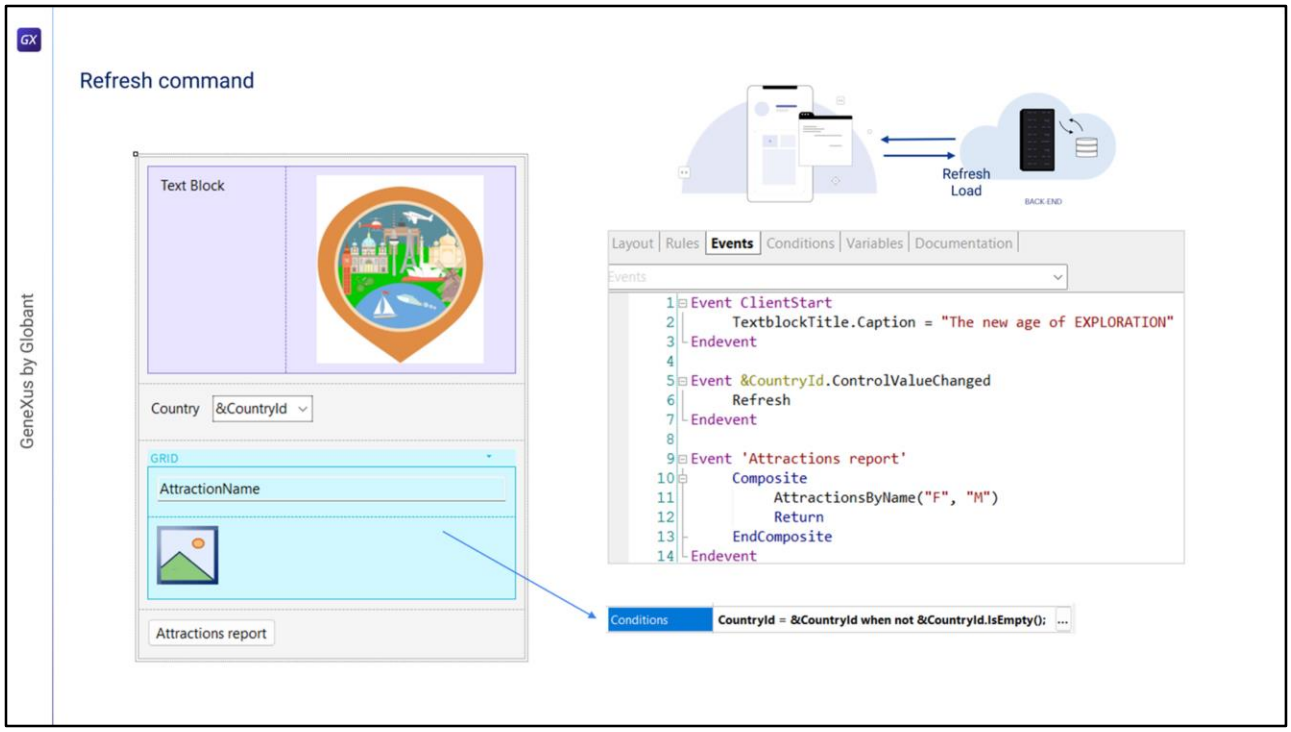


Agora vamos ver o que acontece com os eventos quando executamos um objeto Panel.

Primeiro é executado o evento ClientStart, que é executado apenas uma vez no cliente. No caso de aplicações móveis, em seguida é disparado o evento Start correspondente ao tipo de navegação estabelecido no objeto main, através da propriedade Navigation Style. Então é executado 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, a informação para carregar a parte fixa. Em seguida é desenhada a parte fixa do Panel. A seguir é executado 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, uma única vez se o grid não tiver tabela base e nenhuma vez se o grid for proveniente de uma variável SDT coleção. Ao finalizar a execução do Data Provider, ele retornará a informação gerada por todas essas execuções do evento Load em um único resultado, com o qual será carregado o grid e então terminará de desenhar o grid.



Uma das coisas que mencionamos foi que a parte fixa do Panel é carregada de forma independente da carga do grid, invocando Data Providers diferentes, que estão publicados no servidor como serviços e acessam a base de dados para recuperar a informação de cada parte. Quando alteramos o valor de um filtro, é necessário que seja atualizada a informação do grid. Para isso, é necessário que adicionemos o comando Refresh para que dispare os eventos Refresh e Load do servidor, o que fará com que seja carregado novamente o grid aplicando as conditions programadas e mostre os resultados filtrados conforme esperamos. Como devemos invocar o comando Refresh após alterar o valor da variável do filtro, usamos o evento ControlValueChanged da variável para invocar o método. Desta forma, após alterar o valor do filtro, ao sair do campo será disparado o evento correspondente que esperamos que acabe atualizando o conteúdo do grid.



Mas o que acontece quando é invocado o comando Refresh dentro de um evento do lado do cliente? Nesta arquitetura, como o objetivo é que a página seja carregada a menor quantidade de vezes possível, é priorizado o cache de dados, ou seja, trata-se sempre de recuperar a informação previamente armazenada. Isso significa que, dependendo da configuração de armazenamento em cache, é decidido se deve ir ou não para recuperar dados do servidor. Caso seja decidido ir ao servidor, se não houver alterações nos dados do servidor, nada é devolvido ao cliente. Caso contrário, são executados os eventos Refresh e Load (se houver um grid que não seja baseado em um SDT, como é o nosso caso) e são atualizadas a parte fixa e a parte variável do Panel, como esperávamos.

Nos próximos vídeos veremos outros aspectos do desenho e implementação de objetos Panel.