



Atualização GX Ev3

Smart Devices

Exercícios Práticos

Copyright © Artech Consultores S. R. L. 1988-2014.

Todos os direitos reservados. Este documento não poder ser reproduzido sem qualquer consentimento explícito da Artech Consultores S.R.L. A informação contida neste documento é para uso pessoal unicamente.

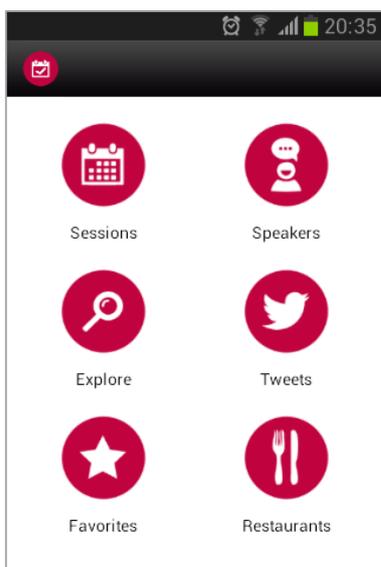
Marcas Registradas

Artech e GeneXus são marcas da Artech Consultores S.R.L. Todas as demais marcas mencionadas neste documento são propriedade de seus respectivos donos.

1. A APLICAÇÃO: PASSOS PRÉVIOS	4
2. DESENHO E COMPORTAMENTO.....	6
Pedidos e filtros.....	6
Imagens.....	6
Sobreposição de controles	8
Opcional ipad: sobreposição com tradutor (transformação)	9
Múltiplos layouts por fila de um grid.....	11
Opcional: Múltiplos layouts por fila de um grid (carga dinâmica)	12
Opcional: Estilo de navegação	13
Opções de invocações (CallOptions)	15
Opcional: abrir para direita o estilo de navegação Split.....	17
3. CONVERTENDO A APLICAÇÃO A PARCIALMENTE OFF-LINE.....	17
Funcionamento off-line	18
Send: Modificação de dados e sincronização com o servidor.....	19
Conflitos por chaves duplicadas na sincronização	19
Receive: Modificação de dados no servidor e sincronização	19
Receive: filtros nos dados enviados ao dispositivo	19
Avançado: sessões favoritas somente na base de dados do dispositivo	20
Avançado: manejo de erros na sincronização	20

1. Aplicação: passos prévios

Ao longo deste prático desenvolveremos uma aplicação para trabalhar com a agenda de um **Evento** (congresso/colóquio/encontro) a realizar-se em certos dias. A realidade deve representar as distintas conferências (**sessions**) com suas horas de início e final, salas onde se levam a cabo (**rooms**), oradores ou conferencistas (**speakers**) de distintos países (**countries**), e os tipos de conferências (**tracks**). Permitirá marcar conferências como favoritas (**favoriteSessions**) –assim como agendá-las no calendário do dispositivo ou compartilhá-las, ver os tweets (**tweets**) relativos ao HashTag que definimos –no nosso caso, para testar, será #GeneXus, e poder ingressar os próprios, assim como poder visualizar restaurantes (**restaurants**) pertos do lugar de evento, com descontos.



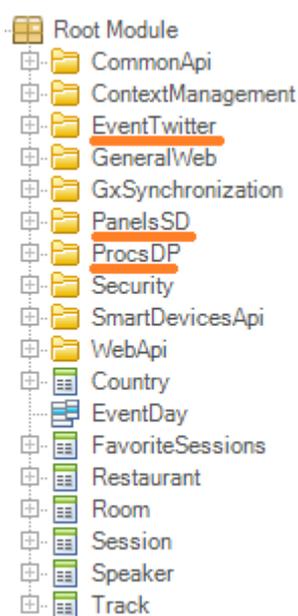
Nota: A aplicação foi desenvolvida colocando o foco no Android, por contar com um emulador apto para toda máquina. Mas também atendemos ao iOS7, por, contar com um dispositivo com qualquer dessas plataformas, pode prototipar nele sem problemas. Deverá instalar a KBN correspondente ao upgrade 1.

- 1) Criar uma nova **Knowledge Base** com ambiente de protótipo C# ou ruby (se usar a versão Trial o Environment configura automaticamente com gerador C#, interface Web e DataStore SQL Server).
- 2) Observe entre as propriedades das plataformas SD (no Folder View/Customization/SDPlatforms) para "Any Android", "Blackberry Phone", "Any iOS", "Any iOS7", "Any Windows 8":
 - a) A propriedade **Theme**: com "SimpleAndroid", "SimpleBlackBerry", "SimpleIOS", "SimpleIOS7" e "SimpleWin8", respectivamente.
 - b) A propriedade **Navigation Style**: com valor "Default"
- 3) Importe o xpz de nome **EventDay Init.xpz**. (se está usando o u2; se está com o u1, então importe **EventDay_Init_u1.xpz**) Aqui estará implementada boa parte da aplicação EventDay. Faça (levará uns minutos) enquanto repara no seguinte:

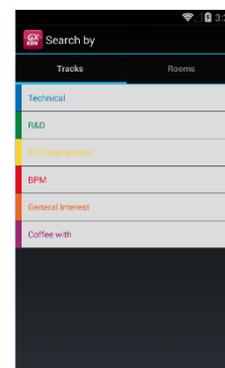
- a) Se importar os themes **EventGXAndroid**, **EventGXiOS**, **EventGXiOS7**, e se modificar a propriedade Theme das plataformas "Any Android", "Any iOS" e "Any iOS7" – e de suas herdeiras- para utilizá-los, respectivamente.
- Nestes themes se tem adicionado classes para serem utilizadas em diferentes objetos desta aplicação; se iram indicando oportunamente.
 - Quando finaliza a importação, abra o theme "EventGXAndroid", edite as propriedades da classe "TableColoredBlue" e assegure de que tenha as propriedades de Background Color como segue. Caso contrário (bug de versões anteriores ao u2), importe também **ThemeForSD.xpz**.

Background Color	 #006FB7
Highlighted Background	 #006FB7

- b) O folder view mostrará as transações com os Work With for web e for Smart Devices já criados e personalizados, o dashboard EventDay que será o ponto de arranque da aplicação móvel, e alguns folders como:

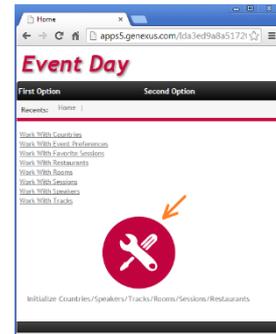


- EventTwitter, com tudo para implementar essa parte da aplicação;
 - PanelsSD, com um painel que mostra os tipos de conferência e salas de evento (como vemos a direita) e três painéis que serão utilizados para o prático.
 - ProcsDP, com procedimentos utilizados, por exemplo, para obter dado de uma session uma lista de seus speakers, de seus tracks, se está marcada como favorita para esse dispositivo, e marcá-la como favorita, ou desmarcá-la. Também terá Data providers para inicializar todas as tabelas com dados.
 - No folder GeneralWeb encontraremos o web panel **Home** que permitirá inicializar com dados de todas as tabelas.
- Se tem adicionado um folder WWSolutions que contém alguns work with solução.



- 4) Vamos prototipar em uma nuvem, pelo que a propriedade **Deploy to Cloud** deve ter o valor YES.
- a) Se está utilizando a versão Trial, não deve fazer nada, pois esta é a opção por padrão. Se está utilizando a versão full, vá ao Gerador Web Default nas **Preferences** e mude o valor da propriedade "**Deploy to cloud**" abaixo do grupo "**Execution**", passando a **True**. Poderá ingressar seu usuário de GXTechnical. Cancele se não tiver um.
- b) Se deseja utilizar o emulador Genymotion, mais veloz em máquinas Intel que o do SDK de Android, abri-lo (descarregar gratuitamente do genymotion.com).

- c) Logo faça "F5" e "Create" para criar as tabelas na DB e na nuvem. Como não temos especificado Startup object, se abrirá o Developer Menu web, também da KBN no emulador com a url da aplicação móvel. No Developer Menu execute o web panel "Home" e ali pressione o ícone para inicializar as tabelas com dados. Ingresse aos distintos WorkWith desta Home para ver como ficaram inicializados.



- d) No emulador (ou em seu dispositivo) teste a aplicação móvel, para familiarizar-se com ela.

2. Desenho e comportamento

Pedidos e filtros

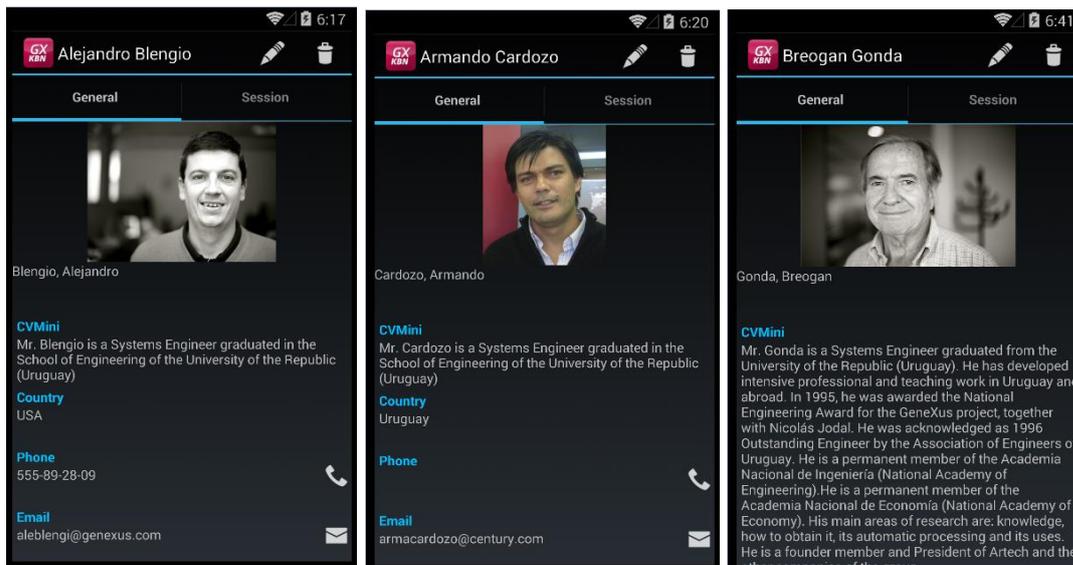
Executar a listagem de Speakers. Observar que sai ordenado por nome. Queremos modificar essa ordem, para que saia por sobrenome (surname-name). Também observar que podemos filtrar os speakers, também por nome e sobrenome, por Curriculum Vitae, Address, E-mail. Excluir estes últimos filtros.

Para fazer isso:

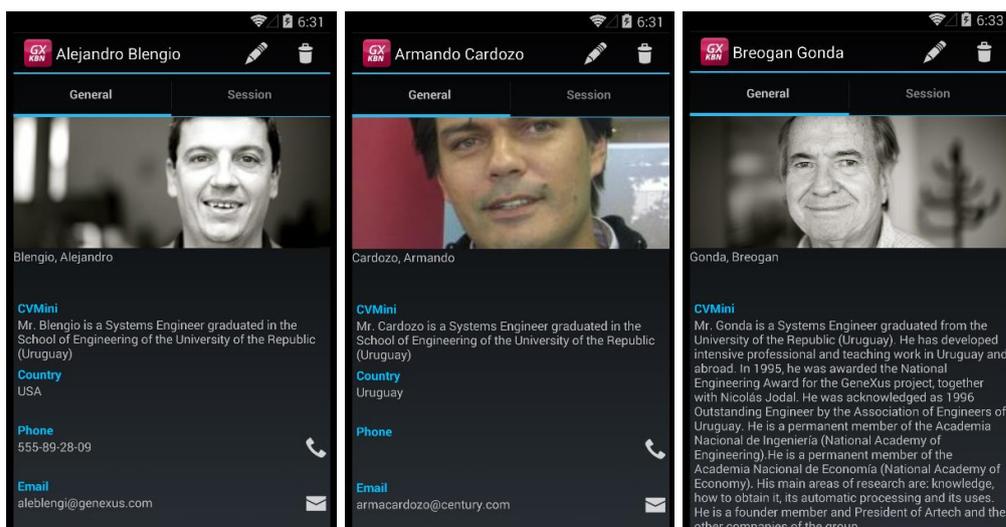
No WorkWithDevicesSpeaker, no List, editar as propriedades do Grid, e ver as do grupo "Data" Recordar que os pedidos e filtros (comuns e avançados) agora são definidos a nível do grid e já não na aba "Conditions".

Imagens

Ver em execução o detalhe de um speaker. Se observar o layout no GeneXus, verá que o atributo SpeakerImage está ocupando a primeira fila, de 150dips de alto, e a coluna ocupa o 100% da largura. Veja que a imagem está escalando em largura e longo para poder ser completamente visualizada, mantendo as proporções (as imagens fonte tem todos diferentes tamanhos). Isto é assim porque o controle SpeakerImage tem a classe "Image" (que para a propriedade "Scale Type" assumo o valor default, que é "Fit")



Desejamos modificar este desenho, para que agora a imagem ocupe todo o espaço do celular (ou seja, os 150dips de height e o 100% de width) mas sem escala (ou seja, mantendo o aspecto da imagem):



Para fazer isso:

Abrir o theme EventGXAndroid ou EventGXiOS7, e observar que definimos as classes **ImageFill**, **ImageFillKeepingAspectRatio**, **ImageFit**, **ImageTile** que unicamente se desviam do default no valor que assume a propriedade **"Scale Type"**.

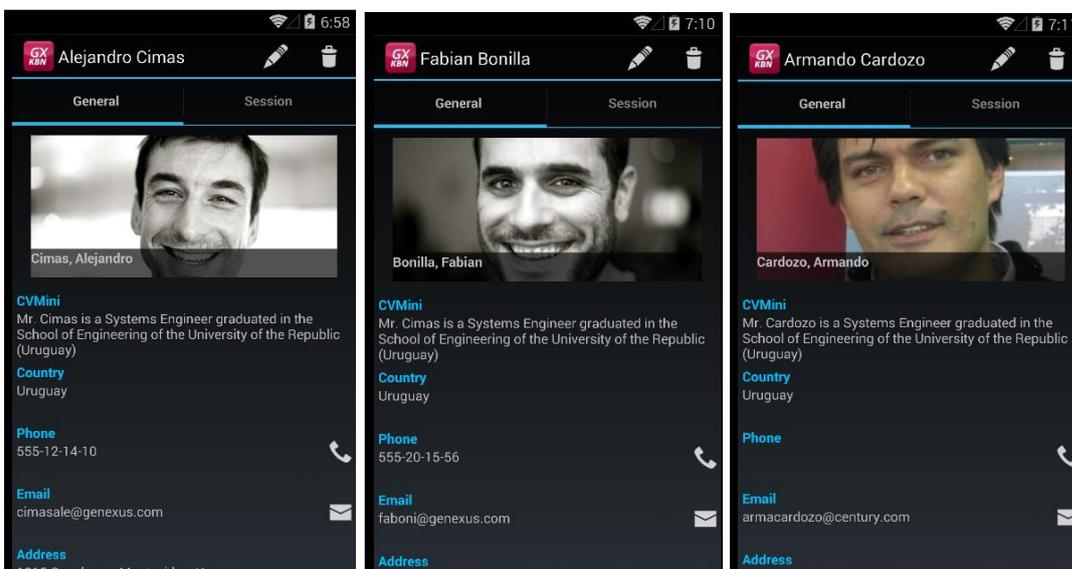
Mudar o controle do atributo SpeakerImage a classe por "ImageFillKeepingAspectRatio" e teste.

Faça também para a imagem do List:



Sobreposição de controles

Agora queremos que no detalhe do speaker, o nome completo apareça em cima da imagem, e não abaixo:



Solução:

Mover uma fila para baixo o controle SpeakerFullName e o SpeakerImage, de maneira que fique livre a primeira fila. Inserir ali um controle Canvas. Logo, arrastar SpeakerImage dentro do Canvas e fazer o mesmo com SpeakerFullName. Excluir a segunda filada MainTable, que já não será necessária. Observe que todos os controles dentro do Canvas entre suas propriedades estão adicionados as do grupo **Absolute position**, que são as que permitem indicar a posição do controle a respeito do ponto (0,0) do canvas (que é o superior, esquerdo).

Atribuir os seguintes valores para cada controle, pensando que significam:

Absolute position	
Left	20dip
Width	100%
Right	20dip
Top	10dip
Height	150dip
Bottom	100%
ZOrder	0

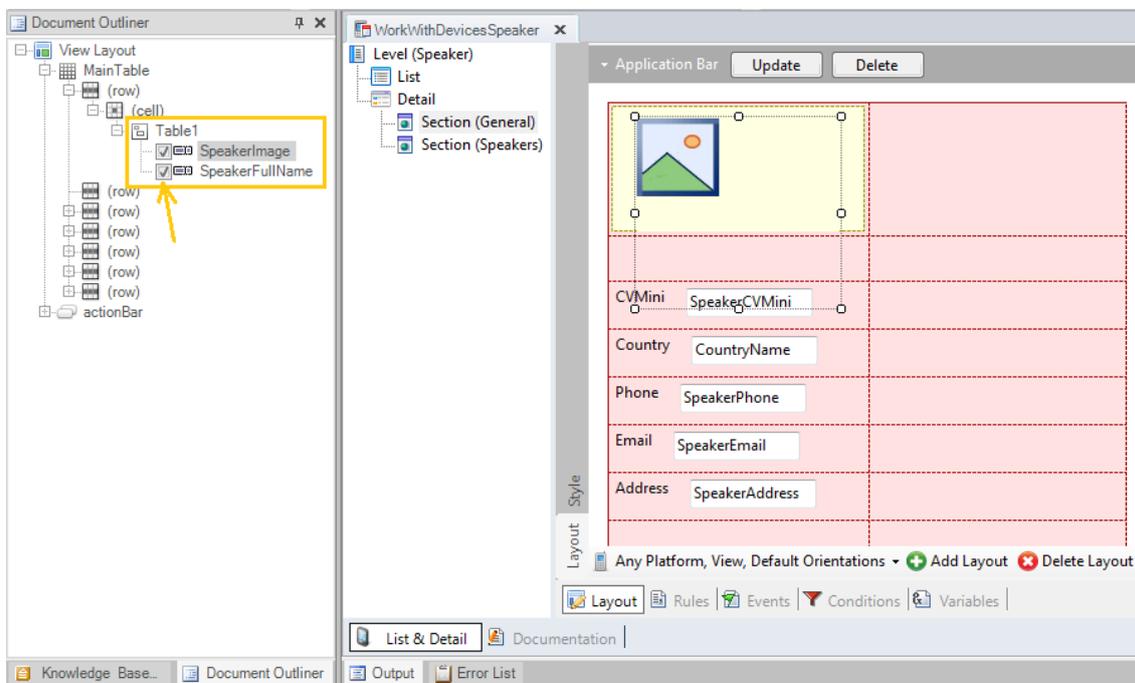
SpeakerImage

Absolute position	
Left	20dip
Width	100%
Right	20dip
Top	130dip
Height	30dip
Bottom	100%
ZOrder	1

SpeakerFullName

Para que o controle SpeakerFullName saia com um fundo, mude sua classe por "AttributeFontColorWhiteWithBackground" que vem nos themes.

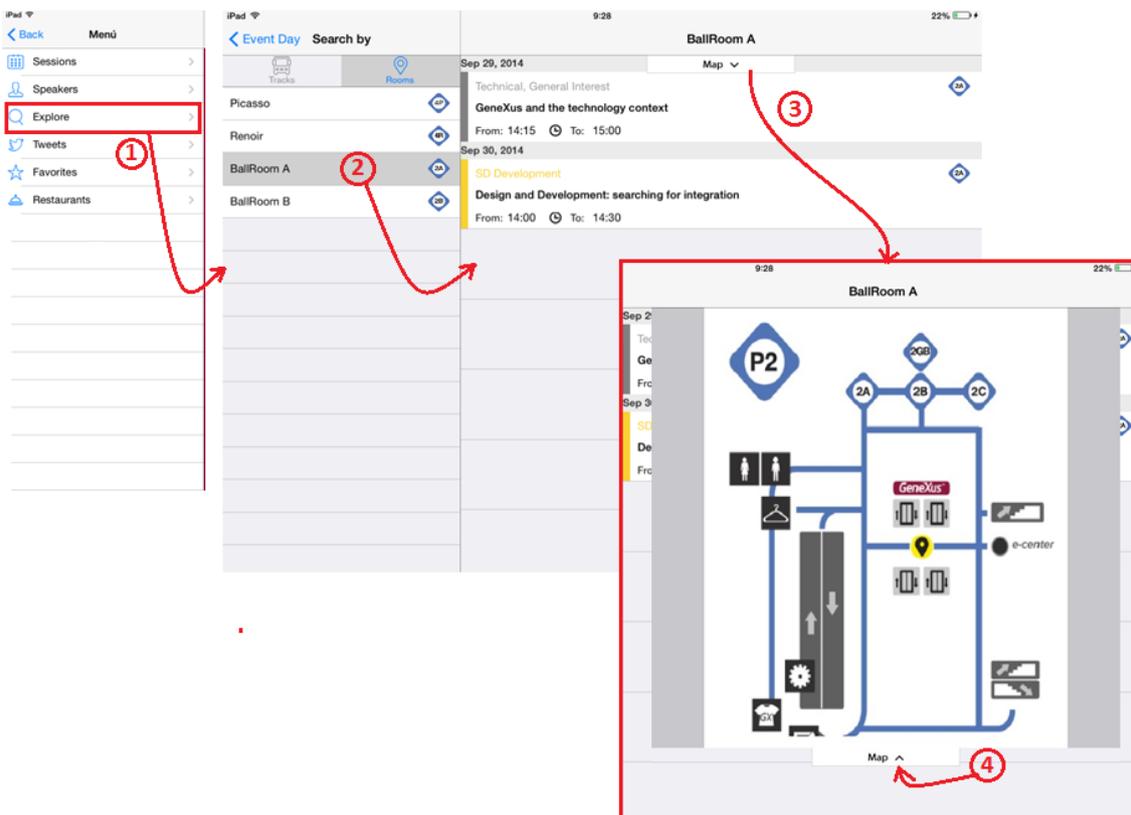
Nota: lembre-se que para poder trabalhar com os controles dentro de um canvas, será necessário utilizar a janela do Document Outliner. Mesmo assim poderá ocultar com os check boxes alguns dos controles para poder deixar visíveis os que estão nas capas mais profundas.



Pode observar o painel solução "Solution WorkWithDevicesSpeaker" do folder WWSolution para comparar.

Opcional ipad: sobreposição com translação (transformation)

Se conta com um iPad, execute a aplicação desde a KBN, como indica na seguinte imagem:



Vera que ao fazer tap na imagem “Map” (ponto 3), irá dobrar para baixo uma imagem com um plano do piso onde se encontra a sala, indicando sua localização, por cima da lista de conferências –que seguem ficando debaixo (se tem utilizado um controle canvas)-. Se voltar a fazer tap (ponto 4), irá dobrar para cima, deixando de ver o plano. Para isso, os controle se movem para baixo ou para cima, fazendo uma das **transformações**.

Para observar a implementação no GeneXus:

Vá ao WorkWithDevicesRoom, Detail/Section(Session), e escolha ver o layout para iPad. Observe as classes TableMapCollapsed e TableMapExpanded do theme EventGXiOS7, que utilizam como Transformations as classes MapCollapse e MapExpand, abaixo o folder Transformations do theme.

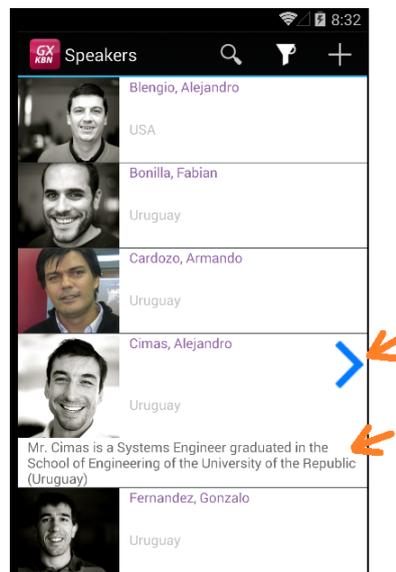
Múltiplos layouts por fila de um grid

Queremos modificar o List do Speakers, para que a cor de fundo seja branco e violeta e a cor do Font do nome, e para que quando o usuário faça tap sobre um dos speakers, em lugar de chamar o Detail, mostre uma porção do Curriculum Vitae (CV). Para poder invocar o Detail, adicionaremos uma imagem (arrow):

Implementá-lo.

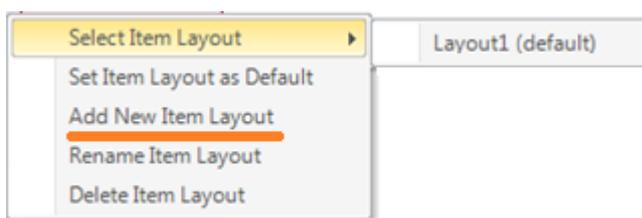
Nota: encontrar várias imagens "arrow" na kb; existe nas classes "TableColoredWhite" e "AttributeFontColorViolet"; também da classe "AttributeDarkMargin10" que poderá utilizar para que o CV fique a 10 dips da borda esquerda.

Se não deseja perder tempo com o layout, abra o objeto solução Solution WorkWithDevicesSpeaker, edite o layout **Selected** do grid, e copie em seu layout.



Solução:

Deverá criar outro layout para o grid (clique sobre o canto superior direito do grid/Add new item layout)



Criamos um que chamamos "Selected", personalizando o layout da seguinte forma:

Table: Table1	
Control Name	Table1
Columns Style	30%;10dip;60%;10% ...
Rows Style	pd;pd;pd
Width	100%
Height	pd
Auto Grow	True
Layout Name	Selected

Utilizamos variável **&SpeakerCVMini** para poder cortar o CV em 145 caracteres, no evento Load do grid. Deixamos que o controle expanda para ocupar as 4 colunas e o especificamos o AutoGrow em True. Mudamos a classe por AttributeDarkMargin10 para que tenha um espaço esquerdo de 10 dips e não fique pegando a borda.

Logo, devemos desabilitar o Default Action do grid, para que quando se faça tap sobre um elemento, não se faça nada, mais que carregue o layout "Selected":

Grid: Grid1	
Control Name	Grid1
Collection	
Default Action	<none>
Show Selector	<new>
Selection Type	<default>
Selection Flag	<none>
Selection Flag Field Specifier	'Insert'
Default Selected Item Layout	Selected
+ Control Info	

Por último, programamos o evento Load, e o evento Tap associado a imagem da data, a que chamamos "arrow":

```

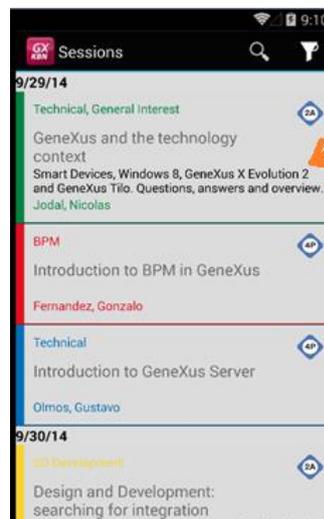
Event Load
  Table1.Class = "TableColoredWhite"
  SpeakerFullName.Class = "AttributeFontColorViolet"
  If SpeakerCVMini.Length() > 145
    &SpeakerCVMini = substr(SpeakerCVMini,1,145) + "... "
  Else
    &SpeakerCVMini = SpeakerCVMini
  EndIf
EndEvent

Event arrow.Tap
  WorkWithDevicesSpeaker.Speaker.Detail(SpeakerId)
EndEvent
  
```

Opcional: Múltiplos layouts por fila de um grid (carga dinâmica)

De todas as conferências algumas são Keynotes. Isso está especificado no atributo SessionIsKeynote. Desejamos que quando uma session é keynote, na listagem de conferências seja adicionado sua **descrição** (como se vê na tela que segue, para a primeira conferência "GeneXus and the technology contexto" que é a única keynote das que se veem nesta tela). Como implementar?

Nota: Pode utilizar o "Solution WorkWithDevicesSession" para copiar o layout keynote se não quer perder tempo no desenho.



Solução:

Adicionar um layout ao grid, ao que chamamos keynote, com o seguinte desenho:

The screenshot shows a grid layout on the left and a configuration table on the right. The grid contains several controls: &SessionTracks, SessionName, &SessionDescripti, and SessionSpeakers. A red arrow points to the &SessionDescripti control with the text "ColSpan 2" and "Autogrow True". The configuration table on the right is titled "Table: Session" and has the following properties:

Table: Session	
Control Name	Session
Columns Style	10dip;85%;15%
Rows Style	10dip;30dip;pd;35dip;30dip ...
Width	100%
Height	pd
Auto Grow	True
Appearance	
Class	Table
Visible	True
Invisible Mode	Keep Space
Enabled	True
Background	(none)
Scroll Behavior	
Cell information	
Row Span	1
Col Span	1
Horizontal Alignment	Default
Vertical Alignment	Default

A variável &SessionDescription mudamos a classe a AttributeDarkFont. Logo, no evento Load adicionamos o código que já havia, o seguinte:

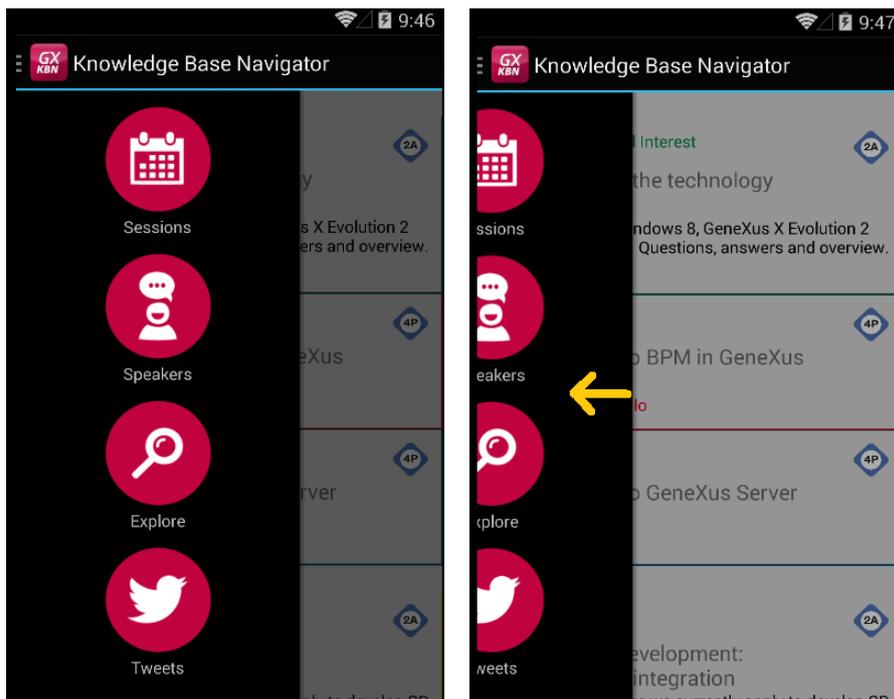
```

If SessionIsKeynote
    Grid1.ItemLayout = !"Keynote"
EndIf

If SessionDescription.Length() > 145
    &SessionDescription = substr(SessionDescription,1,145) + "..."
Else
    &SessionDescription = SessionDescription
EndIf
    
```

Opicional: Estilo de navegação

Modificar o estilo de navegação da aplicação para Android, para que seja "Slide". Observar que automaticamente se abre o List de Sessions, primeiro item do menu e que sempre pode acessar o menu deslizando desde a esquerda:

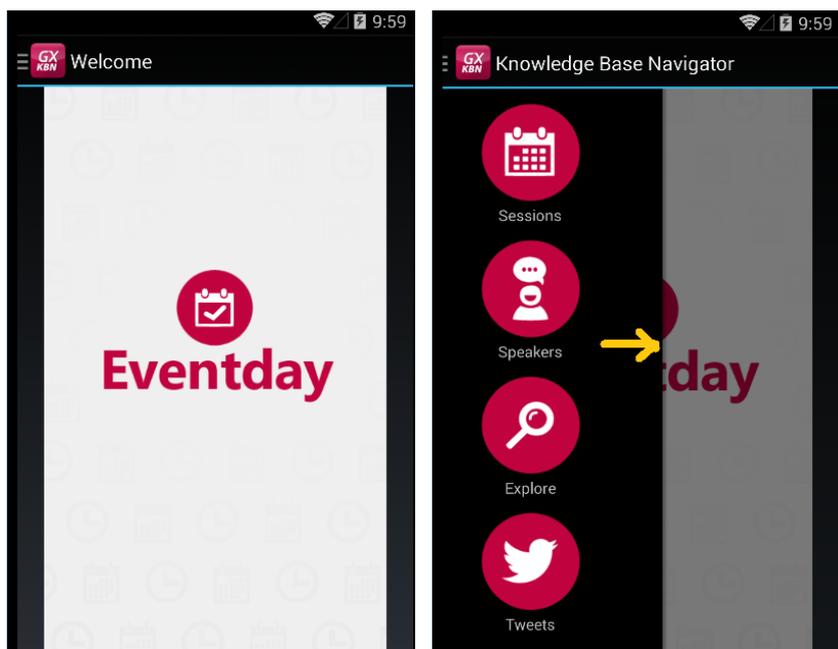


Solução: no folder view/SD Platforms, editar as propriedades de "Any Android". Ali mudar a propriedade "Navigation Style"

Se agora deseja que para o caso de usar-se estilo de navegação Slide, ou quando o dashboard utiliza como tabs (opção por padrão para iPhone, como mostra na imagem da direita), no lugar abrir-se automaticamente a lista de Sessions, queremos que abra uma tela de boas-vindas (observe o painel Welcome no folder Panels SD), como implementar?



Queremos Obter o seguinte:



Solução: programe a invocação ao painel Welcome nos eventos "Slide.Start" e "Tabs.Start" do objeto main (o dashboardEventDay).

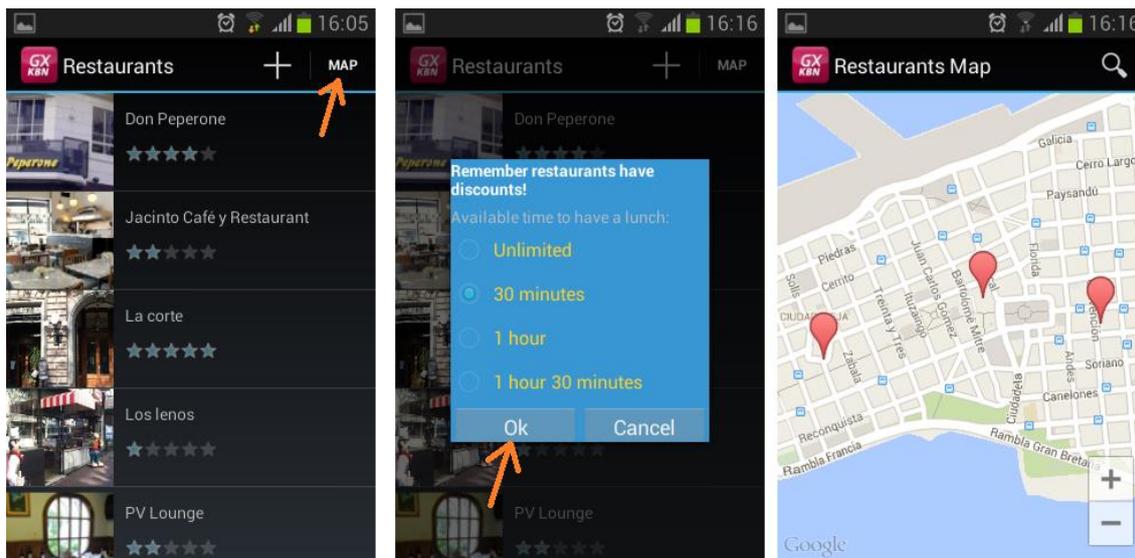
Se conta com um iPad, observe que ali não abre o painel de boas-vindas. Teste utilizando para iOS o estilo "Cascade", e veja que tampouco se abre o Welcome.

Agora voltar a deixar em Default os estilos de navegação (para Android e para iOS), e observe que abre neste caso o painel de boas-vindas.

Opções de invocação (CallOptions)

Quando o usuário escolher ver a lista de restaurantes - fazer ao explica - queremos dar-lhe a opção, através de um botão "Map", de que os possa ver em um mapa. Mas antes de mostrá-los, queremos lembrar-lhe que os restaurantes têm diferentes descontos para os participantes do Evento. Também, queremos dar-lhe a possibilidade de mostrar só aqueles restaurantes que se compromete com o cliente durante almoço em nenhum tempo disponível.

Verá algo parecido ao seguinte:



Observe que a primeira tela, é o nó List do Work With de Restaurants; a segunda é um painel de nome "RestaurantFilter" (que estará mostrando não como se vê aqui, mas como uma tela independente) e a terceira é um painel de nome "RestaurantsMap", cujo layout é muito similar ao do List de Restaurantes. A diferença é que este grid tem na propriedade control type o valor "SD Maps" e que filtra em suas conditions pelo tempo que dispõe o cliente para almoçar (recebido por parâmetro).

Desejamos que a tela "RestaurantFilter" não abra como uma tela independente, mas como um popup. Como modificar este comportamento?

Notas:

1. Lembre que existem os domínios predefinidos **CallType** e **CallTargetSize**.
2. Para usar Google Maps em uma aplicação Android, deve contar com uma API Key de Google. Ao contrário, os mapas não são mostrados. Para saber como obtê-los, vá a nossa wiki:

<http://wiki.genexus.com/commwiki/servlet/hwikibypageid?19055>.

Solução:

```
Event 'Map'
  Composite
    RestaurantFilter.CallOptions.Type = CallType.Popup
    RestaurantFilter.CallOptions.TargetSize = CallTargetSize.Small
    RestaurantFilter.Call(&timing, &ok)
    if &ok
      RestaurantsMap(&timing)
    endif
  endcomposite
Endevent
```

Se agora deseja que quando abra a tela com o mapa, ao fazer back não voltar a List de Restaurantes, mas se voltar o que chamou, como implementá-lo?

Solução: adicione antes da invocação o painel RestaurantsMap, o seguinte:

```
RestaurantsMap.CallOptions.Type = CallType.Replace
```

Lembre que dessa maneira estará sobre escrevendo o stack de invocações.

Opcional: abrir para direita para estilo de navegação Split

Se está prototipando em um iPad, observe que se desde o dashboard escolha ver os tweets, no lugar de abrir-se o panel na própria janela do dashboard, se abre para a direita. Por que?

Solução: observe que no dashboard, o evento associado aos tweets é:

```
Event 'EventTweets'
  composite
    EventTweets.CallOptions.Target = "right"
    EventTweets()
  endcomposite
EndEvent
```

3. Convertendo a aplicação a parcialmente off-line

Queremos que a aplicação possa funcionar sem conexão com a internet, salvo a tela que mostra os tweets, que queremos que funcione online, devido a velocidade com que muda essa informação.

Não temos aplicado o GAM, para evitar demoras, porque não temos informação de usuários. Se a tivéssemos, indicariamos que somente aqueles usuários com rol de administrador teriam permitidas as telas de Detail no modo edit –com as operações de CRUD (Create/Update/Delete)-de Speakers e Restaurants. Para poder testar a sincronização com essas operações, temos deixado habilitadas sem restrição (o que em verdade não tem muito sentido para um usuário que não seja administrador do backend).

Salvo os tweets que serão consultados só online, a informação mudará muito pouco. Não obstante, para realizar os testes, lembramos que a sincronização, a hora de receber as mudanças no dispositivo, seja a cada 60 segundos e que a granularidade seja por fila, não por tabela.

Implemente estes requerimentos e realize os testes que indicados mais abaixo.

Solução:

Observe o valor da propriedade "Connectivity Support" para:

- 1) O Dashboard (que é main): "Online"
- 2) Para as transações que tem Business Components associados, para os WorkWithDevices, para o painel EventTweets e para os data providers e procedimentos do folder ProcsDP: "Inherit"

Para que a aplicação seja Off-line, no Dashboard: "Connectivity Support = Off-line". Todos os objetos que se encontram na árvore de invocações deste main, e que tenham o valor "Inherit" para a propriedade, passarão serem implementados "Off-line" (serão codificados em java para Android, e em Objective C para iOS).

Queremos que o painel EventTweets mostre os tweets obtidos pelo servidor, ou seja, deve executar-se online. Para isso, modificamos a propriedade "Connectivity Support = Online".

Faça um **Rebuild all** para que sejam criadas o objeto OfflineDatabase (abaixo do Dashboard), os programas de criação da base de dados local no dispositivo (tanto em Android como em iOS), junto com os programas de sincronização e observe:

- As tabelas que foram criadas locais no dispositivo. Por que aparecem todas as tabelas, salvo a de EventPreferences? (está não se encontra na árvore que parte do dashboard)
- No Navigation View, veja por exemplo que:

- EventExplore aparece 2 vezes: para environment Android e para ObjectiveC.
- EventTweets aparece 1 vez, com environment Default (C# ou ruby, dependendo da plataforma que você escolheu para web)
- Speaker_BC aparece 3 vezes: com environment Default (C# ou ruby) para utilizar na aplicação web, Android e ObjectiveC, para quando utilizar na aplicação mobile.
- Observe também as listas de seleção.

Modifique as propriedades do grupo Receive.

Offline Database: EventDayOfflineDatabase	
Name	EventDayOfflineDatabase
Description	Event Day Offline Database
Event Replicator	GxOfflineEventReplicator
Qualified Name	EventDayOfflineDatabase
Object Visibility	Public
Receive	
Data Receive Criteria	After Elapsed Time
Minimum Time Between	60
Data Receive Granule	By Row
Send	
Send Changes	When connected

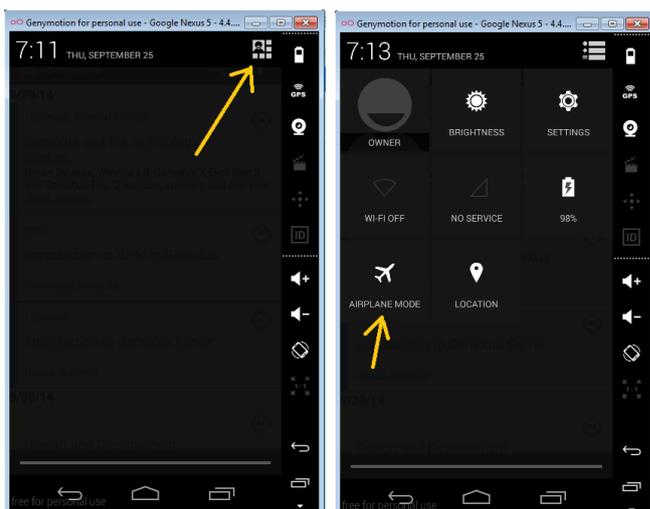
Lembre que deve compilar o main para que se possa gerar a aplicação off-line (código nativo). Para isso, faça Run sobre o dashboard.

Funcionamento off-line

Observe que quando são executados a aplicação no dispositivo, executam os programas de criação da DB, e realiza a primeira sincronização (receive) onde são carregadas essas tabelas com os dados.

Estando conectado à internet, veja as Sessions, e logo veja os Tweets. Desconecte-se (utilizando o modo Airplane) e tente ver o Tweets. Com as Sessions não terá problema.

Nota: para usar o modo Airplane usando genymotion:



Send: Modificação de dados e sincronização com o servidor

Tente modificar o dado de um speaker desde o dispositivo, estando conectado. Observe desde o web panel Home se esse dado foi automaticamente modificado na base de dados central (observe a propriedade Send Changes do objeto OfflineDatabase).

Tente agora desconectar da internet o dispositivo (ponha em modo avião), e fazer o mesmo (inserindo, modificando ou excluindo um speaker). Logo conecte-se e veja se ocorre imediatamente a sincronização.

Conflitos por chaves duplicadas na sincronização

Desconecte o dispositivo da internet. Agora ingresse um restaurante através do dispositivo. Observe o ID que se atribuiu. Simultaneamente, ingresse um restaurante através da web panel Home e observe o ID (lembre-se que é autonumerado). Conecte o dispositivo a internet, espere 60 segundos e observe o que aconteceu com os dois restaurantes ingressados. Que identificador ficou?

Receive: Modificação de dados no servidor e sincronização

Atualize algum dado desde o web panel Home, tendo o dispositivo desconectado da Internet. Logo conecte-o e espere 60 segundos. Observe o que acontece.

Receive: filtros nos dados enviados ao dispositivo

Se agora deseja que no caso do dispositivo estiver em inglês, enviem a base de dados local somente os dados correspondentes a USA, e caso contrário, enviem somente os do Uruguai?

Experimente

Solução:

Terá que programar no objeto OfflineDatabase o evento Start

```
Event Start
    &language = ClientInformation.Language
    If &language = "en-US,en"
        &Country = "USA"
    else
        &Country = "Uruguay"
    endif
Endevent
```

E nas conditions do objeto especificar:

```
CountryName = &country;
```

Avançado: sessões favoritas somente na base de dados do dispositivo

Ao não ter registro de usuário, as sessões favoritas somente se registram por dispositivo. Isso faz desnecessário que essa informação esteja na base de dados centralizada. Alcançará com que está na base de dados local do dispositivo. Como consegui-lo?

Solução:

É o procedimento SetFavoriteSession o que ingressa um registro para marcar a sessão como favorita, ou para excluí-lo. Tem "Connectivity Support = Inherit", pelo que, como é invocado desde um evento do cliente (o evento 'SessionFavorite' do WorkWithDevicesSession/Detail/Section (geral)) de um objeto que executa off-line (pois herdado este modo do dashboard), então o procedimento executará off-line.

Lembre que se em um objeto off-line ingressa informação a base de dados local através de new/delete/foreach –ou seja, não através de um business component-, essas mudanças não sejam levados no Send a base de dados centralizada.

Avançado: manejo de erros na sincronização

Se sobrar tempo, podemos observar os procedimentos GxOnPendingEventFailed (chamado quando falha um evento na sincronização; devolve variável booleana que indica se continuar ou não a sincronização) e GXAfterEventReplicator (chamado logo de terminar de processar todos os eventos), para manejar os erros que podem ter surgido na sincronização.