PROGRAMAR

REVISTA PORTUGUESA DE PROGRAMAÇÃO • WWW.PORTUGAL-A-PROGRAMAR.PT

EDIÇÃO #45 - MAIO 2014

ISSN 1647-0710

EXTENDENDO UMA APLICAÇÃO
CRIADA NO APP STUDIO DA MICROSOFT

A PROGRAMAR

PASCAL ARRAY DE ARGUMENTOS

JSF PARTE III

ANDROID APP COM MAPAS

C# CRUD (CREATE, READ, UPDATE & DELETE)

COLUNAS

PROGRAMADOR GESTOR COREDUMP 12

C#

MÉTODOS DE EXTENSÃO VISUAL (NOT) BASIC

COMUNIDADES

NETPONTO CRIANDO APLICAÇÕES WINDOWS PHONE 8.1 E WINDOWS 8.1 USANDO O APP STUDIO DA MICROSOFT

NO CODE

O WINDOWS PHONE 8.1 E A ATUALIZAÇÃO DO WINDOWS 8.1 WINDOWS

EDITORIAL

EQUIPA PROGRAMAR

try: keepUpWork: Except Error: print ”again” break

Tentamos… Ousamos falhar, tentamos de novo… E no final do dia, tentamos outra vez!

Esta semana vi um cartoon engraçado sobre o desenvolvimento de aplicações open-source! Achei particularmente caricato, pelas semelhanças do cartoon com esta nossa e vossa revista. Vou tentar “codificar” o cartoon em palavras, apesar do sucesso ser pouco provável! No cartoon aparecia como título: O que as pessoas pensam sobre como as aplicações open-source são feitas” e tinha vários grupos de pessoas, a trabalharem simultaneamente em ideias a construir algo, a testar a construir, a idealizar… E abaixo tinha uma segunda quadricula intitulada “O que realmente acontece”, onde se via um sujeito solitário, com aspecto de cansado com o balão de fala onde se lia “let me just close one more ticket before sleep”, e via-se no fundo um relógio onde as horas marcavam perto de 2h da manhã. Era mais ou menos isto. Recomendo verem a imagem original na web, pois é bem mais ilustrativa e engraçada.

De facto tantos projectos são começados por um monte de gente e mantidos por um “idealista” que fica acordado todas as noites até altas horas, apenas para manter o projecto vivo. De certa forma com este projecto que é a revista, é exactamente isso que acontece! As pessoas pensam algo, mas a realidade é “ligeiramente” diferente! Caricata coincidência!

Cada noite passada a programar faz-me pensar na nostalgia de outros tempos em que programar era mais desafio que trabalho, era mais prazer que ocupação, era mais paixão que dever… Era como escrever num newsgroup a resposta a algo, ou partilhar aquela “coisa que a gente tinha acabado de fazer”, apesar das horas já irem bem avançadas e o dia já se fizesse notar, por entre as janelas fechadas.

Toda esta nostalgia me fez lembrar o gosto que pessoalmente tenho em editar esta revista. As horas que ela consome a todos nós que a fazemos e o gosto que temos em fazê-lo. As vezes que pensamos que “não vamos conseguir”, mas continuamos em frente e almejamos mais! Porque na realidade somos programadores! Isso de alguma forma está “no nosso ADN”.

Por todas estas razões, pela nostalgia e o gosto de trabalhar nesta edição prometemos continuar o trabalho, aprendendo, superando, tentando de novo!

"Porque nós, programadores, somos “gente de ideias”, e não desistimos!

Por isso e por muito mais… Programar… ontem, hoje e amanhã!

Até à próxima edição.

António Santos

ÍNDICE

TEMA DE CAPA

A PROGRAMAR

COLUNAS

ANÁLISES

COMUNIDADES

NO CODE

EVENTOS

27 de Maio a 26 de Junho - OpenDays Engenharia Informática IPVC
13 a 14 de Outubro RubyConf

Para mais informações/eventos: http://bit.ly/PAP_Eventos. Divulga os teus eventos para o email eventos@portugal-a-programar.pt

NOTICIAS

“Dar Voz ao AVC" aplicação informática para ajuda a pessoas com AVC

“Dar Voz ao AVC" é uma aplicação de software recentemente desenvolvida por Fethi Houita, ex-aluno do curso de Engenharia Informática [LEInf] da Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Viana do Castelo [ESTG-IPVC].

A aplicação (app) visa poder dotar as pessoas que sofreram AVC (e ficaram afásicas), de uma ferramenta para poder expressar sentimentos, emoções, e desejos simples. É uma aplicação que emula parte do trabalho do terapeuta da Fala, com elementos de comunicação alternativa que, em vez de estarem no papel, são dotados de maior interactividade e têm som, comportando mais vantagens.

Nem todas as pessoas que sofreram AVC e têm afasia poderão usar a app, porque necessitam que a compreensão esteja intacta e de possuir recursos intelectuais e cognitivos para utilizar o tablet e a app.

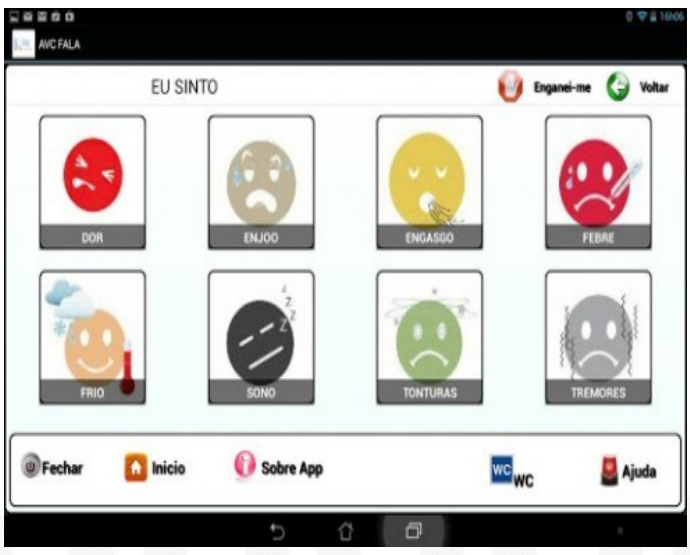
Para Jorge Ribeiro, atual coordenador do curso de Engenharia Informática, este e outros projetos de ex-alunos refletem o conhecimento e as competências adquiridas ao longo do curso, possibilitando-lhes reunir um conjunto de competências e “know-how” na área da Engenharia Informática. Reflete também o facto de permitir aos alunos estarem preparados para “saber fazer e por as mãos na massa”, refletindo-se o seu desempenho na elevada taxa de empregabilidade e na boa reputação do curso por parte dos empregadores, quer a nível regional, nacional e internacional.

Adicionalmente, este é mais um exemplo de sucesso em empreendedorismo.

Link da aplicação: https://play.google.com/store/apps/details?id=darvozao.avc

Link para a página de Fethi Houita, ex-Aluno ESTG-IPVC: fr.linkedin.com/pub/fethi-houita/4b/627/185

Fonte «IPVC.pt»



Português garante lugar no muro da fama da Adobe

Um jovem de Leiria garantiu um lugar no "muro da fama" dos caçadores de bugs da Adobe, com a descoberta de uma falha que permitia injetar código malicioso.

Manuel Sousa, 17 anos, garantiu um lugar na lista de programadores que descobrem bugs e falhas de segurança nas soluções da Adobe. A entrada nesta página que disntingue descobridores de falhas ocorreu há três semanas, depois de alertar os responsáveis da Adobe para uma falha que poderia ser usada para ataques de Cross Site Scripting (XSS).

«A falha permitia injetar código-fonte na página, precisamente naqueles números de navegação que costuma estar no fundo de algumas páginas de procura, para avançar de uma página para as seguintes», explica Manuel Sousa num e-mail enviado para a Exame Informática.

Os ataques de XSS distinguem-se por permitirem a inserção de códigos que alteram parte ou a totalidade de um site. Este tipo de falhas pode ser usada para alterar a aparência do site, ou, numa variante mais comum e grave, para inserir módulos que aparentam pertencer ao site legítimo, mas que na verdade encaminham os internautas para um endereço malicioso, que poderá ser usado para suportar estratagemas que desviam dados confidenciais.

Segundo Manuel Sousa, a falha «afetava todo o domínio adobe.com», não sendo necessário o utilizador estar registado no site da companhia que produz o Photoshop para se tornar vítima de um hacker que tentasse explorar a vulnerabilidade

O jovem de Leiria descobriu a falha ao cabo de uma semana de análise ao site principal da Adobe. A companhia demorou cerca de um mês a remendar a vulnerabilidade – reconhecendo o mérito do jovem finalista do secundário com a atribuição de um lugar no muro da fama dos caçadores de bugs.

Não é a primeira vez que Manuel Sousa garante uma entrada direta para um dos muros da fama dos caçadores de bugs: no início do ano, o jovem de 17 anos recebeu idêntica distinção da Google – só que dessa vez a descoberta de uma falha XSS valeu-lhe também um prémio de 3000 dólares.

«Só costumo realizar testes em sites que têm programas já estabelecidos para ajudar os "caçadores" a reportarem falhas. Algumas dessas empresas oferecem prémios monetários, distinções nos muros da fama, ou apenas brindes», explica.

Além da Google e da Adobe, Manuel Sousa também se distinguiu com a descoberta de uma falha no Portal das Finanças que permitia ataques de phishing e o consequente desvio de dados dos internautas.

Fonte «exameinformatica.sapo.pt»

Blog: http://manuel-sousa.blogspot.pt/

TEMA DE CAPA

Estendendo uma aplicação criada no App Studio da Microsoft

Estendendo uma aplicação criada no App Studio da Microsoft

Este artigo tem como objetivo explicar como extender uma aplicação que foi inicialmente gerada pelo App Studio da Microsoft.

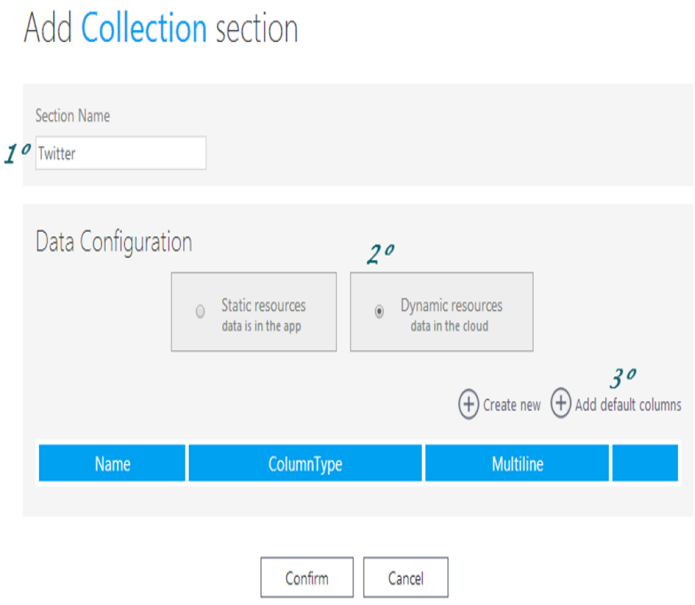
No meu último artigo, Criando aplicações Windows Phone 8.1 e Windows 8.1 usando o App Studio da Microsoft, foi apresentado o serviço da App Studio, os vários tipos de templates para criar aplicações, as várias fases do desenvolvimento da aplicação no serviço e por fim a geração dos pacotes e a obtenção do código.

Atualmente o App Studio não tem suporte para feed do Twitter, no entanto é possível extender a aplicação para suportar esta funcionalidade, ora vejamos como o podemos fazer.

Na aplicação do Contoso Ltd, criada a partir de um template fornecido pelo App Studio temos várias secções:

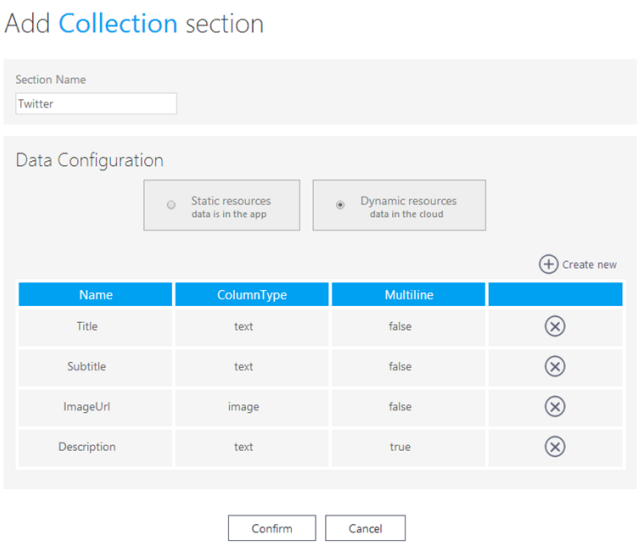
- About us é uma secção do tipo HTML
- Catalog é uma secção do tipo coleção dinâmica
- Team é uma secção do tipo coleção dinâmica
- News é uma secção do tipo Bing
- Contact us é uma secção do tipo Menu, que contém ações de menus.

Para adicionar o feed do Twitter, iremos ter uma nova sessão chamada Twitter. Esta secção pode ser criada no App Studio usando uma das opções de secções avançadas, a Collection.

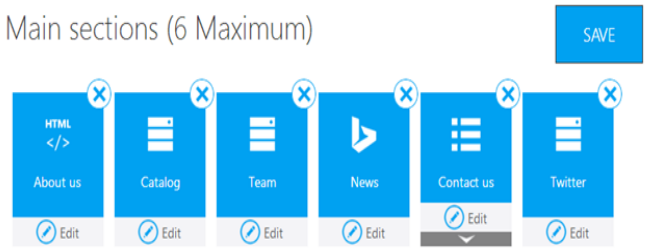


Ao criar a secção do tipo Collection, temos que definir o Nome, escolher a opção “Dynamic resources data in the cloud” e em seguida clicar em “Add default columns” (irá criar toda a estrutura de dados necessária para a coleção, caso o utilizador pretenda criar a sua própria estrutura deve usar a opção “Create new”).

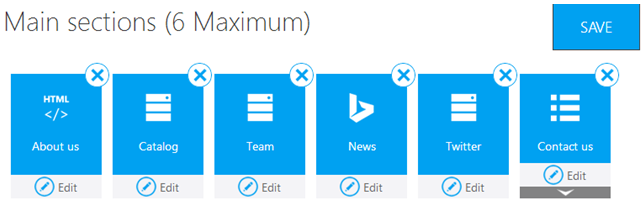
O resultado será o seguinte,



Depois de confirmar, iremos ter mais uma seção na página de conteúdos

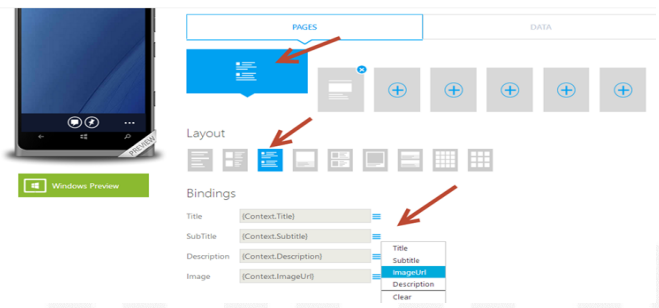


Como referi no meu artigo anterior, podemos alterar a ordem das seções, fazendo arrastar e largar (Drag & Drop), desta forma iremos ter

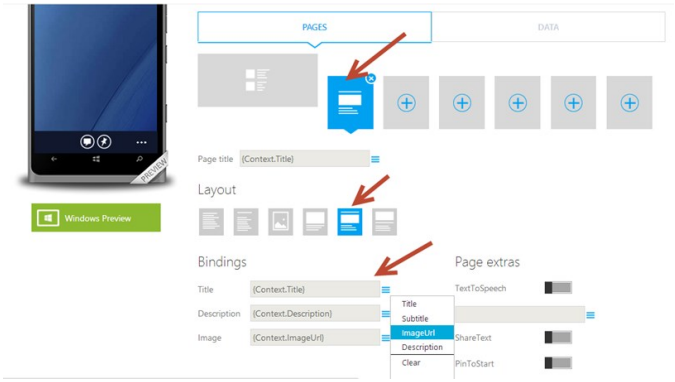


De seguida, devemos editar a seção Twitter, para definir uma entrada de dados para teste e para definir os Bindings do DataTemplates.

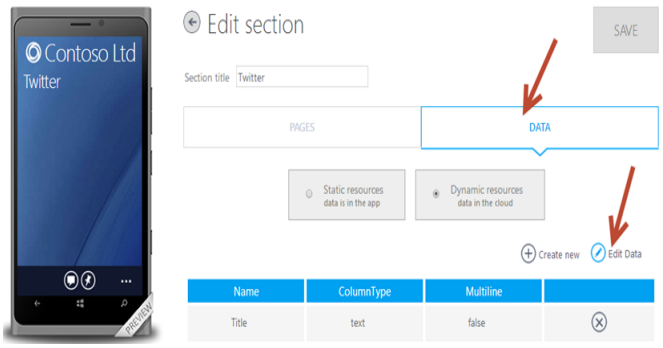
Comecemos por definir os Bindings, primeiro da página principal:

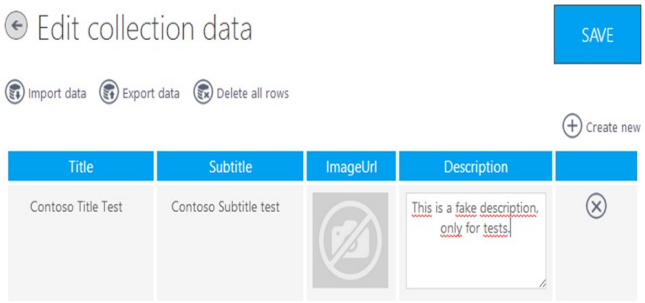


Em seguida da página de detalhe:



Por fim vamos criar uma entrada falsa, no separador de dados:





É possível importar dados de um ficheiro CSV ou inserir manualmente, neste caso, apenas foi criada uma entrada manualmente.

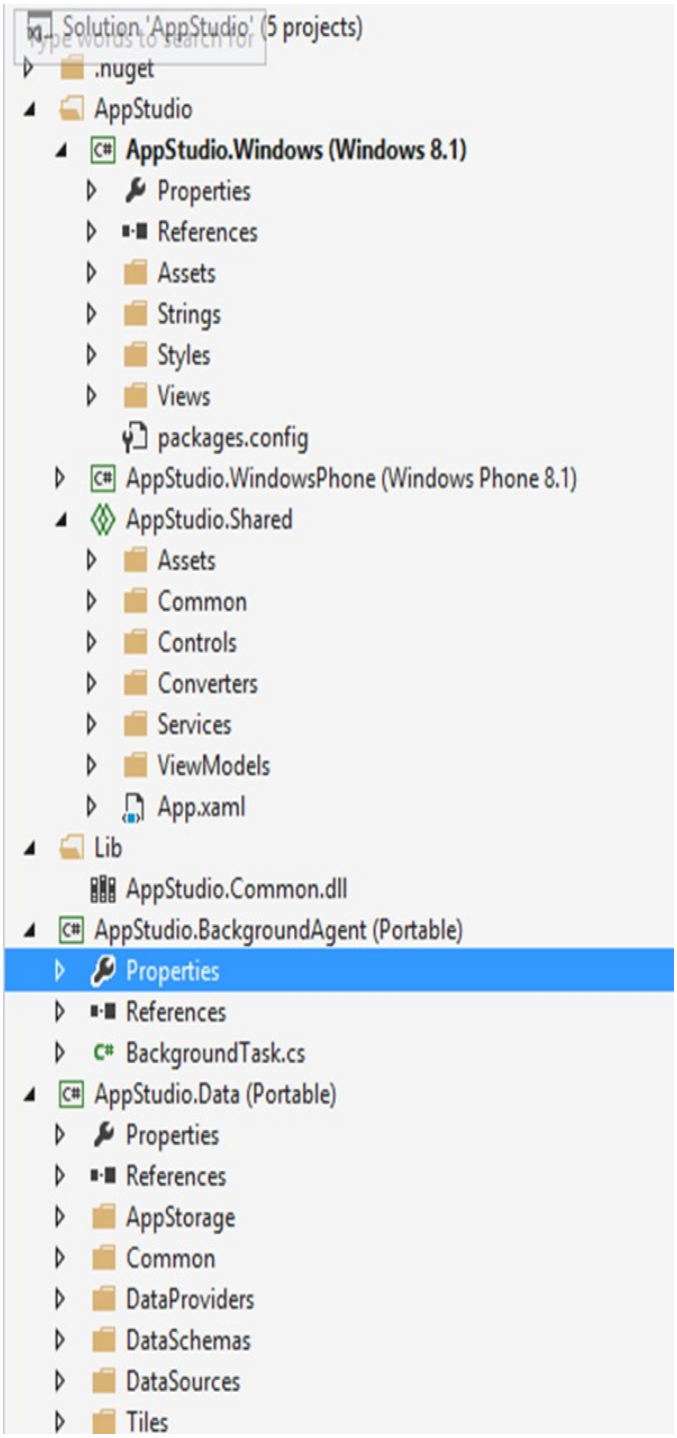
Neste momento, estamos prontos para gerar novos pacotes e com isto obter nova versão do código fonte, para assim continuar o desenvolvimento no Visual Studio 2013.

No Visual Studio, o projeto irá ter a seguinte estrutura:

- AppStudio.Windows – projeto para a aplicação Windows 8.1
- AppStudio.WindowsPhone – projeto para a aplicação Windows Phone 8.1

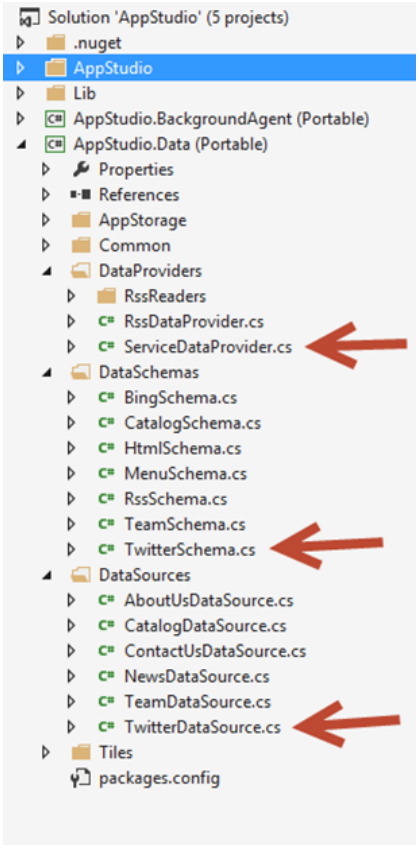
EXTENDENDO UMA APLICAÇÃO CRIADA NO APP STUDIO DA MICROSOFT

- AppStudio.Shared – projeto partilhado pelas duas aplicações
- AppStudio.BackgroundAgent – projeto que dá suporte a background agents
- AppStudio.Data – projeto que consiste numa portable class library e contém toda a informação sobre a estrutura de dados usadas nas duas aplicações e contém ainda os "providers" de dados (facebook, feeds,…)



O resultado do deploy da aplicação no Windows Phone (secção do Twitter) é

Para alterar esta página, para que use dados reais do Twitter é necessário fazer algumas alterações no projecto AppStudio.Data



Em TwitterDataSource, devemos alterar o método LoadData para:

```
public   async   Task<IEnumerable<TwitterSchema>>
LoadData()
  {
    if (_data == null)
     {
       try
        {
          var serviceDataProvider =
                        new TwitterDataProvider();
         _data = await serviceDataProvider.Load();
         }
         catch (Exception ex)
          {
           AppLogs.WriteError
           ("TwitterDataSource.LoadData",
                                  ex.ToString());
                }
            }
            return _data;
        }
```

Em vez de usar o ServiceDataProvider devemos criar o TwitterDataProvider que será

```
public class TwitterDataProvider
    {
        TwitterContext twitterCtx;

        public TwitterDataProvider()
        {
            var auth = new SingleUserAuthorizer()
            {
                CredentialStore =
              new SingleUserInMemoryCredentialStore
                {
                    ConsumerKey = ConsumerKey,
                    ConsumerSecret =
                                    ConsumerSecret,
                    AccessToken = AccessToken,
                    AccessTokenSecret =
                                   AccessTokenSecret
                }
            };

            twitterCtx = new TwitterContext(auth);
        }


        public async Task
                <IEnumerable<TwitterSchema>> Load()
        {
            var searchresults = await (from tweet
              in twitterCtx.Search where tweet.Type
             == SearchType.Search && tweet.Query ==
                   "#contoso" && tweet.Count == 100
              select tweet).SingleOrDefaultAsync();

            var data = new List<TwitterSchema>();
            foreach (var status in
                            searchresults.Statuses)
            {
                data.Add(new TwitterSchema()
                {
                    Title = status.User.Name,
                    Description = status.Text,
                    ImageUrl =
                       status.User.ProfileImageUrl,
                    Subtitle =
                     status.CreatedAt.ToString("t")
                });
            }
            return data;
        }
```

Para obter as chaves

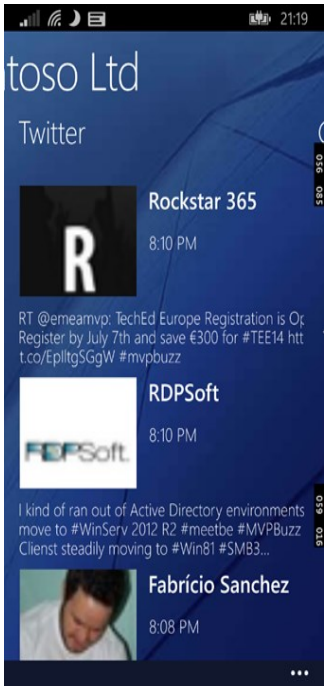
```
const string ConsumerKey = "….";
const string ConsumerSecret = "…";
const string AccessToken = "… ";
const string AccessTokenSecret = "…";
```

deverá aceder ao dev.twitter.com e criar uma aplicação e em seguida obterá as chaves.

É necessário instalar o pacote do Nuget linqtoTwitter no projeto AppStudio.Data, mas como este projeto é uma Portable Class Library com targets para Windows Phone 8.1 e Windows 8.1, a instalação do pacote irá falhar. Para dar a volta a este problema, uma vez que ainda não existe uma atualização, deve-se instalar o pacote no projecto AppStudio.Windows e depois no projecto AppStudio.Data, deve-se fazer *Add Reference>Browser* e em seguida procurar pela pasta que contém o pacote instalado, que será algo do género

App\packages\linqtotwitter.3.0.2\lib\portable-win8+net45+wp8 e deve-se selecionar a dll *LinqToTwitterPcl.dll* e assim o projeto AppStudio.Data irá ter a referência do LinToTwitterPcl.

Depois de correr a aplicação iremos ter o seguinte resultado, para Windows Phone 8.1



Secção da página principal

Página de detalhes

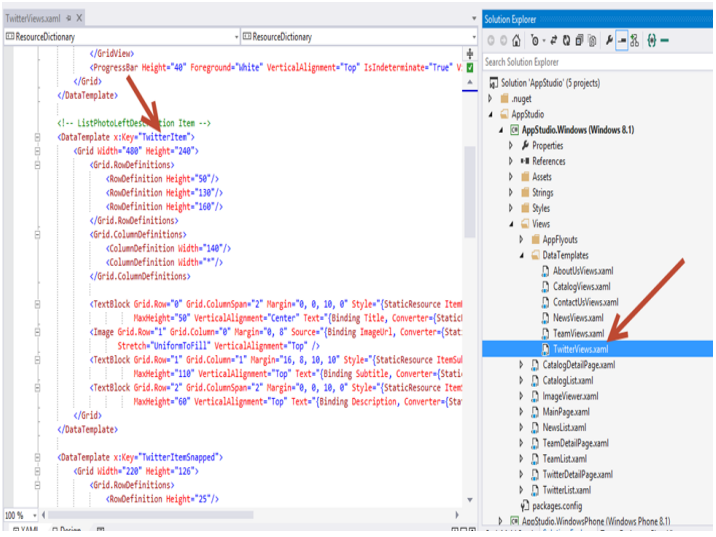
A versão Windows 8.1 tem o seguinte aspeto

TEMA DA CAPA

EXTENDENDO UMA APLICAÇÃO CRIADA NO APP STUDIO DA MICROSOFT

Como se pode observar o aspeto da interface para a secção do Twitter, não é muito agradável e como tal podemos alterar o código XAML para assim obtermos um resultado melhor. Outro aspeto que podemos alterar é o facto de ao clicar no item navegamos para a página de detalhe, o que neste caso não faz muito sentido porque irá apresentar a mesma informação que na página inicial, por tanto iremos remover.

No projeto de cada aplicação, podemos encontrar uma pasta chamada Views que contém uma pasta chamada DataTemplates, que por sua vez contém o dicionário de recursos para cada secção. Se abrirmos o ficheiro TwitterViews, encontramos o DataTemplate TwitterItem que representa cada item do feed do Twitter.



Para a versão do Windows Phone 8.1, podemos alterar

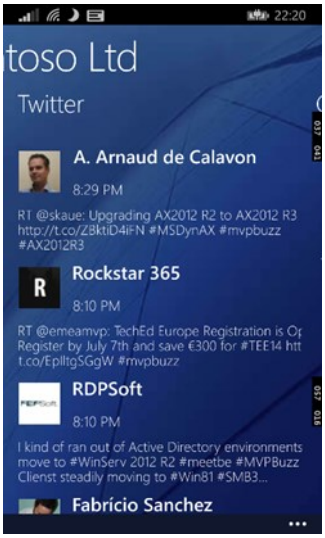
```
<DataTemplate x:Key="TwitterList">
        <Grid>
            <ListView ItemsSource="{Binding
                  Items}" SelectedItem="{Binding
                   NavigationItem, Mode=TwoWay}"
                    SelectionMode="None"
                                IsSwipeEnabled="False"
   ScrollViewer.VerticalScrollBarVisibility="Hidden"
                    ItemTemplate="{StaticResource
                                        TwitterItem}">
            </ListView>
            <ProgressBar Width="380" Height="40"
        Foreground="White" VerticalAlignment="Top"
       IsIndeterminate="True" Visibility="{Binding
                        ProgressBarVisibility}" />
        </Grid>
    </DataTemplate>

    <!-- ListPhotoLeftDescription Item -->
    <DataTemplate x:Key="TwitterItem">
        <Grid Width="380">
            <Grid.RowDefinitions>
                <RowDefinition Height="80"/>
                <RowDefinition Height="60"/>
            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="50"/>
                <ColumnDefinition Width="*"/>
            </Grid.ColumnDefinitions>

            <Image Grid.Row="0" Grid.Column="0"
         Margin="0, 8" Source="{Binding ImageUrl}"
                     MaxHeight="50" MaxWidth="50"
                   Stretch="UniformToFill"
                    VerticalAlignment="Center" />

          <StackPanel Grid.Row="0"
                                  Grid.Column="1">
            <TextBlock Margin="16, 8, 5, 10"
       Style="{StaticResource ItemHeaderWrapText}"
        MaxHeight="50" VerticalAlignment="Center"
                        Text="{Binding Title,
    Converter={StaticResource TextPlainConverter},
                     ConverterParameter=140}" />
            <TextBlock Margin="16, 8, 5, 10"
        Style="{StaticResource ItemSubheaderText}"
           MaxHeight="110" VerticalAlignment="Top"
                        Text="{Binding Subtitle}" />
          </StackPanel>

            <TextBlock Grid.Row="1"
          Grid.ColumnSpan="2" Margin="0, 0, 5, 0"
           Style="{StaticResource ItemSmallText}"
          MaxHeight="200" VerticalAlignment="Top"
                    Text="{Binding Description}" />
        </Grid>
    </DataTemplate>
```

Para a versão de Windows 8.1, podemos alterar

```
<DataTemplate x:Key="TwitterListSnapped">
        <Grid Width="230">
            <GridView  ItemsSource="{Binding  Pre-
viewItems}" SelectedItem="{Binding NavigationItem,
Mode=TwoWay}"
                      SelectionMode="None"
IsSwipeEnabled="False"  ScrollViewer.VerticalScrollBarVisibility="Hidden"
                    ItemTemplate="{StaticResource
TwitterItemSnapped}">
            </GridView>
            <ProgressBar  Height="40"  Foreground="White" VerticalAlignment="Top" IsIndeterminate="True" Visibility="{Binding ProgressBarVisibility}" />
        </Grid>
    </DataTemplate>

    <!-- ListPhotoLeftDescription Item -->
    <DataTemplate x:Key="TwitterItem">
        <Grid Width="480" Height="240">
            <Grid.RowDefinitions>
                <RowDefinition Height="0"/>
                <RowDefinition Height="130"/>
                <RowDefinition Height="160"/>
```

```
</Grid.RowDefinitions>
<Grid.ColumnDefinitions>
<ColumnDefinition Width="140"/>
<ColumnDefinition Width="*"/>
</Grid.ColumnDefinitions>

<Image Grid.Row="1" Grid.Column="0" Margin="0, 8"
Source="{Binding ImageUrl, Converter=
            {StaticResource ThumbnailConverter},
         ConverterParameter=220}" MaxHeight="110"
                  Stretch="UniformToFill"
                     VerticalAlignment="Top" />
           <StackPanel Grid.Row="1"
          Grid.Column="1" Orientation="Vertical">
                <TextBlock   Margin="16, 0, 10,
   0" Style="{StaticResource ItemHeaderWrapText}"
                                    MaxHeight="50"
 VerticalAlignment="Center"  Text="{Binding Title,
Converter={StaticResource TextPlainConverter},
                     ConverterParameter=140}" />
                <TextBlock  Margin="16, 8, 10,
   10" Style="{StaticResource ItemSubheaderText}"
                                  MaxHeight="110"
 VerticalAlignment="Top" Text="{Binding Subtitle,
 Converter={StaticResource TextPlainConverter},
                     ConverterParameter=280}" />
            </StackPanel>
                          <TextBlock Grid.Row="2"
        Grid.ColumnSpan="2" Margin="0, 0, 10, 0"
          Style="{StaticResource ItemSmallText}"
                                    MaxHeight="60"
                     VerticalAlignment="Top"
                  Text="{Binding Description}" />
        </Grid>
    </DataTemplate>
```

E assim obtemos um novo aspeto da aplicação, para Windows Phone 8.1 iremos ter:



E para Windows 8.1 iremos ter

Em conclusão, conclui-se que é muito simples extender a aplicação gerada pelo App Studio, no entanto não é possível fazer o upload desta versão para o App Studio, todas as alterações feitas na aplicação no App Studio implicam um "merge" entre versões.

AUTOR

Escrito Por Sara Silva

Licenciada em Matemática – Especialidade em Computação, pela Universidade de Coimbra e é Microsoft Certified Profissional Developer. Atualmente o seu foco de desenvolvimento incide em Windows Phone e Windows 8 Store Apps. O seu Blog **é** www.saramgsilva.com e o twitter é @saramgsilva.

A PROGRAMAR

JSF - Parte 3 (Managed beans)

Pascal – array de argumentos

Criar uma aplicação para Android com mapa

C# CRUD (Create, Read, Update & Delete)

JSF - Parte 3 (Managed beans)

Este é o 3º artigo da série. Assume portanto que já tem um projeto de exemplo JSF a funcionar. É altura de fazer algumas experiências, nomeadamente criar um "backing bean".

Um backing bean é uma classe Java que responde a pedidos e gere o estado dos componentes JSF. É uma espécie de "servlet de alto nível". Entre várias outras funções, faz a ligação à camada dos serviços que pode ser uma BD, um servidor remoto ou outro. Uma aplicação pode conter múltiplos backing beans.

Os backing beans mais conhecidos em Java são os "managed beans" e os "named beans" (ou CDI beans). Os primeiros são mais comuns; os segundos mais flexíveis. Os conceitos são muito semelhantes. Usaremos os primeiros devido à facilidade de utilização e porque o Tomcat apenas suporta esses.

Um managed bean representa o Controller no padrão MVC. Caracteriza-se consoante o seu âmbito (scope). Os mais importantes são:

•**Application**: o bean persiste na aplicação estando disponível para todas as sessões;

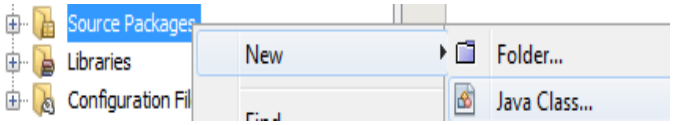
•**Session**: o bean existe por utilizador; é útil para dados referentes ao que utilizador está a fazer durante um periodo de utilização;

•**View**: semelhante ao "session", mas por "tab"; é como que se cada "tab" do browser representasse uma sessão diferente;

•**Request**: a única gestão de estado que existe dura apenas o pedido e por isso é o mais leve.

Para exemplificar como funciona um managed bean… criemos um. Para efeitos de aprendizagem, comecemos com um bean de sessão, o que apresenta menos desafios. Para tal, após ter o projeto JSF (criado no artigo anterior) aberto:

1. Clique com o botão direito do rato em "Source Packages" e faça "New" > "Java Class…":



2. Dê-lhe o nome de "ClientesBean" e coloque-o no package "pt.revista.programar".

3. O bean criado estará automaticamente disponível para ser usado nos XHTML sob a forma de "clientesBean" (técnica conhecida como convention over configuration). Para tal, anote a classe com as anotações @ManagedBean e @SessionScoped (disponíveis em javax.faces.bean).

4. Agora, crie uma propriedade com getter/setter chamada "nome". Ficará com o seguinte código:

```
package pt.revista.programar;

import java.io.Serializable;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;

@ManagedBean
@SessionScoped
public class ClientesBean implements Serializable
{

String nome;

      public string getNome(){
            return nome;
      }

      Public void setNome(String nome) {
            this.nome = nome;
      }
}
```

5. Agora, na pasta "Web Pages", vá a "index.html" e dentro de "h:body" coloque o seguinte código:

```
<h:form>
 <h:inputText Value="#{clientsBean.nome}" />
 <h:commandButton Acton="resultado
                                  "value="enviar" />
</h:form>
```

Tal código irá comunicar com o managed bean criado anteriormente. O botão irá submeter o formulário.

6. Agora precisamos da página para onde seremos redirecionados. Teremos de chamá-la de "resultado.xhtml" para que a "action" no botão anterior seja respeitada.Para criar está página carregue com o botão direito sobre "Web Pages" > "New" > "Other" > "Web" > "XHTML" > "Next"; dê-lhe o nome "resultado" e faça "Finish".

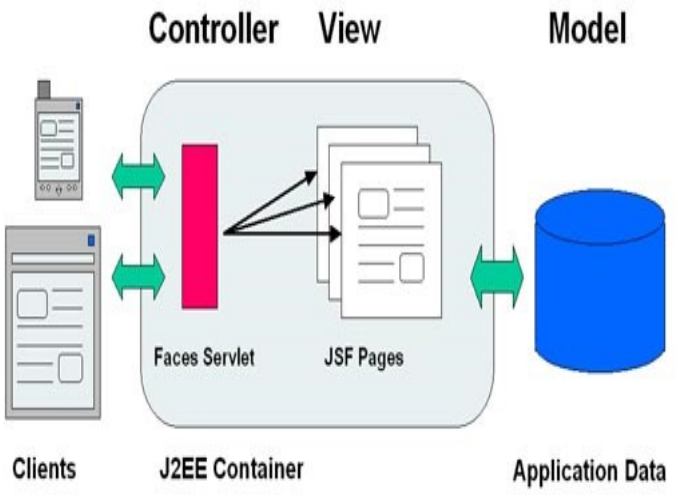
7. Apague todo o conteúdo e copie o conteúdo de "index.html".

8. Vamos apenas substituir o conteúdo de "h:body" de maneira a ficar:

```
<h:body>
    <h:outputText value="Boa tarde #
                              {clientesBean.nome}" />
</h:body>
```

JSF - PARTE 3 (MANAGED BEANS)

9. Agora está tudo em ordem. Já temos uma página que recebe o seu nome, um bean que o guarda e uma página que o exibe. Para ver tudo em ação, clique com o botão direito no projeto e faça "Run".



O JSF é um mundo construído em cima das tecnologias que já conhece (HTTP, servlets, JavaScript, etc.), pelo que por vezes é estranho que seja tão alto nível. Após alguns anos a usá-lo posso garantir que depois do período inicial de aprendizagem, compensa: a velocidade de desenvolvimento aumenta consideravelmente e damos por nós a fazer coisas muito complexas em muito pouco tempo.

(1 of 5) 1 2 3 4 5 10

Id	Year	Brand	Color
7859102e	1984	Volkswagen	Maroon
f1416fce	1962	Honda	Green
82388fa0	2000	Volvo	White
1bdd8d20	1970	Mercedes	Silver
8a4d149a	1971	Renault	Brown
742a65a5	1965	Renault	Yellow
f3859980	2007	Volvo	Orange
92521131	1969	Audi	Silver
01ea11ea	1981	Volvo	Silver
99a9cad8	1998	Volkswagen	Red

(1 of 5) 1 2 3 4 5 10

Há que assinalar também que, ao usar JSF, devemos tentar sempre conhecer e fazer uso das suas funcionalidades.

> ***"(...) É uma espécie de "servlet de alto nível". Entre várias outras funções, faz a ligação à camada dos serviços que pode ser uma BD, um servidor remoto ou outro (...)Um managed bean representa o Controller no padrão MVC (...)"***

Isto porque é provável que ele resolva o nosso problema e não seja preciso escrever código específico. Seja chamadas Ajax, personalização do componente, utilização da query string, entre outros, o JSF já pensou nisso. Se precisamos de um componente mais avançado, existem as bibliotecas de componentes (ICEfaces, RichFaces, PrimeFaces), que trazem componentes muito ricos, flexíveis e funcionais. É improvável que tenhamos de recorrer a código JavaScript para operações comuns; a ideia de usar uma framework server-side é mesmo essa: ser tudo personalizável por via dos atributos XHTML dos componentes.

Espero que tenha gostado do artigo. Qualquer dúvida, sinta-se à vontade para me contactar. Consoante o interesse da comunidade poderei escrever mais artigos sobre o tema, nomeadamente sobre a integração e utilização do PrimeFaces.

AUTOR

Escrito por Luís Soares

Formado em Engenharia Informática e de Computadores no Instituto Superior Técnico (Licenciatura e Mestrado). Sou *web developer, _tendo já colaborado em projetos de telecomunicações e dos _media*. Gosto de linguagens de alto nível, de reutilizar código, de refactoring para simplificar. Gosto de ensinar. Escrevi um livro sobre jQuery (goo.gl/nw2Zb).
Os meus contactos estão em luissoares.com para qualquer dúvida sobre o artigo ou outra informação.

Pascal – array de argumentos

A definição da linguagem Pascal estabelece desde os seus primórdios regras bastante rígidas acerca da passagem de argumentos a uma função ou procedimento. No seu conjunto, uma das consequências destas regras é a impossibilidade de se implementarem **funções variádicas**, isto é, funções com um número indefinido de argumentos às quais podemos fornecer virtualmente uma infinidade de argumentos, não havendo a restrição de ser possível passar apenas N argumentos em determinada ordem e com determinados tipos.

Várias linguagens permitem a implementação de funções variádicas, como por exemplo C e até Haskell (com recurso a alguns truques que envolvem aspectos avançados dos tipos de dados).

O facto de Pascal não permitir a implementação de funções variádicas pode suscitar algumas dúvidas. Métodos *standard* que formam a base do Pascal são aparentemente variádicos, como o writeln e o readln. No entanto, estes métodos não são exactamente funções ou procedimentos verdadeiros. A sua implementação não é definida em termos da linguagem Pascal (que, como foi dito, não permite este tipo de definições); são apenas instruções que o compilador trata de forma especial para nos permitir utilizá-las como se fossem variádicas.

Todavia, o rio não encontra a sua foz neste ponto. Apesar de esta funcionalidade não ser permitida, existe uma forma de a simular. Compiladores e dialectos mais recentes, como o Delphi e o Free Pascal, permitem a passagem de ***arrays* de argumentos**, a qual tem por princípio a passagem de *arrays* abertos, conceito que será brevemente revisto.

Todo o código presente neste artigo foi compilado recorrendo ao *Free Pascal Compiler*, versão 2.6.2, em ambiente *Windows*®, sendo totalmente portável para outras plataformas.

Passagem de *open arrays*

Quando o Pascal foi dado a conhecer ao mundo em 1971, os *arrays* eram estáticos, com uma dimensão bem definida, uma vez que na altura não fora ainda introduzido o conceito de ***array* dinâmico**.

Com o evoluir da linguagem, este conceito foi implementado, e pela primeira vez os *arrays* não necessitavam de ter uma dimensão bem definida – esta podia ser controlada em *runtime*. Dadas as suas características, tornou-se uma forma de criar o equivalente a pequenas listas ligadas com muito maior segurança uma vez que a gestão de recursos não fica a cargo do programador. De referir que os *arrays* dinâmicos têm os seus dados contíguos em memória à semelhança de um *array* estático e ao contrário das listas ligadas.

Da mesma forma, deixou de ser obrigatória a criação de tipos de dados para a passagem de *arrays* em argumentos. Se antes era necessário um tipo para cada dimensão e um procedimento para cada tipo, o trabalho do programador fora imensamente facilitado.

Por exemplo, pode-se receber um *array* de números inteiros de pequena dimensão e sem sinal da seguinte forma:

```
function Average(list : array of byte) : real;
```

Recorrendo às funções **Low** e **High**, e mais recentemente com a estrutura de repetição **for-in**, torna-se simples iterar os elementos do *array*.

```
FUNCTION AVERAGE(LIST : ARRAY OF BYTE) : REAL;
VAR I : BYTE;
BEGIN
   AVERAGE := 0.0;
   FOR I:=LOW(LIST) TO HIGH(LIST) DO
      AVERAGE := AVERAGE + LIST[I]);
   AVERAGE := AVERAGE / LENGTH(LIST);
END;
```

Outra característica de elevado interesse é o facto de se poder passar um fragmento de um array. Vejamos o seguinte exemplo:

```
VAR XS : ARRAY[1..20] OF BYTE;
// ...
WRITELN(AVERAGE(XS[10..20]):0:3);
```

Neste caso, apenas é do nosso interesse calcular a média da segunda metade do *array* **xs**. Portanto, definimo-lo com a sintaxe **xs[10..20]**. Desta forma, apenas os elementos do índice 10 ao 20 são passados à função **Average**.

Todas estas funcionalidades compõe aquilo a que se denomina de **passagem de *arrays* abertos** (em inglês, ***open arrays***): a possibilidade de receber um *array* sem conhecimento prévio da sua dimensão, bem como a possibilidade de realizar a passagem parcial de um *array*.

Passagem de um *array* de argumentos

A evolução da passagem de *arrays* abertos levou naturalmente à implementação de uma funcionalidade útil: o ***array of const***.

Classicamente, os elementos de um *array* são de um tipo bem definido. No entanto, ***const*** é uma palavra reservada que define constantes. Desta forma, um *array of const* define algo diferente: é um *array* cujos elementos são constantes de tipo indefinido.

PASCAL – ARRAY DE ARGUMENTOS

Como primeira nota, há que referir que estruturas não podem ser passadas nestes *arrays*. Apenas tipos de dados simples (tipos numéricos, alfanuméricos e apontadores), objectos, classes e interfaces podem.

Vamos criar um procedimento que recebe um *array of const* e nos dê informações acerca dos argumentos recebidos:

```
PROCEDURE FOO(ARGS : ARRAY OF CONST);
```

Quando o procedimento recebe este *array*, ocorre uma conversão dos seus elementos para um *record* variante com características particulares. Segundo a documentação do Free Pascal, esta é a sua definição:

```
TYPE
PTRINT = LONGINT;
PVARREC = ^TVARREC;
TVARREC = RECORD
  CASE VTYPE : PTRINT OF
    VTINTEGER    :
      (VINTEGER: LONGINT);
    VTBOOLEAN    :
      (VBOOLEAN: BOOLEAN);
    VTCHAR       :
      (VCHAR: CHAR);
    VTWIDECHAR   :
      (VWIDECHAR: WIDECHAR);
    VTEXTENDED   :
      (VEXTENDED: PEXTENDED);
    VTSTRING     :
      (VSTRING: PSHORTSTRING);
    VTPOINTER    :
      (VPOINTER: POINTER);
    VTPCHAR      :
      (VPCHAR: PCHAR);
    VTOBJECT     :
      (VOBJECT: TOBJECT);
    VTCLASS      :
      (VCLASS: TCLASS);
    VTPWIDECHAR  :
      (VPWIDECHAR: PWIDECHAR);
    VTANSISTRING :
      (VANSISTRING: POINTER);
    VTCURRENCY   :
      (VCURRENCY: PCURRENCY);
    VTVARIANT    :
      (VVARIANT: PVARIANT);
    VTINTERFACE  :
      (VINTERFACE: POINTER);
    VTWIDESTRING :
      (VWIDESTRING: POINTER);
    VTINT64      :
      (VINT64: PINT64);
    VTQWORD      :
      (VQWORD: PQWORD);
  END;
```

Portanto, cada elemento será um *record* no qual o campo **VType** indica qual o tipo de dados do argumento, e conforme o valor deste campo haverá um segundo que contém o valor do argumento. Por exemplo, se **VType** for **vtString**, então o argumento é do tipo **ShortString** (comummente designado apenas como *string*), e o seu valor pode ser acedido através do campo **VString**.

Uma vez que cada elemento pode ter um tamanho em *bytes* diferente dos restantes, o armazenamento dos dados referentes aos argumentos não pode ser feito da forma tradicional: um *array* clássico tem os seus elementos armazenados em áreas contíguas de memória em que cada elemento ocupa exactamente N bytes. Neste caso, cada argumento terá a sua localização num bloco da memória distinto. Isto explica o facto de haver um apontador (o tipo **PVarRec**).

Conclui-se que um *array of const* é, em última instância, um *array of TVarRec* dentro do procedimento ou função. Mas atenção, nunca se deve declarar o argumento desta forma!

```
PROCEDURE FOO(ARGS : ARRAY OF TVARREC);
```

Para implementar esta função, vamos primeiramente definir o seu objectivo:

- Receber um *array of const* e devolver, no monitor, o tipo de dados de cada argumento, assim como o seu valor ou nome.

Para tal, será útil controlar quantos argumentos foram passados. Esta informação pode ser obtida com a função **Length**. Caso não haja argumentos, iremos mostrar a mensagem "Sem argumentos."

```
PROCEDURE FOO(ARGS : ARRAY OF CONST);
VAR I : SMALLINT;
BEGIN
  IF LENGTH(ARGS) > 0 THEN
    // ANÁLISE DOS ARGUMENTOS
  ELSE
    WRITELN('SEM ARGUMENTOS.');
END;
```

Para proceder à análise dos argumentos, iremos iterar pelos elementos do *array*. Sendo este um *open array*, cuja dimensão desconhecemos, necessitaremos das funções **Low** e **High** que devolvem, respectivamente, os índices menor e maior do *array* (isto é, as posições do primeiro e último elemento).

```
FOR I:=LOW(ARGS) TO HIGH(ARGS) DO
  // ANÁLISE DOS ARGUMENTOS
```

Não sendo objectivo do presente artigo fazer uma exploração exaustiva de todos os tipos de dados possíveis de serem passados no *array of const*, iremos tratar apenas os mais comuns e alguns que possuem algumas particularidades.

Como foi referido, os elementos do *array* são convertidos ao *record* infra-apresentado, pelo que possuem **campos**. Naturalmente, uma estrutura de decisão **case-of** irá analisar o valor do campo **vtype**, informando acerca do tipo de dados do argumento. Caso não seja conhecido, mostrar-se-á a mensagem "desconhecido". Comecemos por analisar um caso simples: o argumento é um **Integer**. Desta forma, **vtype** irá assumir o valor **vtInteger** (uma constante do tipo **LongInt**):

```
CASE ARGS[I].VTYPE OF
        VTINTEGER :
          WRITELN('INTEGER = ',
```

Sendo um **Integer**, o campo variante **vinteger** possui o valor deste argumento. Notificamos acerca do tipo de dados do argumento, seguido do valor. O mesmo se aplica aos tipos de dados **Boolean** e **Char**.

No entanto, alguns tipos de dados necessitam de uma forma diferente de aceder ao valor. Alguns campos variantes são, na verdade, apontadores (o seu tipo de dados começa por **P**, como por exemplo **PShortString**). Para alguns destes necessitamos de recorrer ao operador **^**, o qual nos indica o valor armazenado no ponteiro que lhe fornecemos (**apontador^**). Para outros casos, necessitaremos de fazer ***type casting*** uma vez que os seus tipos de dados permitem a recepção de um apontador, devolvendo automaticamente o valor lá armazenado (por exemplo, o tipo de dados **AnsiString**).

Comecemos por analisar o tipo de dados **Extended**:

```
VTEXTENDED :
  WRITELN('EXTENDED = ',
  ARGS[I].VEXTENDED^);
```

Este é um caso simples. No entanto, as *strings* têm algumas diferenças. No Pascal moderno (leia-se Free Pascal, Object Pascal e Delphi), não existe apenas um tipo de dados *string*. O tipo de dados *string* apareceu após o aparecimento do Pascal, e apenas podia armazenar 255 caracteres. Hoje em dia, existem vários tipos de dados da família da *string*, sendo os mais proeminentes os seguintes:

- **ShortString –** apenas permite 255 caracteres;
- **AnsiString –** é *null-terminated* e não tem limite de caracteres;
- **WideString –** semelhante ao *AnsiString*, cada caracter ocupa 2 bytes ao invés de apenas 1, e permite armazenar caracteres no formato UTF-16.

Desta forma, uma **ShortString** é o equivalente a um *array of char* com 255 elementos. Naturalmente, o argumento será um ponteiro para a localização deste *array*. Internamente, o compilador trata as *strings* de forma automática, não sendo, portanto, responsabilidade do programador determinar onde esta começa e termina (isto difere do C, por exemplo). Bastará, portanto, aceder ao valor armazenado no bloco de memória indicado pelo apontador, e ser-nos-á devolvido o conteúdo da *ShortString*:

```
VTSTRING :
  WRITELN('SHORTSTRING = ',
  ARGS[I].VSTRING^);
```

O tipo **AnsiString** necessita de ser tratado com um processo ligeiramente diferente. Uma vez que o campo **VAnsiString** é do tipo ***Pointer*** (ou seja, o tipo de dados do valor armazenado no endereço é desconhecido ou tem uma dimensão indefinida), não basta recorrer ao operador **^**. Como foi dito, o compilador trata deste processo automaticamente, e sempre que necessário fornece ao próprio programa ferramentas para gerir estas *strings* de forma autónoma. Portanto, bastará neste caso realizar ***type casting*** – o programa irá aceder à localização na memória onde a *AnsiString* está armazenada, e automaticamente vai processá-la:

```
VTANSISTRING :
  WRITELN('ANSISTRING = ',
  ANSISTRING(ARGS[I].VANSISTRING));
```

Para apontadores, e tendo em conta a quantidade de endereços que actualmente uma memória RAM possui, fazemos *type casting* para o tipo **LongInt** de forma a conhecermos o endereço:

```
VTPOINTER :
  WRITELN('POINTER = ',
  LONGINT(ARGS[I].VPOINTER));
```

Para finalizar, iremos analisar as classes e os objectos. Estes não possuem um valor visto não serem tipos de dados simples. Apesar disso, é possível passar um objecto ou uma classe como argumento num *array of const*. Ambos os tipos de dados **TObject** e **TClass** possuem uma propriedade denominada **ClassName**. Desta forma, o nosso objectivo neste procedimento é determinar o nome da classe ou objecto a que pertence o nosso argumento:

```
VTOBJECT :
  WRITELN('OBJECT = ',
                    ARGS[I].VOBJECT.CLASSNAME);
VTCLASS :
  WRITELN('CLASS REFERENCE = ',
                     ARGS[I].VCLASS.CLASSNAME);
```

O nosso procedimento **Foo** terá então este aspecto:

```
PROCEDURE FOO(ARGS : ARRAY OF CONST);
VAR I : SMALLINT;
BEGIN
  IF LENGTH(ARGS) > 0 THEN
    FOR I:=LOW(ARGS) TO HIGH(ARGS) DO
      CASE ARGS[I].VTYPE OF
        VTINTEGER :
          WRITELN('INTEGER = ',
          ARGS[I].VINTEGER);
        VTBOOLEAN :
          WRITELN('BOOLEAN = ',
          ARGS[I].VBOOLEAN);
        VTCHAR :
          WRITELN('CHAR = ',
          ARGS[I].VCHAR);
        VTEXTENDED :
          WRITELN('EXTENDED = ',
          ARGS[I].VEXTENDED^:0:10);
        VTSTRING :
          WRITELN('SHORTSTRING = ',
          ARGS[I].VSTRING^);
        VTANSISTRING :
          WRITELN('ANSISTRING = ',
          ANSISTRING(ARGS[I].VANSISTRING));
        VTPOINTER :
          WRITELN('POINTER = ',
          LONGINT(ARGS[I].VPOINTER));
        VTOBJECT :
          WRITELN('OBJECT = ',
          ARGS[I].VOBJECT.CLASSNAME);
        VTCLASS :
          WRITELN('CLASS = ',
          ARGS[I].VCLASS.CLASSNAME);
      ELSE
        WRITELN('DESCONHECIDO!');
```

PASCAL – ARRAY DE ARGUMENTOS

```
      END
  ELSE
    WRITELN('SEM ARGUMENTOS.');
  WRITELN;
END;
```

Ensaio com os *arrays* de argumentos

Coloquemos o procedimento **Foo** numa *unit* denominada ***ArgMgr*** (de *Argument Manager*). Para a testar, iremos importar a *unit* ***classes*** para podermos testar a passagem de classes e objectos. Note-se que é necessária a *compiler directive* **{$mode objfpc}** para permitir o seu uso.

```
{$MODE OBJFPC}
PROGRAM ARTIGO45;
USES ARGMGR, CLASSES;

VAR S : STRING = 'VARIAVEL S';
    SL : TSTRINGLIST;

BEGIN
  S := TSTRINGLIST.CREATE;
  FOO([]);
  FOO(['IGOR NUNES', 31]);
  FOO([@FOO, NIL, FALSE, 'K']);
  FOO([S, SL, 3.14]);
  S.FREE;
END.
```

Analisemos o output do programa:

```
Sem argumentos.

AnsiString = Igor Nunes
Integer = 31

Pointer = 471399
Pointer = 0
Boolean = FALSE
Char = K

ShortString = variavel s
Class = TStringList
Extended = 3.1400000000
```

Repare-se que a variável **s**, do tipo *string*, foi considerada uma ***ShortString***. Isto acontece uma vez que, por defeito, o Free Pascal considera que o tipo de dados *string* se refere ao tipo **ShortString**.

Por outro lado, uma *string* escrita directamente no argumento é considerada uma **AnsiString**. Mais uma vez, o Free Pascal assume que uma *string* escrita directamente como argumento é deste tipo, uma vez que, segundo as actuais regras, estas *strings* não têm um tamanho definido e só podem assumir caracteres ASCII por defeito.

Como última referência, é de notar o *output* para a variável **sl**: o procedimento informou-nos que esta é uma classe, mais propriamente a classe **TStringList**.

***printf* – criação de *bindings* para a linguagem C**

```
{$MODE OBJFPC}
PROGRAM ARTIGO45;

// DECLARAÇÃO STANDARD DA FUNÇÃO
PROCEDURE PRINTF(FMT : PCHAR;
          ARGS : ARRAY OF CONST);
CDECL; EXTERNAL 'C';

BEGIN
  PRINTF('%S IGUAL A %D.',
         ['DOBRO DE 3', 6]);
END.
```

O Free Pascal oferece uma utilidade que outros compiladores, incluindo o próprio Delphi, não oferecem do mesmo modo. Com os *arrays of const*, é possível criar facilmente um *binding* com a função **printf** da linguagem C. O seguinte código indica como se deve declarar a função do *standard* da linguagem C, exemplificando de seguida o seu uso:

Concluímos assim a nossa viagem pelos *arrays* de argumentos, comummente designados apenas por *arrays of const*, os quais nos permitem simular procedimentos e funções variádicos. Não é, decerto, uma das ferramentas mais utilizadas pelos programadores de Free Pascal, Object Pascal e Delphi. Todavia, é um instrumento útil que está à disposição, o qual fornece imensa flexibilidade na passagem de argumentos a procedimentos e funções.

AUTOR

Escrito por Igor Nunes

Curioso na área da tecnologia e em especial da programação, tem uma saudável relação com o Object Pascal e é conhecedor das bases de outras linguagens de programação, como Haskell, C, Python e VB.NET. No P@P, é membro da Wiki Team e Moderador Global.

Criar uma aplicação para Android com mapa

Android

Android é um sistema operativo baseado no núcleo do Linux 6 para dispositivos móveis, desenvolvido pela *Open Handset Alliance*, liderada pela Google. Segundo a Google, mais de 1 milhão de utilizadores. Este sistema foi adoptado por várias marcas de topo de mercado HTC, Samsung, Sony, Motorola, LG e Nokia.

O Android está disponível com código aberto desde 21 Outubro de 2008. O Google publicou todo o código sob a licença Apache. No entanto, ele depende de uma autorização da próprio Google para poder aceder á biblioteca de aplicações, Play Store.

Com a API do Google Maps v2 para Android, é possível adicionar mapas com base em dados do Google Maps numa aplicação. A API, que é lida automaticamente com acesso aos servidores do Google Maps, possui métodos que permitem adicionar marcadores, polígonos, sobreposições, e ainda mudar a visão de uma área do mapa.

Ide

O Ide mais utilizado para o desenvolvimento de *android* é o Eclipse. A empresa multinacional de serviços e *software,* Google, disponibiliza uma versão optimizada onde permite o *download* gratuito, em http://developer.android.com/sdk/index.html.

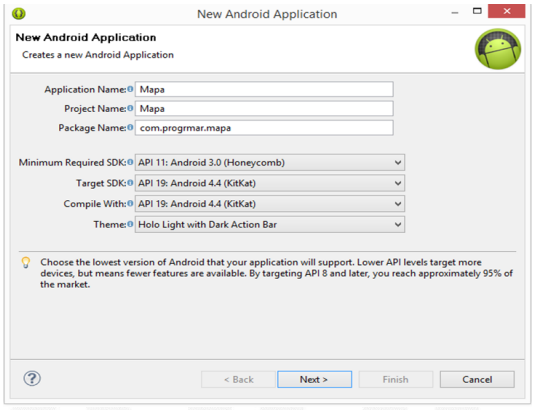
Após descarregar o IDE tem reunidos os elementos necessários para o desenvolvimento de aplicações para o *android*. Com o processo de *download* terminado, irá ter uma pasta de trabalho contendo o nome: adt-bundle-windows-x86_64-20131030 (pode diferir um pouco mediante a versão).

Passos a seguir:

- Abrir a pasta “adt-bundle-windows-x86_64-20131030“ > eclipse > eclipse.exe > ok

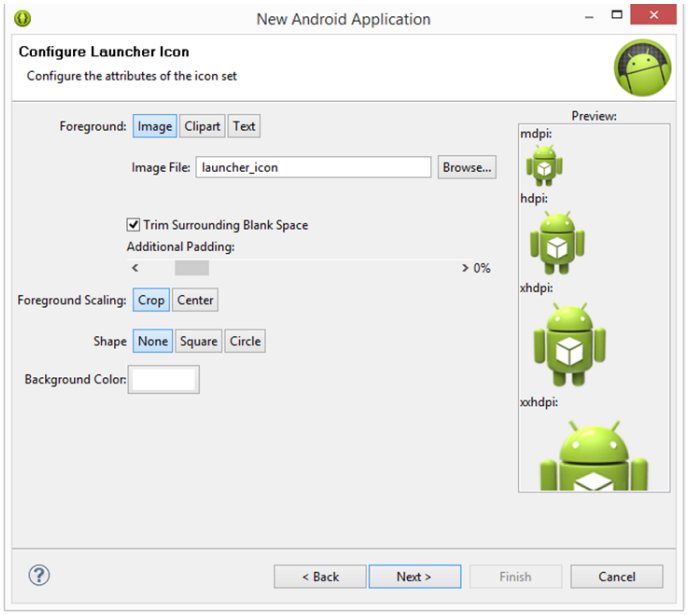
Criar um projecto

- File > New > Android Application Project



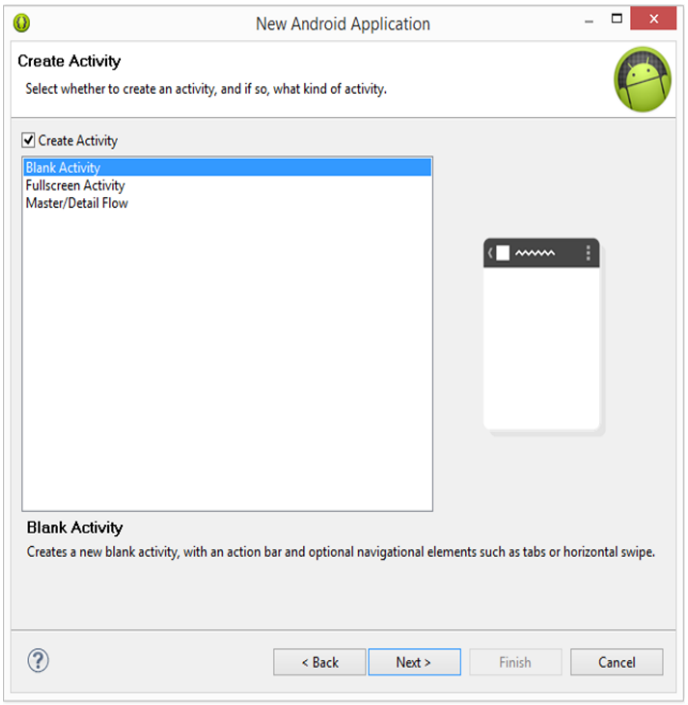
Neste passo é possível dar o nome ao projeto e escolher as versões para que estará disponível a aplicação desenvolvida. Neste caso irá correr em dispostivos com versão 11 até à 19.

- Next

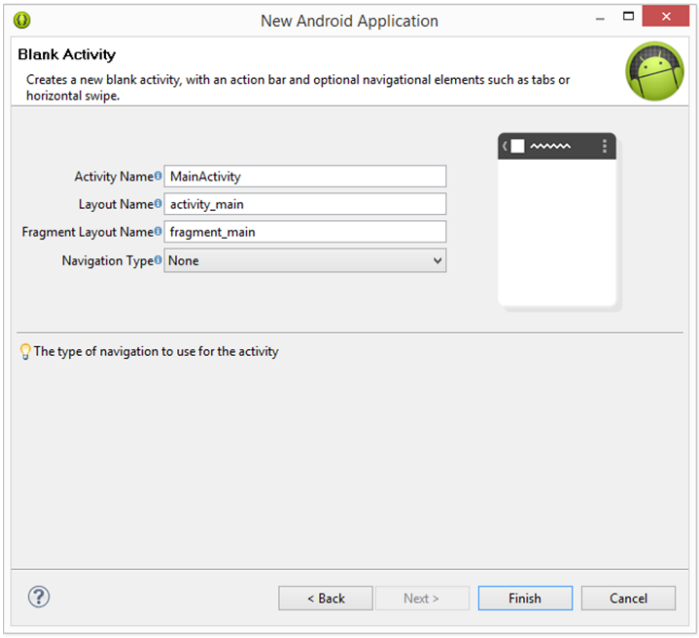


Nesta janela pode escolher o logotipo por defeito ou criar o seu próprio.

- Next



- Next



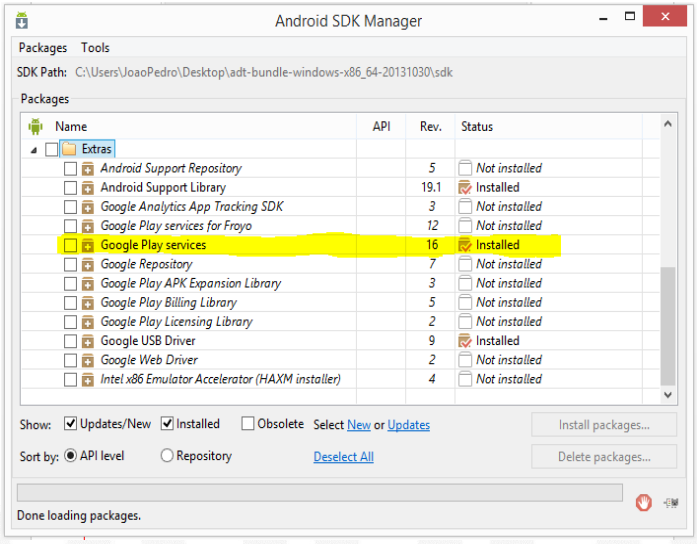
Cada atividade (janela) é composta por três ficheiros e aqui podemos alterar o seu nome se pretendermos.

- Finish

Importar a biblioteca, uma vez que esta permite aceder ao mapa.

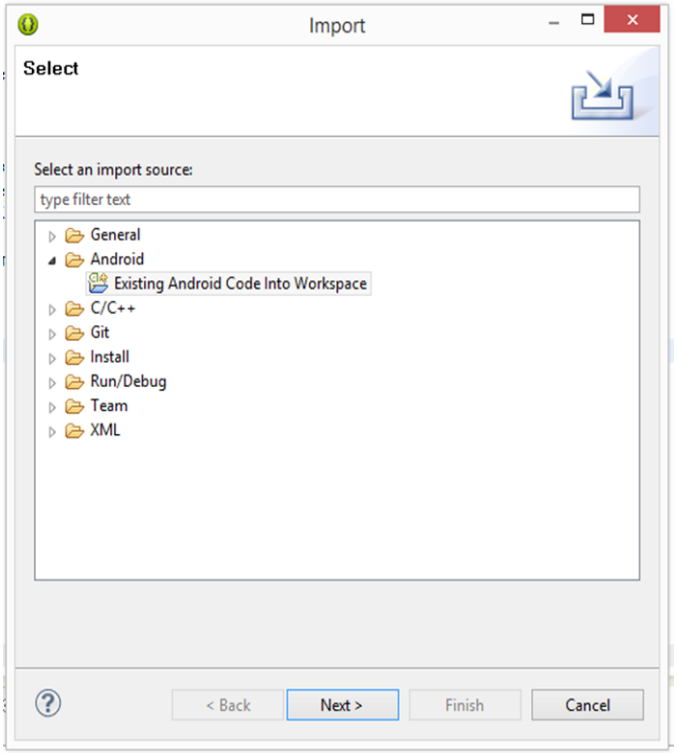
Neste passo iremos importar a biblioteca Google Play Services, ou seja, vamos juntar à nossa pasta de trabalho uma pasta que contem todos os serviços de mapas da Ggoogle. Desta forma, ser-nos-á permitido invocar todas as funcionalidades no nosso projeto.

O projeto está localizado na diretoria adt-bundle-windows-x86_64-20131030 \sdk\extras\google\google_play_services\libproject. Caso não encontre esta diretoria, deve verificar se o Google Play Services no "Android SDK Manager", está instalado, se não estiver apenas tem de selecionar e descarregar.



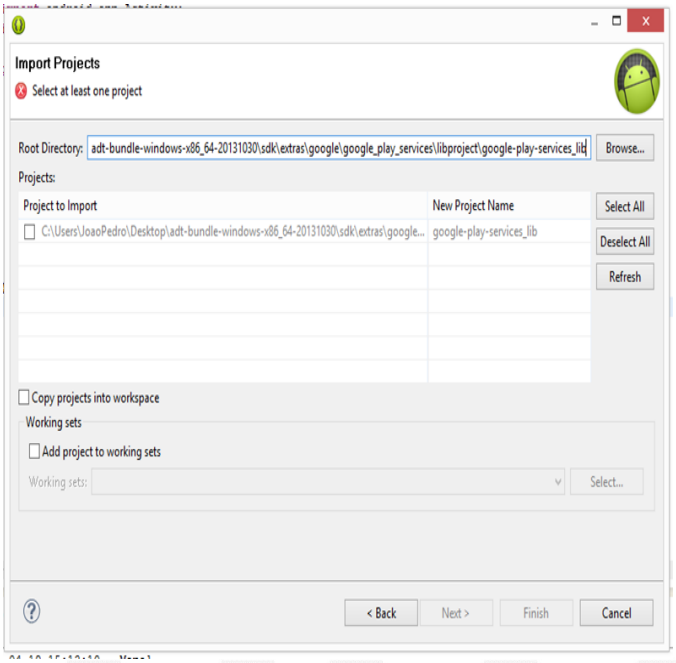
Após garantir que o projeto google_play_services_lib existe, prosseguir da seguinte forma:

- Eclipse > Import > Android > Existing Android Code Into Workspace> Next



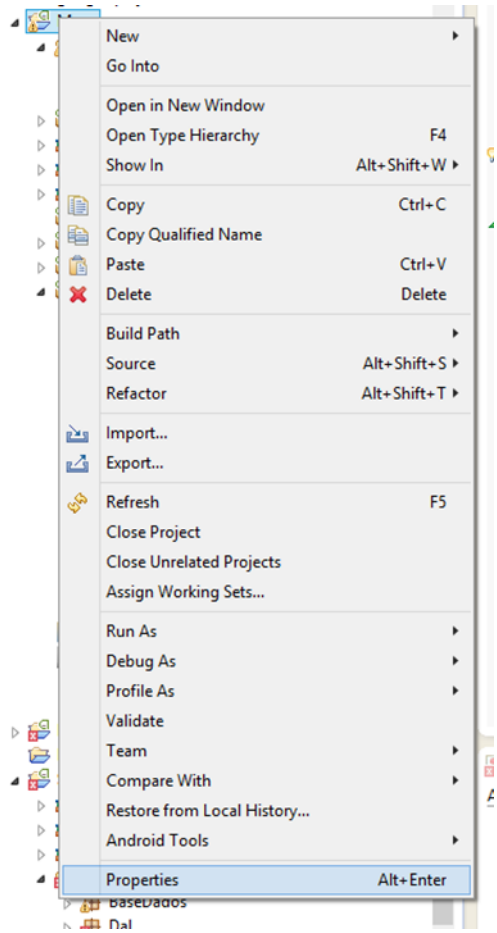
Passos a seguir:

- Browse > ir a pasta de trabalho adt-bundle..XXXXX./ sdk/extras/google/google_play_services/libproject/ google-play-services_lib

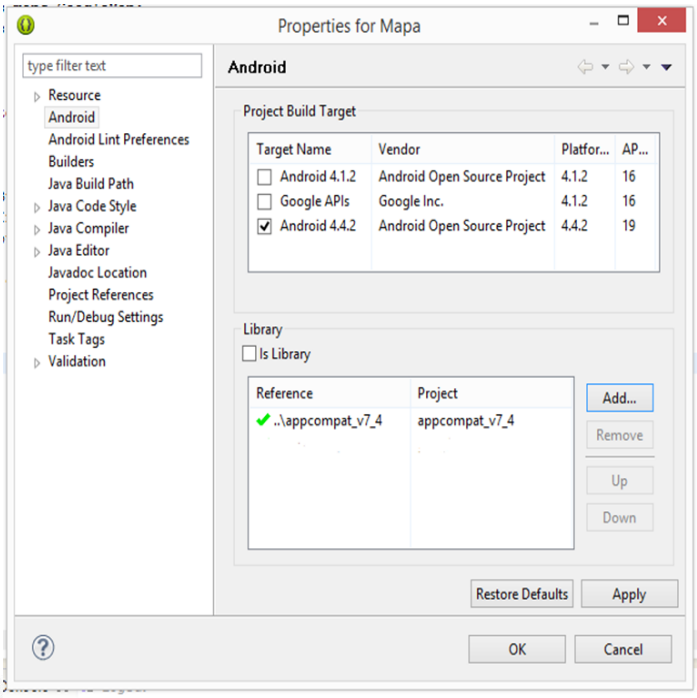


Passo a seguir:

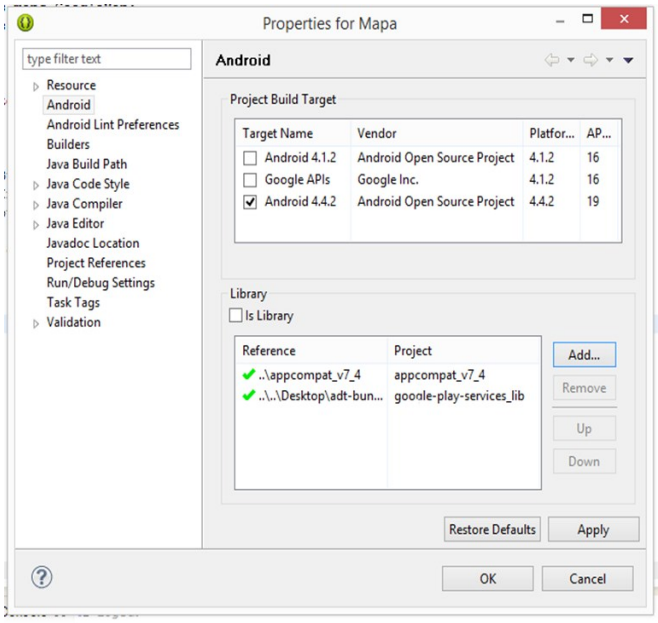
- Mapa > Properties



- Android > Library > add..



- Select Google-play-services_lib > ok



- Ok

Alterar Manifest

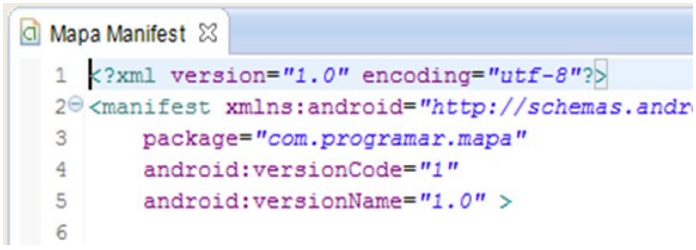
Neste momento o projeto está criado, é necessário configurar o ficheiro AndoridManifest.xml. Este ficheiro tem como objetivo, entre outros, definir as permissões que a aplicação terá. Imaginemos um exemplo prático, os mapas da google permitem saber a nossa localização. Para isso o nosso telemóvel precisa de utilizar o gps ou a internet, e se essas permissões não forem dadas neste ficheiro, essa funcionalidade não está disponível.

Passos a seguir:

- Eclipse > Mapa > bin > AndroidManifest.xml

```
<permission
android:name="<NAMESPACE>
                        .permission.MAPS_RECEIVE"
              android:protectionLevel="signature"/>

<uses-feature android:glEsVersion="0x00020000"
                        android:required="true"/>

<uses-permissionandroid:name="<NAMESPACE>
                      .permission.MAPS_RECEIVE"/>
<uses-permissionandroid:name="android.permission.
                                    INTERNET"/>
<uses-permissionandroid:name="android.permission.
                      WRITE_EXTERNAL_STORAGE"/>
<uses-permissionandroid:name="com.google.android.
       providers.gsf.permission.READ_GSERVICES"/>
<uses-permissionandroid:name=
    "android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission
                      .ACCESS_FINE_LOCATION"/>
<uses-permissionandroid:name="android.permission
                                  .READ_CONTACTS"/>
<uses-permissionandroid:name="android.
                permission.ACCESS_NETWORK_STATE"/>
```

CRIAR UMA APLICAÇÃO PARA ANDROID COM MAPA



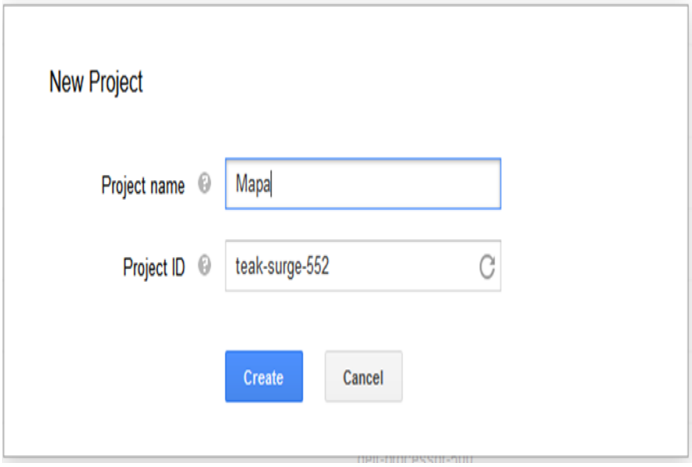
- Logo de seguida inciciar o manifest encontras package="NAMESPACE"
- Copias o teu namesapce
- No AndroidManifest.xml trocar onde tem <NAMESPACE> pelo que acabas de copiar

Obter a API_Key

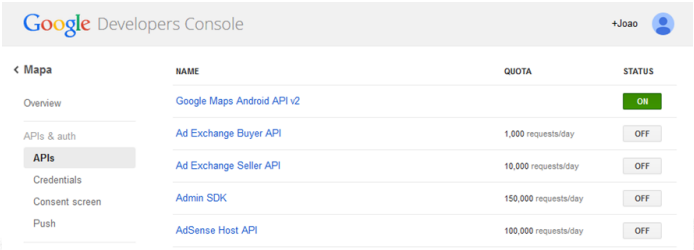
A chave do Google Maps API v2 é baseada num pequeno formulário de certificado digital do aplicativo, conhecido como **SHA-1 fingerprint**. Porque a fingerprint é única, o Google Maps pode usá-la como forma de identificar a sua aplicação. (foste buscar isto a algum lado?)

Existem dois tipos de certificado: o certificado de debug (o que vamos usar e serve apenas para propósito de testes e desenvolvimento) e certificados gerados que nos permitem, por exemplo, colocar a aplicação no Google Play.

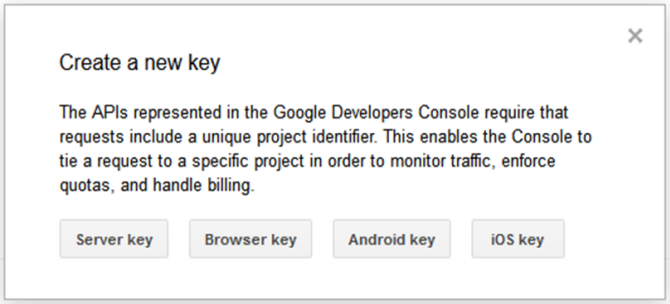
- Ir a este link https://console.developers.google.com/project e, posteriomente, criar um novo projeto no google console.



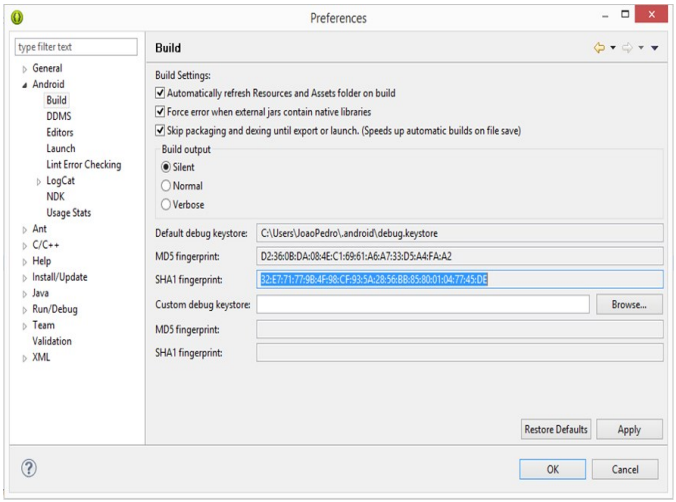
- Abrir o projeto > APIs & auth > APIs > Activar o Google Maps Android API v2



- APIs & auth > Credentials > Create New Key > Android Key
- Passos a seguir:

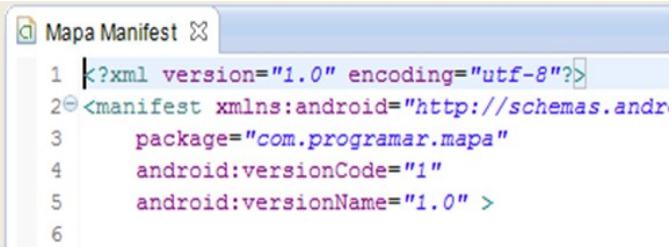


- Voltar ao Eclipse > Window > Preferences >Android > Build
- Copiar a tua SHA1 fingerprint

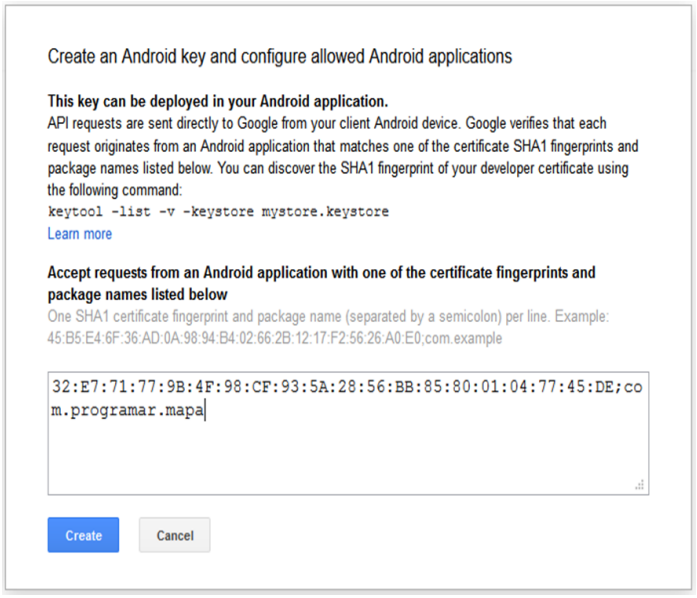


Passos a seguir:

- Colar na janela do google console seguida de ";"
- Buscar o nome completo da aplicação
  - Eclipse Mapa > bin > AndroidManifest.xml



  - Logo de seguida a inciciar o manifest encontras package="NAMESPACE"
  - Copias o teu namesapce
- Retornar ao Google Console e colar o nome da app depois do ";"



Passos a seguir:

- Depois de carregar em criar, a google irá fornecer uma *key*

- Deve substituir no *manifest*

```
<meta-dataandroid:name=
                "com.google.android.gms.version"
                       android:value="@integer/
                 google_play_services_version"/>

<meta-data
android:
      name="com.google.android.maps.v2.API_KEY"
                 android:value="Substituir aqui"/>
```

Atenção isto deverá ser colocado dentro da tag application

Alterar Layout

Em Android, o aspeto gráfico de cada atividade é definido num ficheiro XML.Neste caso, vamos editar o ficheiro activity_main.xml

Passos a seguir:

- Eclipse > Mapa > res > layout >activity_main.xml

activity_main.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <fragment xmlns:android="http://schemas.android.com/apk/res/android"
3   android:id="@+id/map"
4   android:name="com.google.android.gms.maps.MapFragment"
5   android:layout_width="match_parent"
6   android:layout_height="match_parent" />
```

```
<?xmlversion="1.0"encoding="utf-8"?>
<fragmentxmlns:android="http://schemas.android.com/
                                    apk/res/android"
                                android:id="@+id/map"
                                    android:name=
          "com.google.android.gms.maps.MapFragment"
                          android:layout_width="match_parent"
                       android:layout_height="match_parent"/>
```

Alterar a Activity

Uma Activity representa uma interface que nos é apresentada e, além do ficheiro XML que define o aspeto gráfico, existe um ficheiro .JAVA onde é definido o código das funcionalidades.

Neste caso, bastará codificarmos o método onCreate que faz parte do ciclo de vida de uma atividade e é executado sempre que a atividade é mostrada ao utilizador. Neste método o que iremos fazer é associar o layout XML que acima definimos.

Passos a seguir:

- Eclipse > Mapa > src >MainActivity.java

MainActivity.java

```
1 package com.programar.mapa;
2
3 import com.google.android.gms.maps.MapFragment;
4 import android.app.Activity;
5 import android.os.Bundle;
6
7 public class MainActivity extends Activity {
8
9
10    @Override
11    protected void onCreate(Bundle savedInstanceState) {
12        super.onCreate(savedInstanceState);
13        setContentView(R.layout.activity_main);
14
15        ((MapFragment) getFragmentManager().findFragmentById(R.id.map)).getMap();
16    }
17
18 }
19
```

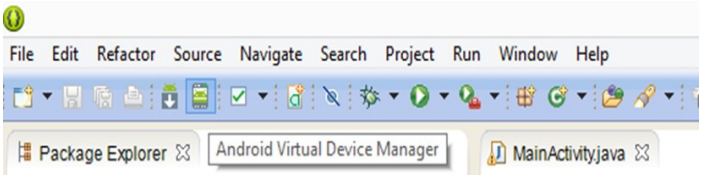
CRIAR UMA APLICAÇÃO PARA ANDROID COM MAPA

```
package com.programar.mapa;

import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.MapFragment;
import android.app.Activity;
import android.os.Bundle;

publicclass MainActivity extends Activity {

@Override
protectedvoid onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

 ((MapFragment) getFragmentManager
          ().findFragmentById(R.id.map)).getMap();
    }

}
```

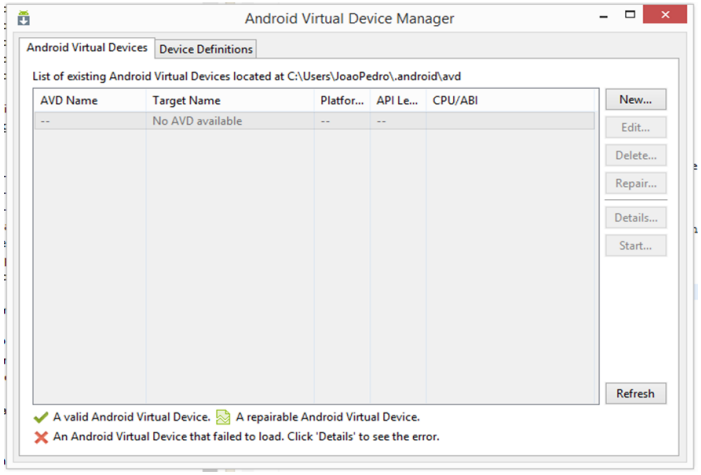
Executar aplicação

A criação do AVD (emulador de dispositivos android) é feita da seguinte forma:

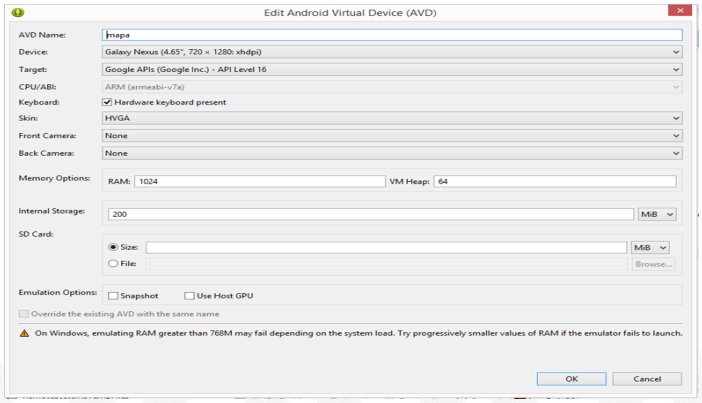
Ir a Window\Android Virtual Device Manager



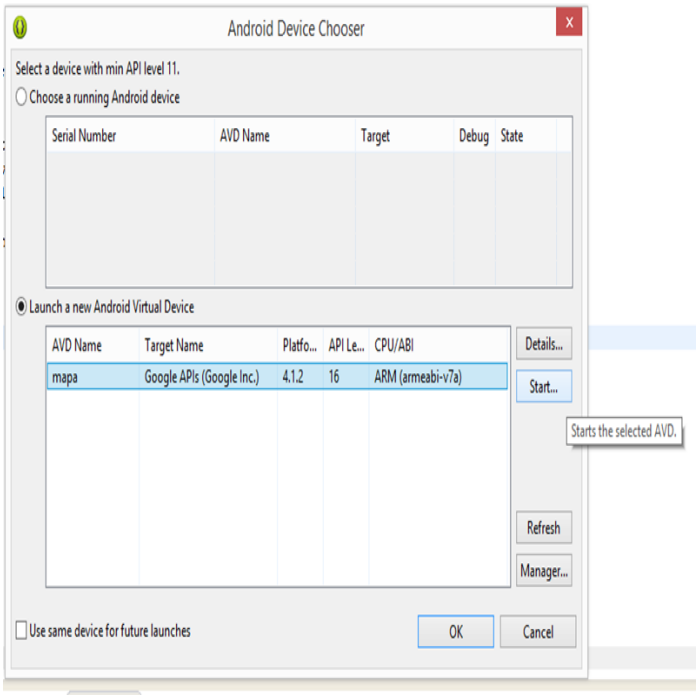
- New



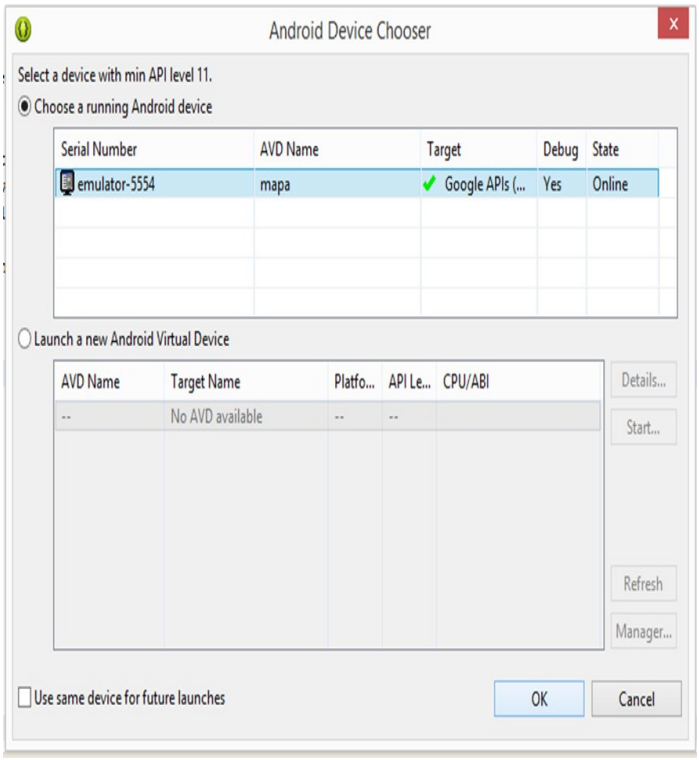
- Preencher segundo a imagem seguinte



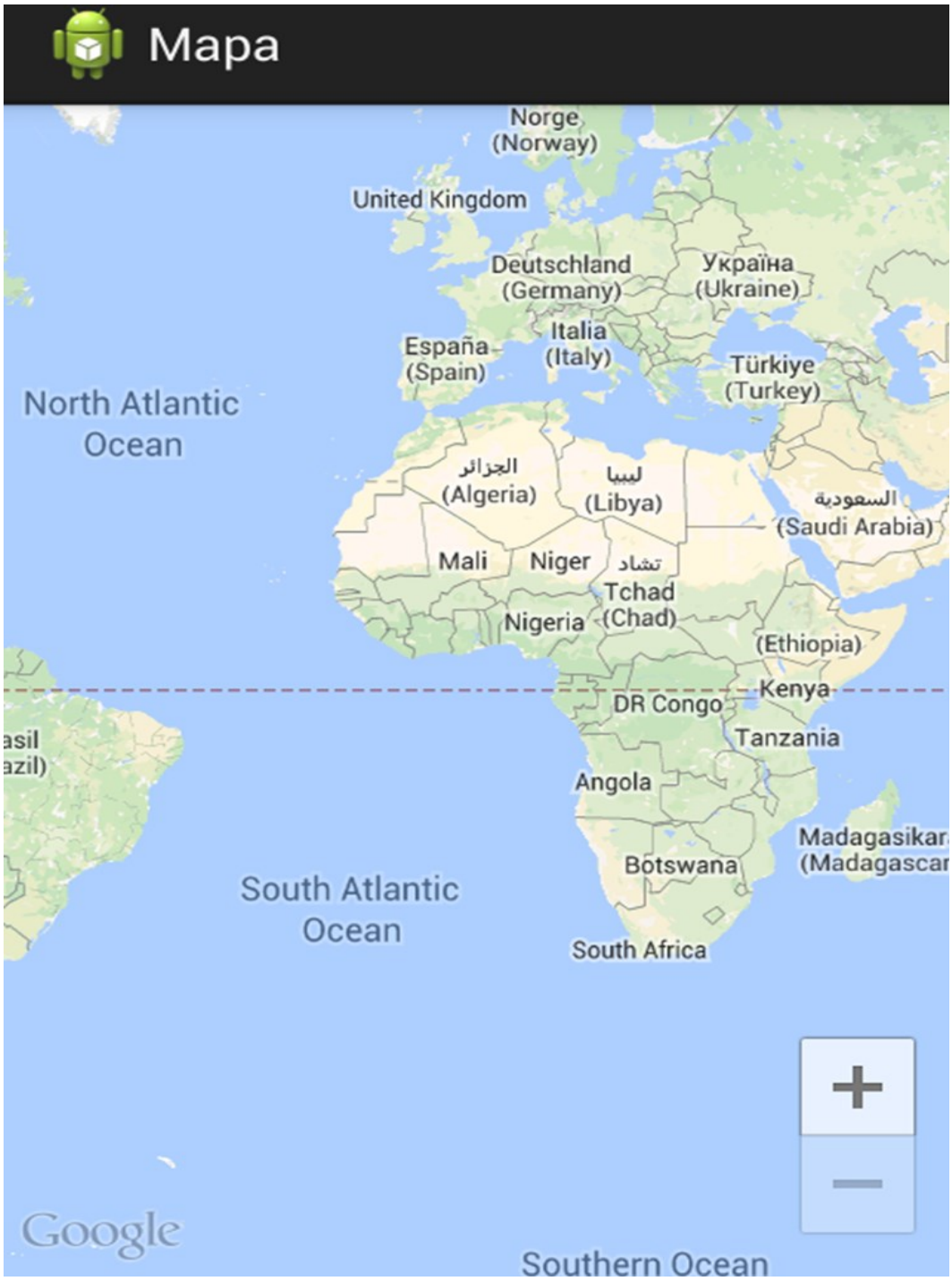
- Ok



- Start Avd
- (Depois do emulador iniciar) Selecione Choose a running Andorid device



- Ok



No final deste tutorial é possível visualizar no emulador ou dispositivo móvel um mapa como este.

AUTOR

Escrito por João Silva

natural de Viana Castelo. Atualmente é aluno do curso de Engenharia Informática no Instituto Politécnico de Viana do Castelo (IPVC). Teve o primeiro contacto com programação para dispositivos móveis nas Unidades Curriculares de Projeto e Computação Móvel do referido curso.

C# CRUD (Create, Read, Update & Delete)

Introdução

A partir deste artigo o leitor terá a possibilidade de aprender a utilizar as quatro operações mais comuns na interação com a base de dados.

O artigo utiliza o sistema de base de dados MySQL mas poderá ser adaptável, sem grande esforço, a outro SGBD.

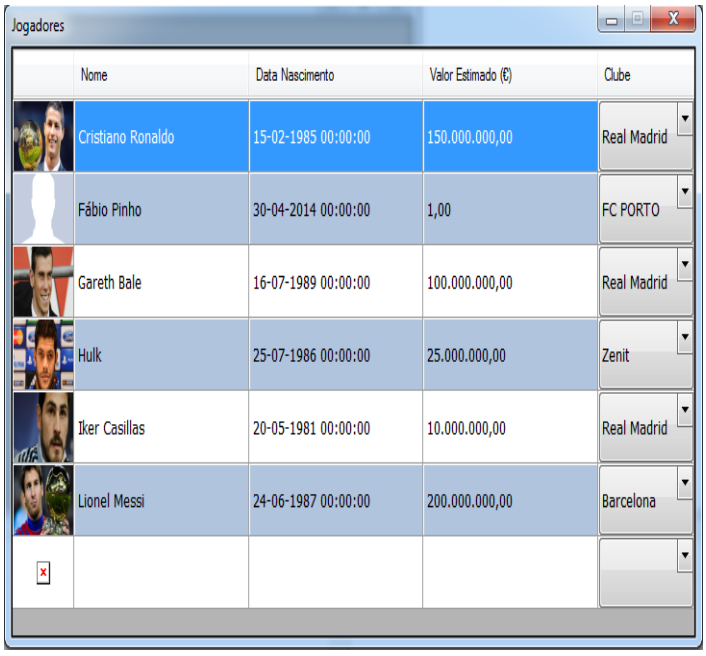
Instalação

Para que o leitor consiga estabelecer uma conexão com o MySQL é necessário:

1. A instalação do MySQL Connector (ver link 1)
2. Adicionar a referência do MySQL ao projecto (ver link 2)
3. Criar uma base de dados com duas tabelas "jogadores" e "clubes" (ver link 3)

Resultado final

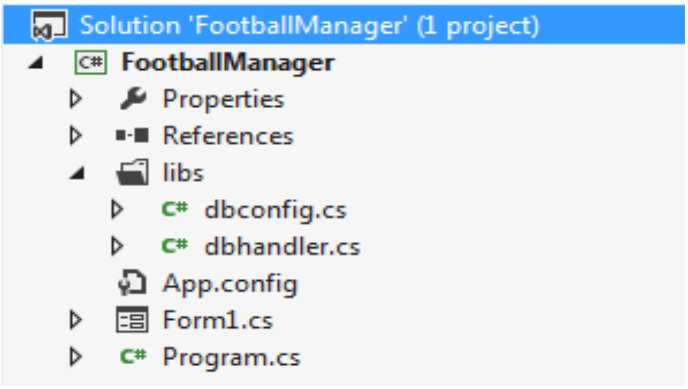
O artigo também exemplifica a utilização do controlo DataGridView em pleno, usufruindo da possibilidade de imagens e listas de itens dentro do mesmo. Todo o código abaixo escrito será utilizado dentro dos eventos deste controlo, pelo que não será necessário a adição de mais nenhum objecto ao formulário.



O leitor poderá fazer download do projecto para fins educacionais (ver link 4).

Estrutura

Uma estrutura de projecto bem delineada poupa-nos imenso tempo. A estrutura deste projecto é muito simples.



Classes

Neste artigo é necessário a criação de duas classes distintas. A primeira classe (*dbconfig*) irá tratar da configuração do acesso à base de dados. A segunda classe (*dbhandler*) irá permitir a realização de *queries* / operações *CRUD*.

```
class dbconfig
    {
        private string server;
        private string user;
        private string password;
        private string database;

        /* Pode-se, desde logo, atribuir valores
                                    por defeito */

        public dbconfig(string __server =
              "localhost", string __user = "root",
                        string __password = "",
            string __database = "footballmanager")
        {
            this.server   = __server;
            this.user     = __user;
            this.password = __password;
            this.database = __database;
        }

        public string get() {
            return "Server=" + this.server +
                   ";Database=" + this.database +
                   ";Uid=" +  this.user + ";Pwd="
                                   + this.password;
            /* Output: Server=localhost;
               Database=footballmanager:Uid=root;
                                          Pwd= */
        }
    }
```

No construtor da classe desde logo foi assumido quais os valores por defeito de ligação à base de dados. No entanto é dada a possibilidade de, aquando a iniciação da classe, definir a configuração desejada.

```
dbconfig db = new dbconfig("localhost", "root", "",
                                "outra_base_dados");
```

Tudo o que o é preciso para estabelecer uma ligação com o servidor de MySQL está nesta classe de configuração.

A classe dbhandler é simplista mas totalmente funcional e segura, em termos de realização de queries, pois permite a passagem de parâmetros. Apenas duas funções foram criadas:

Função ***get***() – Realiza a operação Ler (***Read***), cuja finalidade é devolver informações da base de dados.

Função ***set***() – Realiza as operações de Criar (***Create***), Actualizar (***Update***) e Eliminar (***Delete***).

Para que a utilização desta classe funcione em pleno será necessário adicionar quatro referências.

```
/* Permite a ligação com a base de dados */
using MySql.Data.MySqlClient;
/* Permite a utilização da funcionalidade
[Optional] <- disponível apenas na framework 4.0 */
using System.Runtime.InteropServices;
/* Permitem a chamada das funções directamente */
using System.Data;
using System.Text.RegularExpressions;

class dbhandler
   {
        private dbconfig db;
        private MySqlConnection connection;
        private MySqlCommand command;
        private MySqlDataAdapter dAdapter;
        private DataSet dSet;

       public dbhandler()
       {
         db = new dbconfig();
         connection = new MySqlConnection
                                        (db.get());
       }
    }
```

Sempre que a classe *dbhandler* é iniciada o seu construtor irá chamar o construtor da classe *dbconfig* e por sua vez estabelecer uma ligação com a base de dados MySQL.

```
public DataSet get(string __query, [Optional]
MySqlParameter[] __param)
   {
        this.command = new MySqlCommand(__query,
                                    this.connection);

        if (__param != null)
{ this.command.Parameters.AddRange(__param); }

        this.dAdapter= new MySqlDataAdapter
                                      (this.command);
        this.dSet = new DataSet();

        /* input exemplo: 'SELECT * FROM tabela
           WHERE id = @id' output: 'FROM tabela' */
        Regex rgx = new Regex("FROM\\s\\w+");
        MatchCollection matches = rgx.Matches
                                        (__query);
        /* output: 'tabela' */
        string table = matches[0].Value.Replace
                                      ("FROM ", "");

        this.dAdapter.Fill(this.dSet, table);

        return this.dSet;
   }
```

A utilização de um DataSet é bastante benéfico (em comparação com o DataReader) por várias razões:

- Não é preciso que estabeleçamos uma ligação, pois o método .Fill() já faz isso;
- Podemos de imediato atribuir o DataSource a um controlo (como é o caso da DataGridView);
- É possível fazer queries dentro do DataSet, pois este guarda os dados como se de uma tabela se tratasse.

O leitor poderá verificar que foi incutido nesta função o uso das expressões regulares, que tem como aplicação devolver o nome da tabela da query. Não é obrigatório a utilização deste método, o nome da tabela pode ser passado via parâmetro (para a função) se assim preferir.

```
public int set(string __query, [Optional]
                          MySqlParameter[] __param)
     {
          this.command = new MySqlCommand(__query,
                                    this.connection);

          if (__param != null) {
            command.Parameters.AddRange(__param); }

          if (this.connection.State !=
                          ConnectionState.Open) {
              this.connection.Open();
          }

          return this.command.ExecuteNonQuery();
          // retorna o número de linhas afectadas
     }
```

Sim, a função *set*() contém apenas este trecho de código. Por precaução é retornado o número de linhas afectadas para perceber se foi ou não executado com sucesso.

A nossa classe está concluída. A classe executa eficazmente os métodos estipulados pelo *CRUD* de forma segura e muito simples. Resta agora colocar esta classe em acção.

Formulário

No formulário apenas será necessário adicionar o controlo *DataGridView*. Todas as operações do *CRUD* serão realizadas directamente pelos diversos eventos (*Load*, *userAddedRow*, *RowValidating*, *CellEndEdit*, *onClick*, *MouseDown*).

Referências e Variáveis

Para que tudo funcione em pleno é necessário adicionar a referência do MySQL

```
using MySql.Data.MySqlClient;
```

C# CRUD (CREATE, READ, UPDATE & DELETE)

Bem como adicionar algumas variáveis privadas a este formulário:

```
private libs.dbhandler dbQuery =
                                new libs.dbhandler();
/* Retêm o indíce e ID respectivos a cada jogador
                                    na DataGridView */
private Dictionary<int, int>
        _dicJogadores = new Dictionary<int, int>();
```

Eventos e Funções

No evento *Load* vamos criar toda a estrutura da *DataGridView*, desde os tipos de colunas (*Textbox*, *Image* e *Combobox*), bem como selecionar o conteúdo proveniente da base de dados.

Os dois eventos *userAddedRow* e *RowValidating* tem como finalidade a inserção de informação na base de dados.

O evento *CellEndEdit* terá como objectivo proporcionar a actualização da informação.

E por fim os eventos *onClick* e *MouseDown* vão criar uma estrutura capaz de eliminar informação bem como de actualizar a foto de um jogador, tudo isto através da criação, em *run-time*, de um *ContextMenuStrip* atribuído à *DataGridView*.

Ler dados

O evento *Load* tem a seguinte estrutura:

```
DataSet dSetJogadores = dbQuery.get("SELECT jog.id,
jog.nome, jog.data_nascimento, jog.valor_estimado,
jog.foto, club.descricao, club.id " +
                          "FROM jogadores AS jog " +
"LEFT JOIN clubes AS club ON club.id = jog.id_clube
                       " + "ORDER BY jog.nome ASC");

 /* A tabela clubes é uma tabela secundária que se
                    interliga com a tabela jogadores.
  * Para que a ComboboxColumn contenha todos os
valores desta tabela, é necessário abastecê-la pela
                                  via do DataSource
  */
DataSet dSetClubes = dbQuery.get
              ("SELECT id, descricao FROM clubes");

var nome = new DataGridViewTextBoxColumn(){
         HeaderText = "Nome", MinimumWidth = 150 };
var data_nascimento =
new DataGridViewTextBoxColumn(){
                    HeaderText = "Data Nascimento" };
var valor_estimado =
                     new DataGridViewTextBoxColumn()
               { HeaderText = "Valor Estimado(€)" };

var clube = new DataGridViewComboBoxColumn() {
    HeaderText = "Clube",
    DataSource = dSetClubes.Tables[0],
    ValueMember = "id",
    DisplayMember = "descricao",
    AutoSizeMode =
                DataGridViewAutoSizeColumnMode.None,
    FlatStyle = FlatStyle.Standard
 };

var foto = new DataGridViewImageColumn() {
    HeaderText = "",
    AutoSizeMode =
                DataGridViewAutoSizeColumnMode.None,
    Width = 64
};

this.dataGridView1.Columns.AddRange(foto, nome,
          data_nascimento, valor_estimado, clube);
this.dataGridView1.RowTemplate.MinimumHeight = 50;
```

Como o leitor poderá comprovar, neste trecho de código apenas foram delineadas as queries necessárias à construção da DataGridView, assim como toda a estrutura das colunas. Falta então adicionar dados informativos provenientes da base de dados.

```
for (int i = 0; i <= dSetJogadores.Tables
[0].Rows.Count - 1; i++) {

     string _imgDefault = "no_image.gif";

     Image _img = imageResize(new Bitmap
     (Application.StartupPath + "/imagens/
     jogadores/" + _imgDefault), new Size(64, 64));

     if (dSetJogadores.Tables[0].Rows[i].
          ItemArray[4].ToString() != string.Empty){
               _img = new Bitmap(imageResize(new
        Bitmap(Application.StartupPath + "/imagens/
               jogadores/" +dSetJogadores.Tables[0].
                   Rows[i].ItemArray[4].ToString()),
                                  new Size(64, 64)));
     }

               string _nome = dSetJogadores.
          Tables[0].Rows[i].ItemArray[1].ToString();
               string _datanascimento =
                    dSetJogadores.Tables[0].Rows[i].
                            ItemArray[2].ToString();

              /* O valor, que está em float (10,2),
                       ultrapassa as casas decimais.
               * É então preciso haver uma conversão
                  desse valor para double e, por sua
                  vez, formatar esse valor com duas
                                     casas 'N2'.*/
               string _valorestimado =
          Convert.ToDouble(dSetJogadores.Tables[0].
    Rows[i].ItemArray[3].ToString()).ToString("N2");
                        int _clube = Convert.ToInt32
                   (dSetJogadores.Tables[0].Rows[i].
                            ItemArray[6].ToString());

       /* Guarda a posição do ID na DataGridView */
               _dicJogadores.Add(i, Convert.ToInt32
                   (dSetJogadores.Tables[0].Rows[i].
                           ItemArray[0].ToString()));

               this.dataGridView1.Rows.Add(new
          object[] { _img, _nome, _datanascimento,
                           _valorestimado, _clube });

 }
```

E finalizamos o evento load. Recapitulando o processo: definimos as queries; definimos a estrutura da DataGridView; atribuímos a informação das queries às linhas.

Se o leitor correr a aplicação neste momento, possivelmente dar-lhe-á erro na função imageResize(), pois esta ainda não foi adicionada ao formulário.

```
public static Image imageResize(Image img, Size
                                                size)
{    return (Image)(new Bitmap(img, size));
}
```

Adicionar dados

Para que seja possível a adição de novas linhas directamente na base de dados, precisamos de dois eventos para garantir a fiabilidade dos dados e evitar possíveis erros.

```
private void dataGridView1_UserAddedRow(object
                sender, DataGridViewRowEventArgs e)
{
   _userAddedRowIndex =
              this.dataGridView1.CurrentRow.Index;
}
```

É definido qual o índice de linha actual para ser utilizado no evento *RowValidating*:

```
if(e.RowIndex == _userAddedRowIndex){

     /* Verifica se existe ou não valores nas
                                        células */

     string _nome = (string.IsNullOrEmpty
              (this.dataGridView1.Rows[e.RowIndex].
         Cells[1].Value.ToString()) ? "undefined" :
       this.dataGridView1.Rows[e.RowIndex].Cells[1].
                                  Value.ToString());

    DateTime _datanascimento = (this.dataGridView1.
          Rows[e.RowIndex].Cells[2].Value == null ?
                                       DateTime.Today :
          Convert.ToDateTime(this.dataGridView1.Rows
           [e.RowIndex].Cells[2].Value.ToString()));

     double _valorestimado =
              (this.dataGridView1.Rows[e.RowIndex].
                    Cells[3].Value == null ? 0.0f :
           Convert.ToDouble(this.dataGridView1.Rows
          [e.RowIndex].Cells[3].Value.ToString()));

     int _id_clube = (this.dataGridView1.Rows
          [e.RowIndex].Cells[4].Value == null ? 1 :
            Convert.ToInt32(this.dataGridView1.Rows
          [e.RowIndex].Cells[4].Value.ToString()));

     MySqlParameter[] _sqlParams =
                             new MySqlParameter[] {
        new MySqlParameter("@nome", _nome),
        new MySqlParameter
              ("@datanascimento",_datanascimento),
        new MySqlParameter("@valorestimado",
                                  _valorestimado),
        new MySqlParameter("@id_clube", _id_clube)
     };

     if (dbQuery.set("INSERT INTO jogadores (nome,
     data_nascimento, valor_estimado, id_clube) " +
                  "VALUES (@nome, @datanascimento,
    @valorestimado, @id_clube)", _sqlParams) > 0) {

          // É necessário actualizar os indíces do
          //dicionário
          int _lastid = Convert.ToInt32(dbQuery.get
("SELECT id FROM jogadores ORDER BY id DESC LIMIT
    1").Tables[0].Rows[0].ItemArray[0].ToString());

          _dicJogadores.Add(e.RowIndex, _lastid);

      }
}
```

O leitor decerto notará que nesta inserção não está incutida a coluna foto, por razões estruturais. A inserção da foto será possível após a conclusão da inserção dos dados na base de dados.

Actualizar dados

O trecho de código de actualização de dados é praticamente idêntico ao da inserção de dados, pelo que o leitor não notará grande diferença. É apenas estabelecida uma condição para prevenir enganos.

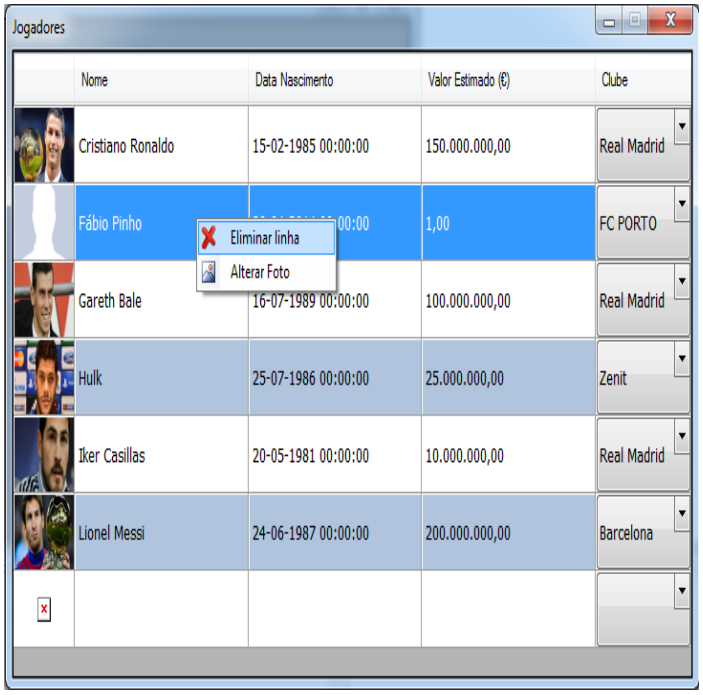
Portanto, no evento *CellEndEdit* (que ocorre após se clicar fora da célula), temos o seguinte código:

```
if (e.RowIndex != _userAddedRowIndex)

{

/* É necessário verificar se não se trata de uma
                                   adição de linha */

     string _nome = (string.IsNullOrEmpty
              (this.dataGridView1.Rows[e.RowIndex].
         Cells[1].Value.ToString()) ? "undefined" :
               this.dataGridView1.Rows[e.RowIndex].
                         Cells[1].Value.ToString());

     DateTime _datanascimento =
              (this.dataGridView1.Rows[e.RowIndex].
          Cells[2].Value == null ? DateTime.Today :
         Convert.ToDateTime(this.dataGridView1.Rows
          [e.RowIndex].Cells[2].Value.ToString()));

     double _valorestimado = (this.dataGridView1.
   Rows[e.RowIndex].Cells[3].Value == null ? 0.0f :
          Convert.ToDouble(this.dataGridView1.Rows
          [e.RowIndex].Cells[3].Value.ToString()));

     int _id_clube = (this.dataGridView1.Rows
          [e.RowIndex].Cells[4].Value == null ? 1 :
            Convert.ToInt32(this.dataGridView1.Rows
          [e.RowIndex].Cells[4].Value.ToString()));

     MySqlParameter[] _sqlParams =
                             new MySqlParameter[] {
        new MySqlParameter("@nome", _nome),

        new MySqlParameter
              ("@datanascimento",_datanascimento),

        new MySqlParameter("@valorestimado",
                _valorestimado), new MySqlParameter
                          ("@id_clube", _id_clube),

        new MySqlParameter("@id", _dicJogadores
                                       [e.RowIndex])

     };


          dbQuery.set("UPDATE jogadores SET nome =
          @nome, data_nascimento = @datanascimento,
        valor_estimado = @valorestimado, id_clube =
      @id_clube  " + "WHERE id = @id", _sqlParams);

}
```

Não é novamente atribuído nenhum valor para a foto pois esta não foi selecionada em nenhuma parte. Será então na criação de um *ContextMenuStrip* que iremos providenciar a eliminação de dados bem como a adição/alteração da foto.

A PROGRAMAR

C# CRUD (CREATE, READ, UPDATE & DELETE)



Eliminação dados

Para que tudo funcione na perfeição, vamos precisar de criar dois métodos de *click* (Eliminar linha e Alterar foto) e utilizar o evento *MouseDown* que a *DataGridView* nos fornece para criar o *ContextMenuStrip*.

No evento *MouseDown* é colocado o seguinte código:

```
if (e.Button ==
         System.Windows.Forms.MouseButtons.Right) {
              var _hit = this.dataGridView1.HitTest
                                          (e.X, e.Y);

      this.dataGridView1.ClearSelection();
      this.dataGridView1.Rows
                  [_hit.RowIndex].Selected = true;
      this.dataGridView1.ContextMenuStrip =
      null; // Limpa os ContextMenuStrip existentes

      /* O 'ContextMenuStrip' só deve ser mostrado
                             nas linhas existentes */
      if (this.dataGridView1.Rows
               [_hit.RowIndex].IsNewRow == false) {
            ContextMenuStrip _menu =
                              new ContextMenuStrip();

            ToolStripMenuItem _eliminarLinha = new
ToolStripMenuItem("Eliminar linha", Image.FromFile
(Application.StartupPath + "/imagens/delete.png"));
            ToolStripMenuItem _alterarFoto = new
ToolStripMenuItem("Alterar Foto", Image.FromFile
(Application.StartupPath + "/imagens/
                                    pick_image.gif"));

          _menu.Items.AddRange(new ToolStripItem[]
                { _eliminarLinha, _alterarFoto });

       this.dataGridView1.ContextMenuStrip = _menu;

            /* É enviado por parâmetro o indíce da
                                        linha actual */
            _eliminarLinha.Click += delegate
(object _sender, EventArgs _e) {
  eliminarLinha_click(sender, e, _hit.RowIndex); };
            _alterarFoto.Click += delegate(object
 _sender, EventArgs _e) { alterarFoto_click(sender,
                                  e, _hit.RowIndex); };

      }
}
```

O código acima representa a criação do controlo *ContextMenuStrip* apenas quando o botão direito do rato for pressionado. Verifica se não se trata de uma nova linha, pois caso se trate não faz sentido mostrar este controlo e por fim cria dois métodos capazes de tratar da informação da forma que desejarmos.

```
private void eliminarLinha_click(object sender,
EventArgs e, int index)
{
     int id = _dicJogadores[index];

     MySqlParameter[] _sqlParams =
                              new MySqlParameter[] {
         new MySqlParameter("@id", _dicJogadores
                                              [index])
     };

     if (dbQuery.set("DELETE FROM jogadores WHERE
                      id = @id", _sqlParams) > 0) {
          this.dataGridView1.Rows.RemoveAt(index);
          _dicJogadores.Remove(index);
     }
}
```

Sempre que elimina uma linha da base de dados é necessário também eliminar essa mesma linha do dicionário de jogadores de forma a manter ambas as estruturas de dados correctas.

```
private void alterarFoto_click(object sender, Even-
tArgs e, int index) {

    var _opf = new OpenFileDialog(){
        AddExtension = false,
        Multiselect = false,
        Filter = "Image files (*.jpg, *.jpeg,
      *.gif, *.png) | *.jpg; *.jpeg; *.gif; *.png",
        Title = "Escolha a imagem"
    };

     if (_opf.ShowDialog() ==
           System.Windows.Forms.DialogResult.OK) {

          int id = _dicJogadores[index];

          MySqlParameter[] _sqlParams =
                              new MySqlParameter[] {
            new MySqlParameter("@id", _dicJogadores
          [index]), new MySqlParameter("@foto", id +
                "/" + _opf.SafeFileName.ToString())
            /* O caminho correcto a inserir na base
             de dados: id/nome_ficheiro.extensão */
          };

          if (dbQuery.set("UPDATE jogadores SET foto
           = @foto WHERE id = @id", _sqlParams) > 0)
            {
               string _path =
    Application.StartupPath + "/imagens/jogadores/"
                                          + id + "/";
               System.IO.Directory.CreateDirectory
```

```
                                                    (_path);
                // Caso não exista, irá criar a pasta

                  /* Copia o ficheiro seleccionado
                             para a pasta do jogador */
                  System.IO.File.Copy(_opf.FileName,
                     _path + _opf.SafeFileName, true);
               this.dataGridView1.Rows[index].Cell[0].
                        Value = new Bitmap(imageResize
               (new Bitmap(_path + _opf.SafeFileName),
                                    new Size(64, 64)));

            }
        }
}
```

Por fim, não se permite a selecção de ficheiros que não preencham os requisitos do filtro; actualiza-se a informação na base de dados; envia-se a foto selecionada para a pasta do jogador; actualiza-se a célula da foto da *DataGridView*.

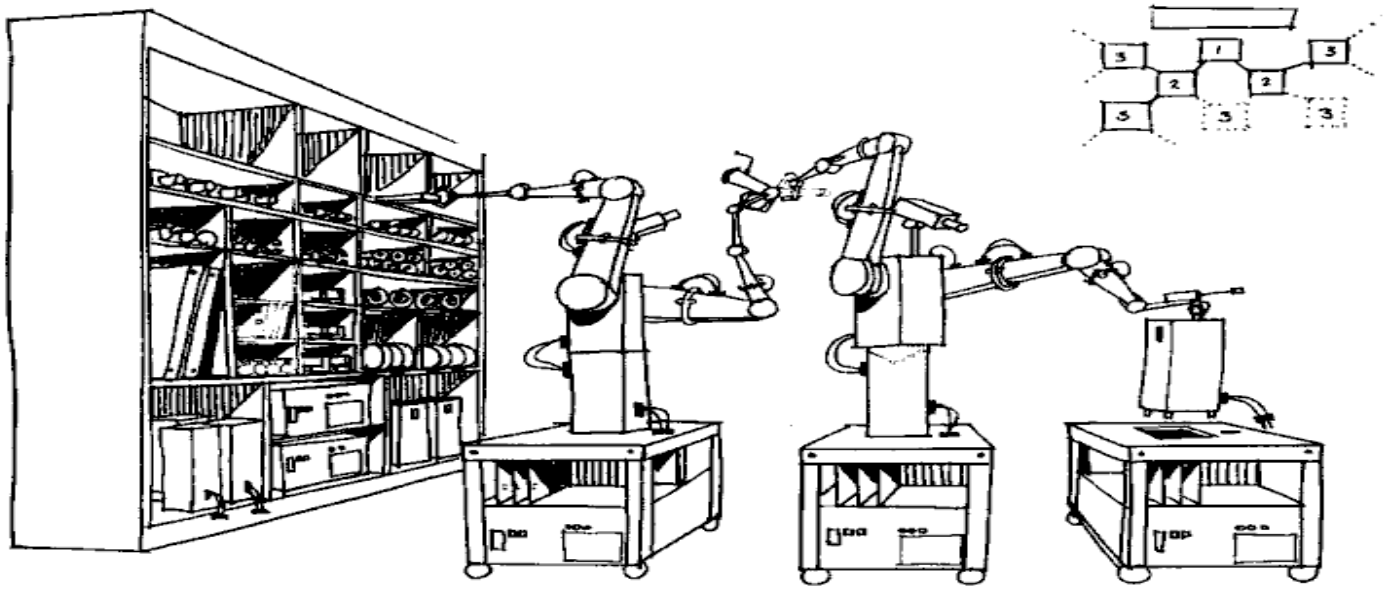
Conclusão

Neste artigo construímos uma estrutura vantajosa em termos de utilização de classes, seguindo um modelo mais orientado a objectos. Definimos também um modelo bastante reduzido em termos de código, mantendo mesmo assim o melhor da segurança e rapidez.

Links de referência

- MySQL Connector - http://tinyurl.com/2r3t2a
- MySQL Referência - http://tinyurl.com/lcdqoao
- Estrutura Base Dados - http://pastebin.com/jPFXtXhE
- Download do projecto - http://tinyurl.com/k2cegr7

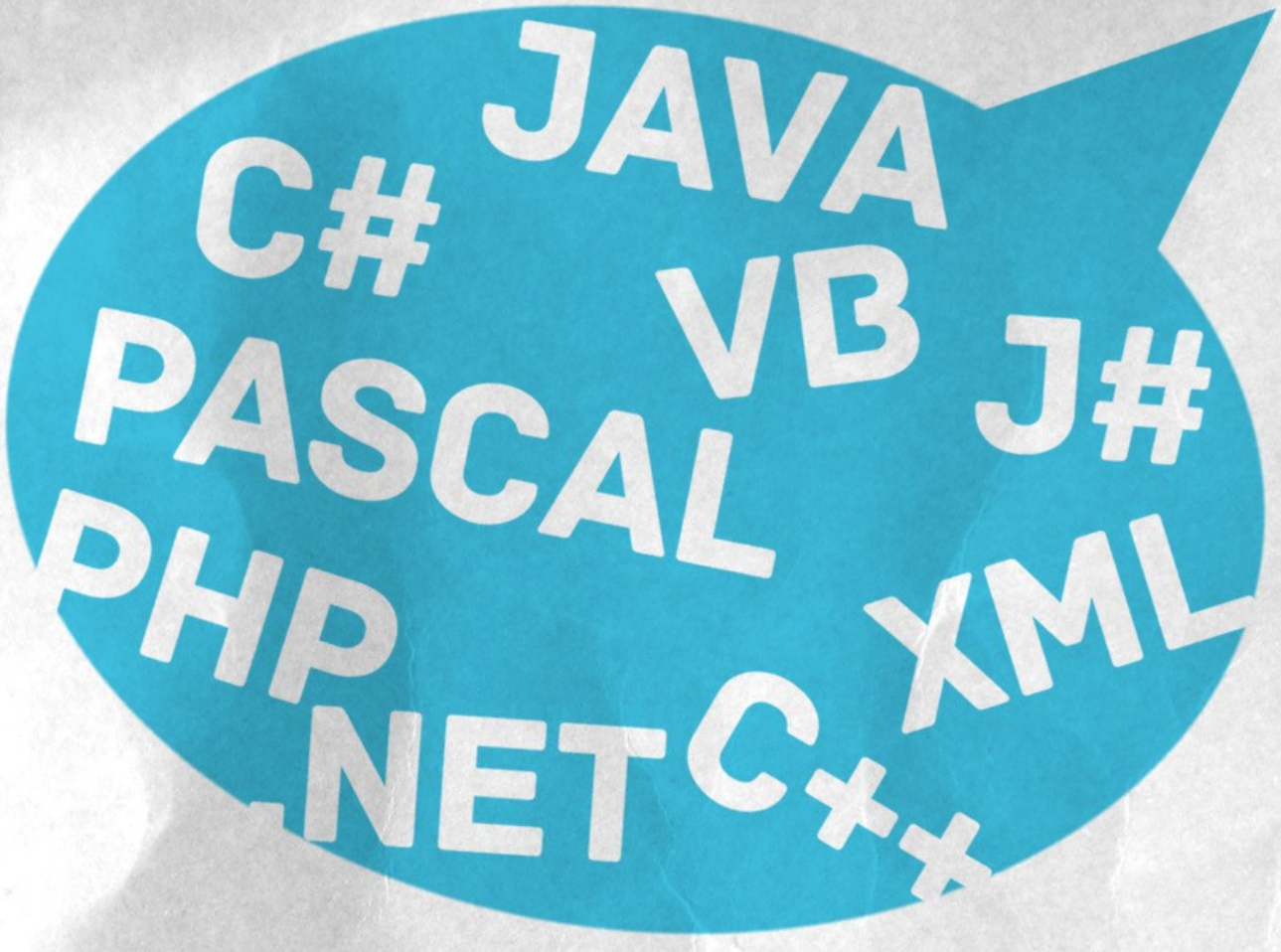
> ***"A primeira classe (dbconfig) irá tratar da configuração do acesso à base de dados. A segunda classe (dbhandler) irá permitir a realização de queries / operações CRUD."***



AUTOR

Escrito por Fábio Pinho

Programador entusiasta nas mais diversas linguagens, sendo PHP, .NET e Java (Android) as suas preferências.
site pessoal: http://stuffpinho.com



ENTÃO, SÓ FALAS EM CÓDIGO?

TEMOS O REMÉDIO CERTO PARA TI!

portugal-a-programar.pt

A MAIOR COMUNIDADE PORTUGUESA DE PROGRAMAÇÃO, APARECE!

COLUNAS

C# - Novas Funcionalidades Do C# 6.0 – Antevisão De Abril De 2014

Visual(NOT)Basic - Métodos de extensão – o que preciso, como quero

Core Dump [12] - Programador ^ Gestor

C# - NOVAS FUNCIONALIDADES DO C# 6.0 – ANTEVISÃO DE ABRIL DE 2014

Introdução

No passado evento **//build/**, a Microsoft disponibilizou uma versão de antevisão da versão 6.0 da linguagem de programação C#.

Trata-se de uma versão preliminar, pelo que algumas funcionalidades poderão sofre alterações, não fazer parte da versão final ou novas funcionalidades poderão ainda ser adicionadas.

Esta nova versão da linguagem inaugura também uma nova postura da Microsoft no que diz respeito ao código aberto e participação da comunidade.

De facto, em direto numa sessão de abertura (*keynote* do segundo dia, aproximadamente a 01:15:50), Anders Hejlsberg publicou no CodePlex código fonte da plataforma de compiladores e os compiladores de C# e Visual Basic – projeto Roslyn.

Além do código fonte, foram publicadas as notas das reuniões de desenho, onde se pode consultar o racional por detrás de cada nova funcionalidade.

Estado Da Implementação Das Funcionalidades

O estado atualizado da implementação das funcionalidades de ambas as linguagens pode ser consultado aqui.

Funcionalidades Implementadas

Comecemos por percorrer, sem qualquer ordem especial, as funcionalidades já implementadas e disponibilizadas na versão de antevisão anteriormente referida.

Inicialização De Auto-Propriedades

Passa a ser possível adicionar inicializadores a auto□propriedades (*auto-property*) da mesma forma que se pode adicionar a campos (*field*):

```
public class Cliente
{
    public string Nome { get; set; } = "José";
    public string Apelido { get; set; } =
                                        "Silva";
}
```

O inicializador inicializa diretamente o campo que dá suporte à propriedade. Por essa razão passa a fazer sentido existirem auto-propriedades sem assessores de escrita (setter):

```
public class Cliente
{
    public string Nome { get; } = "José";
    public string Apelido { get; } = "Silva";
}
```

Auto-propriedades apenas de leitura só são permitidas se existir um inicializador. Caso contrário nunca poderiam conter qualquer valor para além do valor por omissão do seu tipo.

Tal como acontece com os inicializadores dos campos, os inicializadores de auto-propriedades não podem referenciar this – afinal são executados antes do objeto estar devidamente inicializado. Isto faria com que não houvesse muitas escolhas interessantes para inicializar as auto-propriedades. Especialmente as auto-propriedades apenas de leitura pareceriam inúteis se não pudessem, de alguma forma, ser inicializadas a partir de valores passados no construtor do objeto. No entanto, os construtores primários veem mudar isto.

Construtores Primários

Os construtores primários (*primary constructors*) permitem que os parâmetros do construtor sejam declarados diretamente na classe (*class*) ou estrutura (*struct*) sem a necessidade da declaração explícita do construtor no corpo da declaração do tipo. Estes parâmetros estão disponíveis como nomes simples sendo o seu âmbito (*scope*) toda a declaração do tipo.

Nota: O desenho desta funcionalidade foi alterado e segue-se a descrição do novo desenho. No entanto, não foi possível ter estas alterações na versão de antevisão, pelo que, poderão ser encontradas situações que correspondem à semântica anterior: os parâmetros são capturados implicitamente e não existe sintaxe para a captura explícita.

Parâmetros Em Classes E Estruturas

Aqui está um exemplo de uma classe com um construtor primário:

```
public class Cliente(string nome, string apelido)
{
    public string Nome { get; } = nome;
    public string Apelido { get; } = apelido;
}
```

Obter o mesmo efeito sem construtores primários teria requerido campos privados para guardar os valores de nome e apelido, um construtor explícito para os inicializar e um corpo de leitura para Nome e Apelido para os expor:

```
public class Cliente
{
    private string nome;
    private string apelido;
    public Cliente(string nome, string apelido)
    {
        this.nome = nome;
        this.apelido = apelido;
    }
    public string First { get { return nome; } }
    public string Last { get { return apelido; } }
}
```

No extrato de código anterior foram evidenciados os pedaços que se tornam desnecessários com a introdução dos cons-

trutores primários e auto-propriedades apenas de leitura inicializadas.

O objetivo desta funcionalidade é expressar os tipos de forma mais concisa. Note-se, no entanto, que também remove uma diferença importante na linguagem entre tipos mutáveis e tipos imutáveis: as auto-propriedades eram uma forma abreviada apenas disponível se se estivesse na disposição de tornar a classe mutável e, por isso, a tentação para o fazer por omissão era enorme. Agora, com auto-propriedades apenas de leitura, mutável e imutável passam a estar em pé de igualdade.

De momento não existe uma forma de explicitar um corpo para o construtor primário. O comité não conseguiu chegar a uma boa sintaxe e pensa que não será muito importante porque a maioria dos corpos de construtores são constituídos apenas por inicializações. No entanto, muitos são os que se têm manifestado em favor da sua existência. Qual a tua opinião?

Parâmetros Campo

Nota: Os campos parâmetro (*field parameters*) não funcionam na corrente versão de antevisão.

Por omissão, os parâmetros de um construtor primário existem apenas em tempo de inicialização. Membros como as propriedades e métodos não se podem a eles referir porque, quando são chamados, o objeto já foi construído e o parâmetro desapareceu.

Foi considerado (e experimentado) deixar os parâmetros serem implicitamente "capturados" em campos privados gerados pelo compilador se fossem usados depois da construção, mas isso levava a código supérfluo e sujeito a erros onde muitas vezes os parâmetros eram inadvertidamente capturados em campos.

Claro que se podem declarar campos privados e inicializa-los com um parâmetro:

```
public class Cliente(
    string nome,
    string apelido,
    DateTime aniversário)
{
    public string Nome { get; } = nome;
    public string Apelido { get; } = apelido;
    private DateTime _aniversário = aniversário;
}
```

Infelizmente o campo privado não poderia usar o mesmo nome do parâmetro e necessita de um nome alternativo, por exemplo prefixá-lo com uma barra inferior.

Para evitar isto, existe uma sintaxe para capturar explicitamente em campos os parâmetros dos construtores primários

```
public class Cliente(
    string nome,
    string apelido,
    private  DateTime aniversário)
{
    public string Nome { get; } = nome;
    public string Apelido { get; } = apelido;
}
```

Quando se usa o modificador de acessibilidade (o mais provável é ser private) num construtor primário isso significa a existência de um parâmetro e um campo com o mesmo nome e a inicialização do campo com o parâmetro. Desta forma, vai ser possível referir-se a esse campo em métodos e propriedades após a inicialização:

```
public int Idade { get { return CalcularIdade
                                (aniversário); } }
```

Construtores Explícitos

Uma declaração de classe com construtor primário continua a poder definir outros construtores. No entanto, para assegurar que os argumentos são mesmo passados ao construtor primário, todos os construtores devem chamar um inicializador **this(…):**

```
public Ponto()
    : this(0, 0) // chama o construtor primário
{
}
```

Construtores explícitos podem chamar-se entre si, mas direta ou indiretamente terão de chamar o construtor primário no final porque é o único construtor que pode chamar o inicializador base(…).

No caso de estruturas (*struct*) com construtores primários, os construtores explícitos não podem chamar o construtor por omissão (sem parâmetros): têm de chamar direta ou indiretamente o construtor primário.

Incializador Base

O construtor primário chama sempre, implícita ou explicitamente, o inicializador base. Se não for especificado um inicializador base, como com todos os construtores, será chamado o construtor base sem parâmetros.

A forma de explicitamente chamar o inicializador base é passando uma lista de argumentos na especificação da classe base:

```
class BufferFullException()
    : Exception("Buffer full")
{
}
```

Tipos Parciais

Se um tipo for declarado em múltiplas partes, apenas uma das partes pode declarar parâmetros de construtor primário e apenas essa parte pode declarar argumentos na especificação da classe base.

Using Static

Nota: a implementação corrente desta funcionalidade não reflete o desenho atual. Mais detalhes abaixo.

Esta funcionalidade permite especificar um tipo numa cláusula **using**, tornando todos os membros estáticos acessíveis desse tipo sem qualificação no código subsequente:

```
using System.Console;
using System.Math;
class Program
{

  static void Main()
    {
        WriteLine(Sqrt(3 * 3 + 4 * 4));
    }
}
```

Esta funcionalidade tem a capacidade de tornar o código mais curto. Existe alguma preocupação de que possa causar alguma confusão no *namespace* e causar ambiguidades, especialmente em classes que não foram desenhadas para serem "abertas" desta forma. Com quantos métodos Create (…) pode uma pessoa lidar se ficar confusa com o que está a criar?

Métodos De Estensão

Os métodos de extensão são métodos estáticos. Isto levanta a questão de como uma diretiva *using static* os vai expor. Como um método de topo no escopo? Como um método de extensão? Ambos? Vejamos um exemplo:

```
using System.Linq.Enumerable; // Apenas o tipo, e
//não todo o namespace

class Program

{
    static void Main()
    {
      var range = Range (5, 17);// (1)
        var odd = Where
                  (range, i => i % 2 == 1); // (2)
        var even = range.Where
                        (i => i % 2 == 0); // (3)
    }
}
```

Range é um normal método estático, portanto a chamada marcada com (1) pode e deve ser permitida. A implementação atual não faz nada de especial com métodos de extensão, portanto permite a chamada a Where como método estático em (2), mas não como método de extensão em (3).

Isto não é o que é pretendido. Permitir (3) satisfaria o longo pedido para tornar disponíveis métodos de extensão mais seletivamente pelo tipo em que estão definidos e não apenas pelo *namespace*. Por outro lado, os métodos de extensão raramente foram desenhados para serem usados como métodos estáticos (é apenas um situação de recurso para os raros casos em que existem ambiguidades) e não há interesse em venham a causar confusão no *namespace* de topo. Por isso, numa futura versão, (2) não será permitido.

Expressões De Declaração

As expressões de declaração permitem declarar variáveis locais no meio de expressões, com ou sem inicialização. Aqui estão alguns exemplos:

```
if (int.TryParse(s, out int i)) { … }
GetCoordinates(out var x, out var y);
                Console.WriteLine("Result: {0}",
                        (int x = GetValue()) * x);

if ((string s = o as string) != null) { … s … }
```

Isto é particularmente útil para parâmetros de saída (out), onde deixa de ser necessário declará-los previamente, numa linha separada. Isto é agradável na maioria dos casos, mas em cenários onde apenas expressões são permitidas, é necessário para a utilização de parâmetros de saída. Em cláusulas de consulta, por exemplo:

```
from s in strings
select int.TryParse(s, out int i) ? i : -1;
```

Intuitivamente, o escopo das variáveis declaradas numa expressão estende-se para o bloco exterior mais próximo, instrução estruturada (como if ou while) ou instruções embebidas (como o corpo de um if ou while). Também corpos de expressões lambda, cláusulas de consulta e inicializadores de campos e propriedades atuam como limites de escopo.

Na maioria dos casos, as expressões de declaração permitem var apenas se existir um inicializador de que se possa inferir o tipo, tal como nas instruções de declaração. No entanto, se a declaração da expressão é passado num parâmetro de saída (*out*), será tentada a resolução de sobrecargas (*overlaod resolution*) sem o tipo desse argumento e inferido o tipo a partir do método selecionado:

```
void GetCoordinates(out int x, out int y) { … }
GetCoordinates(out var x, out var y); // inferido
//int
```

As expressões de declaração podem requerer alguma habituação e pode ser sujeitas a abusos em novos e interessantes modos. Experimentem e digam o que pensam.

Filtros De Exceções

Os filtros de exceções já existiam no Visual Basic e no F# e agora existem também no C#. Este é o aspeto que terão:

```
try
{ … }
catch (Exception e) if (myfilter(e))
{
    …
}
```

Se a expressão entre parêntesis avaliar para verdadeiro, o bloco catch é executado, caso contrário a exceção continua o seu caminho.

Os filtros de exceções são preferíveis a apanhar e relançar porque deixam o *stack* inalterado. Se a exceção levar posteriormente a que haja um dump do *stack*, pode-se ver de onde foi originalmente lançada em vez de de onde foi relançada.

Literais Binários E Separadores De Dígitos

Nota: Na atual versão de antevisão, esta funcionalidade está disponível apenas para Visual Basic.

Pretende-se permitir literais binários em C#. Já não vai ser necessário pertencer-se à irmandade secreta do Hex para especificar vetores de *bits* ou valores do tipo *flag*!

```
var bits = 0b00101110;
```

Para literais longos (e estes novos literais binários podem facilmente ser muito longos) a possibilidade de especificar dígitos em grupos pode ser muito útil. O C# permitirá o uso da barra inferior em literais para separar grupos de dígitos:

```
var bits = 0b0010_1110;
var hex = 0x00_2E;
var dec = 1_234_567_890;
```

Podem se usar quantos se quiser e onde se quiser, exceto no início.

Membros Indexados E Inicializadores De Elementos

Os inicializadores de objetos e coleções podem ser muito uteis para inicializar campos e propriedades de objetos de forma declarativa ou para dar a uma coleção um conjunto inicial de elementos. A inicialização de dicionários não é tão elegante. Nesta versão é adicionada uma nova sintaxe aos inicializadores de objetos que permite atribuir valores diretamente a chaves através de qualquer indexador que o novo objeto possua:

```
var numbers = new Dictionary<int, string>
{
    [7] = "sete",
    [9] = "nove",
    [13] = "treze"
};
```

No caso de as chaves serem strings, algumas vezes quere-se pensar nelas quase como sendo um tipo fraco de membros do objeto. Este é o caso, por exemplo, quando se tem dados semiestruturados em formatos de comunicação como JSON, que é como se fosse um objeto mas não é fortemente tipado em C#. É adicionada nesta versão uma sintaxe para "membros indexados", que permitem fazer de conta que uma chave literal string é como se fosse um membro:

```
var customer = new JsonData
{
   $first = "Anders",  // => ["first"] = "Anders"
   last = "Hejlsberg" // => ["last"] = "Hejlsberg"
};
string first = customer.$first;
                          // => customer["first"]
```

Os membros indexados são identificados com o símbolo $ e podem ser usados em inicializadores de objetos e acesso a membros. São simplesmente traduzidos para a forma inferior que é indexar com literais do tipo string podendo, portanto, ser utilizados onde quer que o objeto rector tenha um indexador que receba uma única chave do tipo string. No entanto, dada a fraca receção pela comunidade desta funcionalidade, foi decidida a sua remoção na versão final.

Await Em Blocos Catch E Finally

No C# 5.0 a palavra-chave **await** não era permitida em blocos **catch** e **finally** porque se pensava não ser possível a sua implementação. Finalmente descobriu-se como poderia ser implementado.

Isto era uma limitação significativa e obrigava a código nada bonito para dar a volta e compensar esta falta. Tal já não é mais necessário:

```
Resource res = null;
try
{
    // Podia-se fazer isto.
    res = await res.OpenAsync(…);
    …
}
catch (ResourceException e)
{
    // Agora pode-se fazer isto ...
    await res.LogAsync(res, e);
}
finally
{
    // ... e isto
    if (res != null) await res.CloseAsync();
}
```

A implementação é algo complicada, mas não temos de nos preocupar com isso. É essa a razão de ter async na linguagem.

Métodos De Extensão Add Em Inicializadores De Coleções

Quando inicialmente se implementaram os inicializadores de coleções no C#, os métodos Add que são chamados não podiam ser métodos de extensão. No Visual Basic isto foi uma possibilidade desde o início e o C# parece ter sido esquecido. Isto foi resolvido: o código gerado para um inicializador de coleção poderá usar um método de extensão chamado Add. Não é uma grande funcionalidade, mas é útil e, afinal, a sua implementação no novo compilador consistiu em remover a validação que a impedia.

Funcionalidades Não Implementadas

Algumas funcionalidades não foram ainda implementadas mas estão já planeadas ou em consideração para versão final do C# 6.0. Sendo assim, algumas funcionalidades poderão ainda sofrer alterações ou ser retiradas em função do *feedback* da comunidade.

Declaração De Corpo De Membros Por Expressão

Esta funcionalidade planeada consiste na declaração de propriedades apenas de leitura e métodos através de uma simples expressão:

```
public class Point(int x, int y)
{   public int X => x;
    public int Y => y;
    public double Dist => Math.Sqrt(x* x + y* y);
    public Point Move(int dx, int dy) => new
                                Point(x + dx, y + dy);
}
```

Inicializadores De Eventos

Esta funcionalidade planeada vai permitir a inicialização de eventos nos inicializadores de objetos:

```
var botão = new Button
{
    ...
    Click += (source, e) => ...
    ...
};
```

Operador ;

Esta funcionalidade ainda em consideração permitiria que uma sequência de instruções seja combinada de forma a formar uma única instrução lógica:

```
(var x = Foo(); Write(x); x*x)
```

Params IEnumerable

Esta funcionalidade planeada vai permitir a utilização, para além de arrays, de enumeradores em conjugação com o modificador params:

```
int Avg(params IEnumerable<int> numbers) { … }
```

Interpolação De Strings

Esta funcionalidade ainda em consideração permitiria a utilização de identificadores na construção de strings:

```
"\{p.First} \{p.Last}  is \{p.Age}  years old."
```

Trata-se de uma funcionalidade que tem gerado alguma discussão que pode ser seguida aqui.

Operador NameOf

Esta funcionalidade ainda em consideração permitiria a obtenção do nome de um membro de um tipo à semelhança de *tipeof*:

```
string s = nameof(Console.Write);
```

Propagação De Null

Esta funcionalidade planeada vai permitir escrever código mais conciso quando está em causa um acesso encadeado em que podem ocorrer valores *null*.

No caso mais simples, irá permitir quando agora se escreve:

```
var temp = GetCustomer();
string name = (temp == null) ? null : temp.name;
```

passar a escrever:

```
var temp = GetCustomer()?.name;
```

Mas como é que este novo operador (?.) se vai comportar em sequências? Existem duas hipóteses: ou é associativo à esquerda ou é associativo à direita.

Associatividade À Esquerda

Esta opção significaria que sequências do tipo a?.b?.c seriam equivalentes a ((a?.b)?.c). Uma consequência desta abordagem é que, uma vez usado o operador de propagação de *null* (?.), teria de ser usado em todas os acessos posteriores na sequência porque o valor sobre o qual se estaria a aceder ao membro seguinte poderia ser nulo (*null*).

Associatividade À Direita

Esta opção significaria que sequências do tipo a?.b?.c seriam equivalentes a (a?(.b?.c)). Uma consequência desta abordagem é que o operador de propagação de *null* (?.) pode ser usado em combinação com o operador . em qualquer combinação. Código como o seguinte seria válido:

```
int? l = cliente?.Nome.Length;
```

Discussão E Opção Escolhida

Após longa discussão pública, optou-se pela associatividade à direita por ser a que a maioria dos intervenientes achou mais "natural" e útil.

> **"(...)a Microsoft disponibilizou uma versão de antevisão da versão 6.0 da linguagem de programação C# (...) inaugura também uma nova postura da Microsoft no que diz respeito ao código aberto (...)"**

Private Protected

A combinação dos modificadores de acessibilidade protected e internal em C# corresponde ao modificador de acesso FamillyOrAssembly da CLR. A aplicação deste modificador significa que este membro ou tipo apenas é acessível por tipos que estejam na mesma *assembly* do tipo declarante **ou** sejam derivados deste.

Uma vez que a CLR também tem um modificador de acesso FamillyAndAssembly que faz com que o membro seja acessível apenas por tipos que estejam na mesma *assembly* do tipo declarante **and** sejam derivados deste.

A proposta inicial foi a utilização combinada dos modificadores private e protected, mas não foi bem recebida pela comunidade. Após longa discussão pública, ainda não há, à data da escrita deste artigo, uma decisão definitiva. Foi criado um inquérito para escolher a opção mais consensual, mas poderá já não estar disponível na altura que este artigo for publicado.

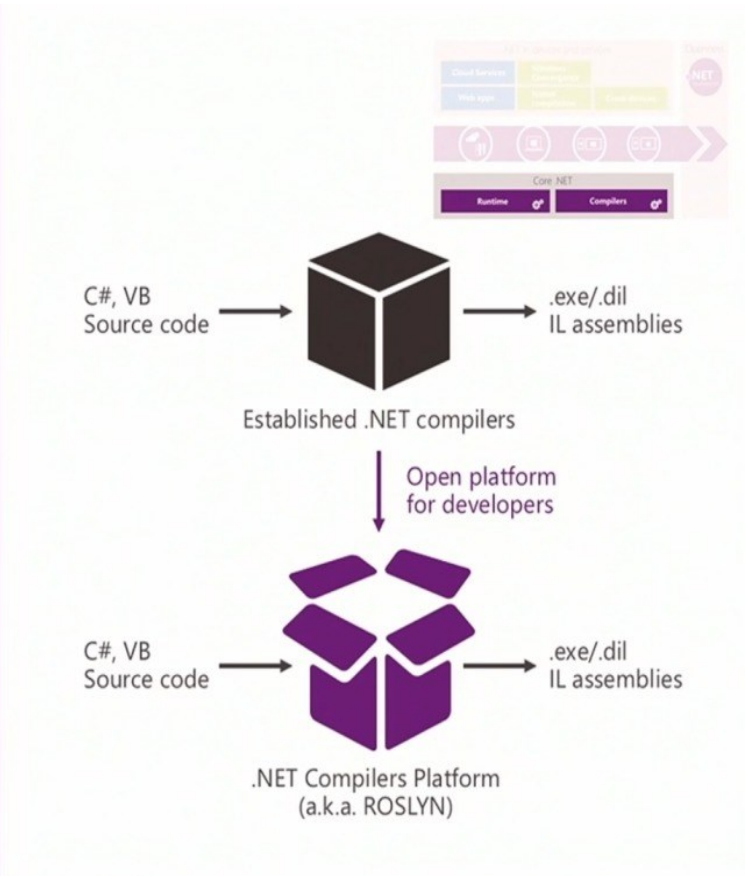
Conclusão

Esta versão não introduz funcionalidades fraturantes como foi o caso do LINQ ou async-await porque os esforços da equipa têm estado na nova plataforma de compiladores. Mas mesmo assim foram introduzidas algumas funcionalidades úteis e a nova plataforma permite explorar mais facilmente novas funcionalidades.

A grande novidade é a abertura ao público em geral da discussão e desenvolvimento da linguagem e plataforma de compiladores. Participem na discussão.

Recursos

- //build/
- Sessões do //build/ 2014
- Anders Hejlsberg
- Projeto Roslyn
- Notas das reuniões de desenho do C#
- Estado da implementação das funcionalidades das linguagens C# e Visual Basic



AUTOR

Escrito por Paulo Morgado
Bacharel em Engenharia Electrónica e Telecomunicações (Sistemas Digitais) pelo Instituto Superior de Engenharia de Lisboa e Licenciado em Engenharia Informática pela Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa exerce variadas funções relacionadas com o desenvolvimento, distribuição e manutenção de software há mais de 10 anos. Participa em diversas comunidades nacionais e internacionais (pontoNETpt, NetPonto, SharePointPT, SQLPort, Portugal-a-Programar, CodeProject, CodePlex, etc.). Pelo seu contributo para com estas comunidades, a Microsoft premeia-o com o prémio MVP (C#) desde 2003. É ainda co-autor do livro "LINQ Com C#" da FCA. http://PauloMorgado.NET/ - @PauloMorgado

MÉTODOS DE EXTENSÃO – O QUE PRECISO, COMO QUERO

O que preciso, como quero… mais ou menos. Mais para mais.

Comecemos por o básico: o que são métodos de extensão?

Em poucas palavras, métodos de extensão são uma forma de injetar funcionalidades escritas por "nós", personalizadas, diretamente em tipos que tomamos como "fechados", quer sejam os escritos por a Microsoft ou os escritos por o vizinho de cima.

Quando escrevo "injetar funcionalidades" estou-me a referir a métodos implementados por nós que para o Visual Studio fazem parte de determinada classe, e que podem ser chamados a partir de uma instância.

Isto significa que podemos até acrescentar métodos diretamente na classe String ou Integer? Bom, sim!

Note-se que os métodos de extensão não se tratam de material novo! Foram implementados na versão 9 do VB (incluída no Visual Studio 2008).

São no entanto, (e por razões que desconheço) pouco conhecidos, se bem que tenha a certeza de que já tenham usado extensões centenas de vezes sem darem por isso.

Basta continuar a ler, não só para saber onde tem usado extensões e como é que as pode identificar, mas também para aprender (naturalmente) como as escrever. É fácil. Prometo.

A esta altura é pertinente, com certeza: "como?"

Como devem calcular, não estamos na verdade a escrever o método diretamente na classe alvo.
O método é descrito e implementado em um módulo e tem de estar devidamente assinalado como extensão. Por exemplo:

```
<Extension()>
Public Sub FazerNada(Str As String)
'faço o que prometo!
End Sub
```

É um método de extensão da classe String.

A classe Extension fornecida como atributo do método pertence à classe CompilerServices em System.Runtime.CompilerServices. O módulo que usarem para descrever as extensões deverá importar este namespace para facilitar o uso do atributo.

O tipo que o método pretende estender é definido por o tipo do primeiro parâmetro, o que obriga a que um método de extensão tenha, no mínimo, um parâmetro. Se existirem mais parâmetros, já irão fazer parte da assinatura do método.
Estamos a falar, portanto, de magia do compilador!

O Visual Basic fica a saber que tem de invocar o método como sendo um método auxiliar comum, mas leva automaticamente a referência à instância que originou a invocação no primeiro parâmetro.

Eu tenho o meu módulo de métodos auxiliares. Vou usar extensões para quê?

Os métodos de extensão são, no fundo, uma encarnação dos vossos métodos auxiliares… mas no devido contexto.

Só para estarmos a pensar na mesma situação, quando escrevo métodos auxiliares estou a pensar naqueles pequenos métodos que ajudam a desempenhar um ou vários processamentos sobre um tipo, como por exemplo, validar uma String, formatar uma String, verificar se um número é ímpar, remover duplicados de um array, contar o número de palavras de uma String, e por aí adiante.

Vou utilizar, a título de exemplo, um método auxiliar que coloca um ponto final em uma frase (String) e uma letra "grande" no início, por exemplo: "gostava de escrever melhor" passa para "Gostava de escrever melhor."

```
    Public Sub Pontuar(ByRef Frase As String)
        Frase =
            Frase(0).ToString.ToUpperInvariant &
            Frase.Substring(1)
       For Each final As String In {".", "!", "?"}
            If Frase.EndsWith(final) Then Exit Sub
        Next
        Frase &= "."
    End Sub
```

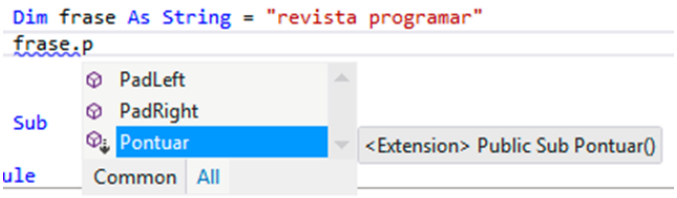
Traduzindo este método auxiliar para extensão, ficaria:

```
<Extension()>
    Public Sub Pontuar(ByRef Frase As String)
        Frase =
            Frase(0).ToString.ToUpperInvariant &
            Frase.Substring(1)
       For Each final As String In {".", "!", "?"}
            If Frase.EndsWith(final) Then Exit Sub
        Next
        Frase &= "."
    End Sub
```

Não precisam de procurar diferenças. Eu digo-vos o que mudou: nada. A grande maioria dos métodos auxiliares podem ser traduzidos apenas colocando o atributo que o transforma em extensão.
Neste caso, a única diferença nem sequer está relacionada com a implementação em si, trata-se apenas do atributo do método.
Onde é que está então a grande diferença? Existem duas grandes diferenças, mas não estão na forma como se descreve ou implementa o método. Estão na forma como o método é **invocado** e **descoberto**.

A primeira grande vantagem de utilizar extensões, em detrimento de métodos auxiliares, é o facto de estas se encontra-

rem perfeitamente integradas com o Intellisense (as caixas de sugestão de contexto), no contexto do tipo que estão a estender:



Sendo **frase** do tipo String, as sugestões dos membros vão incluir no separador "All" as nossas extensões desse tipo. Se não fosse extensão, o método teria de ser chamado de forma arbitrária, ou seja, eu teria de me lembrar que existia um método chamado Pontuar:

```
Dim frase As String = "revista programar"
frase.Pontuar() 'recorrendo à extenção
Pontuar(frase) 'recorrendo à extenção
```

A segunda grande diferença/vantagem passa também por aqui: não é necessário fornecer a referência a trabalhar, explicitamente. O Visual Basic sabe que a referência a trabalhar é a mesma que provocou a invocação. Isto **possibilita invocações em cadeia**. Neste caso não é evidente, porque uma segunda chamada ao método Pontuar na mesma referência não vai alterar nada, e também não é possível, dado que o método Pontuar não produz valor, mas a título de exemplo vamos assumir que o método Pontuar é uma Function que devolve uma String que representa a frase pontuada.
Na abordagem sem extensões, aplicar o método duas vezes seria da seguinte forma:

frase = Pontuar(Pontuar(frase))

O resultado da invocação interior era utilizado como alvo de trabalho na invocação exterior.
Com extensão, a sintaxe torna-se muito mais natural, e está acompanhada por o Intellisense:

frase = frase.Pontuar.Pontuar()

Frase é uma String que é processada com a primeira extensão e produz uma String que é processada com a extensão que lhe sucede.

Existe uma inegável vantagem em utilizar extensões em tudo o que toca, não só à organização e legibilidade do código, mas também ao vosso índice de produtividade. Enfim, todas as vantagens que se podem ter com a utilização do Intellisense, aplicadas a um pedaço de código escrito para trabalhar tipos que não controlamos diretamente. E quem diz métodos auxiliares diz qualquer tipo de método. Manda a necessidade e criatividade.

Alguns exemplos de extensões

É possível estender classes, estruturas, interfaces e delegates.

Para demonstrar vários tipos de assinaturas e extensões, deixo-vos com alguns exemplos.
De notar que são sugestões de implementação, não garanto que sejam as formas mais eficazes de desempenhar a função.

Posso começar por reescrever a extensão Pontuar para que produza um valor, que é como deve ser implementada:

```
<Extension()>
Public Function Pontuar(ByRef Frase As String)
                                        As String
        Frase =
            Frase(0).ToString.ToUpperInvariant &
            Frase.Substring(1)
       For Each final As String In {".", "!", "?"}
            If Frase.EndsWith(final) Then Return
            Frase
        Next
        Return Frase & "."
    End Function
```

Invocação: MinhaFrase.Pontuar()

O único parâmetro indica a classe que vai estender, a String. Sem parâmetros adicionais.

Um outro exemplo de extensão, desta vez uma extensão do Integer que indica se determinada instância representa um número ímpar ou não:

```
<Extension()>
Public Function NumeroImpar(ByVal num As Integer)
                                        As Boolean
        If num Mod 2 = 0 Then Return False Else
  Return True
End Function
```

Invocação: MeuNumero.NumeroImpar()

As extensões também não precisam forçosamente de trabalhar a instância. Podem simplesmente fazer qualquer outra coisa com ela. Segue um exemplo que estende qualquer classe e que escreve para debug a sua representação em String:

```
<Extension()>
Public Sub [Debug](valor As Object)
  Dim momento As DateTime = Now()
  Dim output As New System.Text.StringBuilder("[")
     output.Append(momento.ToShortDateString)
     output.Append("][")
     output.Append(momento.ToShortTimeString)
     output.Append("] Dump de variável: ")
     output.Append(valor.ToString)
     System.Diagnostics.Debug.WriteLine
                                (output.ToString)
End Sub
```

Invocação: MinhaVariavel.Debug()

Uso os parêntesis retos porque Debug já é o nome de uma classe existente nos namespaces importados por defeito. Ainda que neste caso não houvesse perigo de colisão, é boa prática.
Com uma simples invocação, o método regista em debug a data e hora, seguida da representação.

MÉTODOS DE EXTENSÃO – O QUE PRECISO, COMO QUERO

Posso dar um outro exemplo de uma extensão que não transforma a instância, mas faz algo diferente com ela:

```
<Extension()>
Public Sub Falar(texto As String)
Dim s As New System.Speech.Synthesis.
                                SpeechSynthesizer()
        s.Speak(texto)
End Sub
```

Invocação: TextoAFalar.Falar()

Estendemos a String e usamos o sintetizador de discurso para "falar" o conteúdo da String.

As extensões também são úteis quando queremos saber outro tipo de informações relacionadas com o tipo da instância, como por exemplo, devolver o número de palavras numa String:

```
<Extension()>
Public Function Palavras(ByVal texto As String) As
                                              Integer
    While texto.Contains("  ") Or
                            texto.Contains(vbTab)
        texto = texto.Replace("  ", " ").
                        texto.Replace(vbTab, " ")
    End While
    Dim p As String() = texto.Split(" ")
    Dim inc As Integer = 0
    For Each tmp As String In p
        If tmp.Length > 1 Then inc += 1
    Next
   Return inc
End Function
```

Invocação: AMinhaFrase.Palavras()

Por fim, e como exemplo mais interessante, podemos jogar com os tipos genéricos e implementar um método semelhante ao Where das coleções de IEnumerable, mas para aplicar em arrays:

```
<Extension()>

Public Function Filtrar(Of T)(arr As T(), filtro
                  As Func(Of T, Boolean)) As Array
  Dim ret(0) As T
  For i As Integer = 0 To arr.Length - 1
      Dim tmp As T = arr(i)
      If filtro.Invoke(tmp) Then
        If i < arr.Length - 1 And i <> 0 Then
                    ReDim Preserve ret(ret.Length)
         ret(ret.Length - 1) = tmp
        End If
   Next
   Return ret
End Function
```

Ao indicarmos que o primeiro parâmetro é do tipo Array de T, estamos na prática a indicar que estamos a estender qualquer array de qualquer tipo. Como existem especificidades para comparar cada tipo, passamos um segundo parâmetro (que é na verdade o único, como já sabem) que é um delegate. A invocação deste método tem de levar como parâmetro uma função que devolva True ou False, sendo o True para determinar elementos que obedecem ao filtro, e False os que se vão descartar.

Invocação exemplo: MinhaArrayDeInteger.Filtrar(Function(el) el < 10)

Esta invocação resulta num novo array cujos elementos são os da instância invocadora, quando estes inferiores a 10. Ou seja, por exemplo:

{2,45,67,4,9,10}.Filtrar(Function(el) el < 10)

Devolve: {2,4,9}

Como identificar extensões que já usava?

Já tiveram, certamente, oportunidade de reparar que a extensão é identificada na caixa de sugestões com um símbolo ligeiramente diferente do símbolo do método: a extensão apresenta uma pequena seta escura, no sentido descendente, do lado direito.

Quantas vezes já utilizaram métodos Microsoft com este símbolo? As extensões mais famosas, e que mais devem usar, são as extensões LINQ. Repararam que o **Where**, o **Find**, e até o **Count** de coleções de IEnumerables são extensões?

AUTOR

Escrito por Sérgio Ribeiro
Curioso e autodidata com uma enorme paixão por tecnologias de informação e uma saudável relação com a .NET Framework.
Moderador global na comunidade Portugal@Programar desde Setembro de 2009. Alguns frutos do seu trabalho podem ser encontrados em http://www.sergioribeiro.com

CORE DUMP [12] - PROGRAMADOR ^ GESTOR

Uma evolução típica na carreira de programador é chegar a gestor. O PM, a carinhosa abreviatura de Project Manager, é muitas vezes alguém que chega à gestão de projeto tendo feito um percurso técnico e, tantas vezes, sem formação em gestão.

O facto de se ser um bom programador não habilita ninguém a ser um bom gestor de equipa ou de projeto na área das tecnologias de informação. Infelizmente, é comum promover os melhores engenheiros de software a gestores menos capazes.

Este percurso parece lógico na visão das organizações, em particular das mais tradicionais, que não compreendem a necessidade de proporcionar uma carreira técnica a estes profissionais. Como tal, na sua visão, o resultado do bom trabalho que desenvolvem no desenvolvimento de software é a promoção para cargo de gestão.

Se as organizações têm responsabilidade neste ponto, os engenheiros de software também têm a sua quota-parte, uma vez não explicam que essa promoção não faz sentido sem, pelo menos, formação adequada na área de gestão.

Quando uma promoção deste tipo acontece, é comum ver o novo gestor a lutar com extrema dificuldade em saber o que fazer, em especial com imensos problemas em conseguir largar a sua zona de conforto e a fazer aquilo que o seu novo cargo exige. Já assisti variadíssimas vezes a situações destas, em que **o novo PM continua a fazer desenvolvimento porque tem dificuldade em delegar essas tarefas na sua equipa, em gerir os recursos que tem, a chamar a si as tarefas mais complexas por achar que o faz melhor do que qualquer outro na sua equipa** e, pior a não conseguir controlar o projeto e, consequentemente a aumentar os níveis de stress de toda a gente envolvida no processo.

Quando as coisas descarrilam, é comum entrar nos cliches do “vamos fazer um esforço”, o que faz com que o novo PM tenha de se esforçar por fazer com que a sua equipe conclua as tarefas a tempo e com a qualidade necessária para que a entrega do projeto não seja comprometida. Para tal, as atitudes do novo PM são, muitas vezes, mal interpretadas e tantas vezes vista como “alguém a quem a promoção lhe subiu à cabeça”. Simultaneamente, do outro lado, o sentimento do novo PM é de frustração e até alguma agonia por ter de tomar atitudes que já criticou no passado com pessoas de quem gosta.

Foto: filme Office Space

Toda esta situação, embora comum, não condena o novo PM a ser um mau gestor para sempre. Na verdade, **o facto de compreender muito bem toda a componente de engenharia de software e de conhecer os elementos da sua equipa, permite-lhe planear, gerir e controlar de forma a que a sua equipa tenha uma boa produtividade**. O resultado faz com que a organização tenha uma excelente equipa de engenharia de software, dando-lhe assim uma vantagem competitiva sobre a sua concorrência.

Mas para tal, ambas as partes, o novo PM e a organização, têm de ter consciência de que existe um processo de aprendizagem e de que deve ser dada formação adequada à pessoa a promover. Se tal for feito, a transição será mais pacífica, o novo PM não se sentirá frustrado e a organização obterá o rendimento desejado de um PM.

Mas, **e se a pessoa em causa não quiser sair da carreira técnica?** Tal como referi no inicio, as organizações devem estar preparadas para estas situações, e devem conseguir proporcionar uma evolução na carreira técnica.

Cada vez mais as organizações têm esta consciência, mas ainda estamos longe de deixar de promover bons engenheiros de software a maus gestores.

AUTOR

Escrito por Fernando Martins
Faz parte da geração que se iniciou nos ZX Spectrum 48K. Tem um Mestrado em Informática e mais de uma década de experiência profissional nas áreas de Tecnologias e Sistemas de Informação. Criou a sua própria consultora sendo a sua especialidade a migração de dados.

Media Partners da Revista PROGRAMAR

Análises

Segurança em Redes Informáticas (4. Ed. Aumentada)

Estruturas de Dados e Algoritmos em C

Segurança em Redes Informáticas (4. Ed. Aumentada)

Título: Segurança em Redes Informáticas (4.ª Ed. Aumentada)

Autor: André Zúquete

Editora: FCA - Editora de Informática, Lda.

Páginas: 432

ISBN: 978-972-722-767-9

O livro "Segurança em Redes Informáticas (4.ª Ed. Aumentada)" é uma obra, segundo o autor, destinada a administradores de redes ou alunos na área de segurança de redes.

Não é propriamente um livro de consulta de bolso devido ao seu conteúdo técnico e extensividade. O próprio autor o indica que o livro "Segurança em Redes Informáticas não pretende ser um catálogo de problemas e soluções".

O passo-a-passo dos diversos procedimentos dos tópicos descritos no livro foi deixado de lado.

Esta 4.ª edição tem como principais temas os seguintes:

- Criptografia

- Gestão de Chaves Públicas (inclui uma descrição detalhada do funcionamento do nosso Cartão de Cidadão)

- Vulnerabilidades em Máquinas de Sistemas Distribuídos

- Vulnerabilidades em Redes Locais e de Grande Escala

- Firewalls

- Sistemas de Detecção de Intrusão (inclui os casos de estudo dos sistemas mais populares actualmente - Tripwire, Snort e Antisniff)

- Redes Privadas Virtuais

- Segurança em Redes Sem Fios 802.11

- Protocolos de Autenticação

De facto, esta obra ilustra muito bem estes temas, onde saliento a explicação ao detalhe dos algoritmos e técnicas criptográficas, cujo o leitor poderá praticar diversos tipos de cifra. Esta versão aumentada, para além da revisão e o acréscimo de capítulos em relação à edição anterior, inclui mais dois temas: o Cartão de Cidadão e o capítulo novo relativo a Protocolos de Autenticação de pessoas e sistemas em redes informáticas.

Em relação ao Cartão de Cidadão, penso que é importante focar este tema dado que está presente no dia-a-dia dos portugueses. Conhecer a segurança deste smartcard é uma mais-valia. O livro oferece informação relativa ao tipo de autenticação, as assinaturas digitais, hierarquias de certificação e a segurança que o Cartão de Cidadão nos oferece. Na minha opinião, um dos tópicos mais interessantes. Para além disso, juntamente com o capítulo 9 (Segurança em Redes Sem Fios 802.11 / Wi-Fi), estes capítulos podem ser interessantes para o leitor comum, leia-se sem grande conhecimento da área de segurança de redes. Embora tenha termos técnicos, é sempre importante a informação e dicas fornecidas nestes dois capítulos.

Para ajudar o leitor, toda a estrutura da informação da obra está indicada para ser clara e fácil de identificar.

Ao longo do livro, o autor fornece diagramas explicativos e detalhados, pecando apenas na falta de exemplos práticos com ferramentas para o efeito. Talvez este ponto estivesse fora do objectivo do autor.

No final, o livro conta com uma tabela de correspondência dos principais termos técnicos em português europeu e o português brasileiro. Sem dúvida um conteúdo útil dado que a informação em português na web é bastante diversificada em ambos os países.

Para concluir, na minha opinião pessoal, o autor deveria referir por outros temas mais actuais o "ciberpraga" ou "atacante habilidoso". No meu ponto de vista, são termos desactualizados.

Eu recomendo o livro para administradores de redes que queiram aprender as bases e a teoria, dado que por vezes é esquecida em tutoriais e manuais de rede. Também recomendo o livro para uso de estudo aos docentes de "Segurança de Redes".

Parabéns ao autor André Zúquete por esta obra completa. Espero que continue actualizar com a mesma qualidade futuras edições.

AUTOR

Escrito por David Sopas

Analista de segurança web, programador, fundador, editor do WebSegura.net e líder do projeto ScanPW.

Colabora regularmente com a comunicação social em assuntos relacionados com a segurança da informação.

Estruturas de Dados e Algoritmos em C

Título: Estruturas de Dados e Algoritmos em C

Autor: António Adrego da Rocha

Editora: FCA - Editora de Informática, Lda.

Páginas: 616

ISBN: 978 – 972 – 722 – 769 – 3

Programação, a "linguagem do futuro" permite executar praticamente tudo, se não mesmo tudo aquilo que a nossa imaginação possa equacionar. No entanto o desenvolvimento de soluções de software de média e elevada complexidade trás consigo a necessidade de aprofundar os conceitos algorítmicos que no fundo são a base da programação.

É com esse mesmo objectivo de aprofundar os conceitos algorítmicos dos programadores que se apresenta este livro "Estruturas de Dados e Algoritmos em C".

Ao nível físico da edição não tenho nada de errado a apontar, sendo que a tipografia é de qualidade permitindo uma fácil e rápida percepção visual dos conteúdos abordados.

O livro "Estruturas de Dados e Algoritmos em C" é uma obra que se destina a programadores e a alunos de programação, apresentando ênfase na decomposição funcional das soluções recorrendo à implementação de tipos de dados abstractos.

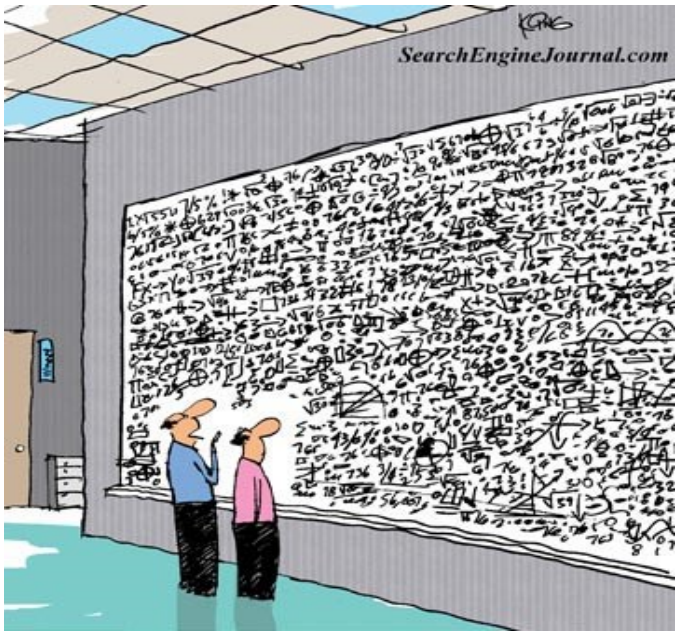
A organização dos conteúdos foi escolhida mediante a sua importância e dependência entre conteúdos, incluindo exemplos de cada tipo de implementação, exercícios complementares e até algumas recomendações de outras leituras sobre a temática em questão.

Dividido em cinco grandes temas e onze capítulos principais, as primeiras páginas da obra são orientadas a uma pequena introdução ao que a obra se propões a abordar assim como a tecer algumas considerações sobre a utilidade e importância das estruturas de dados e algoritmos na programação.

Os grandes temas abordados na obra estendem-se desde o estudo do paradigma da programação modular em C, apresentando os aspectos fundamentais da implementação de tipos de dados abstractos recorrendo à metodologia de programação defensiva, estudo das principais estruturas de dados dinâmicas, principais classes de algoritmos, implementação de diferentes tipos de memórias e por fim o estudo dos tipos de dados abstracto grafo / dígrafo e os seus algoritmos mais importantes.

No final de cada sub-capítulo existe um resumo dos conteúdos a reter o que facilita de certa forma a revisão de conteúdos quando já sabemos previamente em que tema se insere.

O conteúdo é de simples leitura e de fácil percepção, sendo que é usada uma linguagem muito "user-frendly" o que facilita muito a leitura da obra, não se tornando extremamente técnica do ponto de vista da linguagem utilizada. O uso de exemplos explicativos de código implementado e devidamente comentado são uma enorme ajuda na percepção da real implementação dos conteúdos abordados.



Como estudante de Engenharia Informática considero a obra "Estruturas de Dados e Algoritmos em C" uma obra de qualidade na apresentação dos conteúdos necessários para aprender e conhecer com alguma profundidade as estruturas de dados e algoritmos que tão importantes são no dia-a-dia de um programador ou estudante de programação, tendo tudo o que é necessário para se tornar uma obra de referência sobre o tema e escrita em língua Portuguesa.

AUTOR

Escrito por Nuno Santos

Curioso e autodidacta com uma grande paixão pela programação e robótica, frequênta o curso de Engenharia Informática na UTAD alimentando o sonho de ainda vir a ser um bom Engenheiro Informático. Estudante, Blogger, e moderador no fórum Lusorobótica são algumas das suas actividades. Os seus projectos podem ser encontrados em: http://omundodaprogramacao.com/

COMUNIDADES

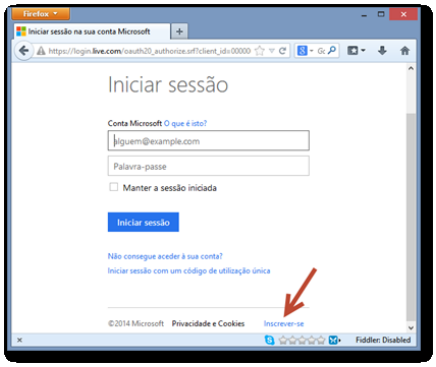
Comunidade NetPonto – Criando aplicações Windows Phone 8.1 e Windows 8.1 usando o App Studio da Microsoft

CRIANDO APLICAÇÕES WINDOWS PHONE 8.1 E WINDOWS 8.1 USANDO O APP STUDIO DA MICROSOFT

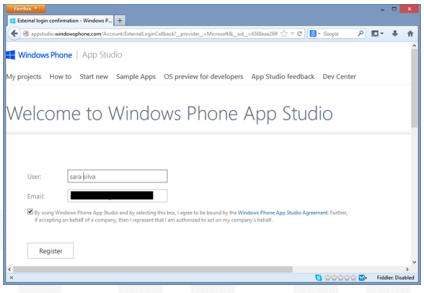
Este artigo tem como objetivo apresentar o App Studio da Microsoft, de forma a criar aplicações para Windows Phone 8.1 e Windows 8.1.

Há uns meses atrás a Microsoft lançou o Windows Phone App Studio Beta, serviço este que permite que qualquer pessoa sem conhecimentos de programação consiga criar aplicações para Windows Phone em apenas 4 passos: Tenha uma ideia, adicione o conteúdo, escolha o estilo e use a aplicação. Recentemente, mais especificamente no evento //Build/ da Microsoft que se realizou no passado mês de Abril, foi lançada uma nova versão deste serviço e foi lançado também a versão Windows App Studio Beta, que para além das aplicações de Windows Phone, agora passamos a poder criar aplicações Windows 8.1 neste serviço.

O serviço pode ser acedido a partir de appstudio.windowsphone.com. E para começar a usá-lo é necessário uma conta Outlook, Live Id ou no caso de não ter uma conta destas deve usar uma conta que esteja associada ao Windows Live Id. Caso não esteja associado, pode associar qualquer endereço de email a este serviço na seguinte referência: http://bit.ly/1gywtmO ou na página de login, clicar em Inscrever.

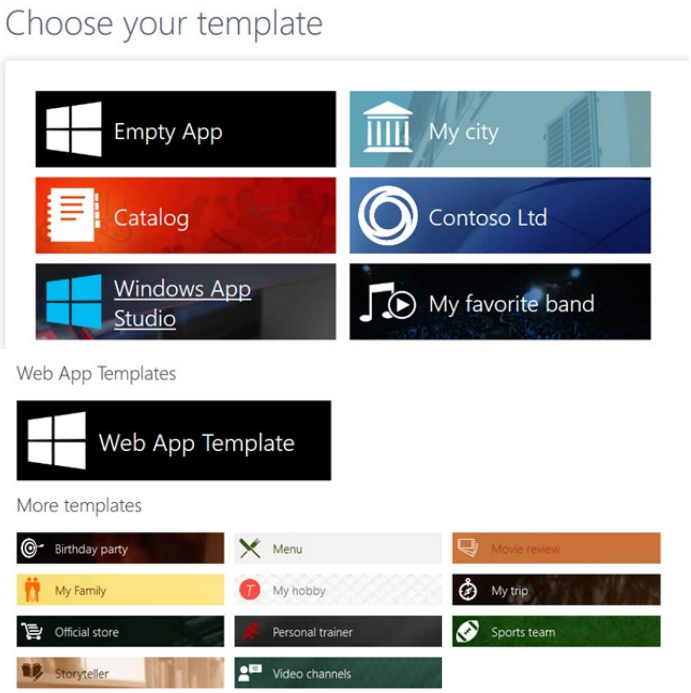


Ao fazer o login pela primeira vez, irá aparecer um ecrã de permissões de acesso à sua conta, no entanto esta permissão pode vir a ser alterada mais tarde se assim o desejar. A seguir, terá que aceitar os temos de utilização do serviço e definir um utilizador (User) para a conta de e-mail que está usar.



Neste momento estamos prontos para começar a criar aplicações! Para tal basta clicar no botão “Start new project”.

O App Studio fornece um conjunto de templates para ajudar os utilizadores na criação das aplicações. Estes templates não são mais que aplicações demos de temas específicos e mais usados.



De salientar que todos os templates são aplicações C#/XAML e estão disponíveis tanto para Windows Phone 8.1 como para Windows 8.1, exceto o Web App Template que apenas está disponível para Windows Phone.

Ao selecionar um template do tipo “Empty App” estamos a criar uma aplicação de raiz, do zero. Este template é muito utilizado em casos em que os outros templates não satisfaçam as necessidades do utilizador ou para o caso de utilizadores mais avançados que já tem alguma experiência com o App Studio. O utilizador irá definir todo o tipo de conteúdo, estilos e terá que usar toda a sua criatividade para obter uma “great app”. No caso do template “Web App Template” o utilizador irá também criar a aplicação do zero, mas neste caso é dado um Url de base que será o ponto inicial da aplicação e terá acesso a alguns botões na “AppBar”, este template é o ideal para site que se adaptam a dispositivos móveis. Por outro lado, os outros templates disponíveis já permitem ao utilizador ter um guia de

orientação na aplicação que está a construir e apenas terá que alterar os dados e customizar a aplicação a seu gosto. Note-se que neste caso o utilizador poderá mudar radicalmente toda a aplicação inicialmente criada pelo template.

Vejamos agora na prática como podemos criar aplicações com estes templates.

Contoso Ltd Template

Depois de selecionar um template aparece um ecrã onde podemos visualizar a aplicação numa espécie de simulador de Windows Phone 8.1 e Windows 8.1. No entanto, não é permitido a navegação para a páginas de detalhe.

Neste ecrã já se consegue ficar com uma ideia da aplicação sem a ter que instalar no dispositivo. Vejamos então as configurações deste template.

Conteúdo (Content)

A primeira página apresenta o nome da aplicação e o logotipo da mesma (no topo do lado esquerdo) e apresenta a definição do conteúdo da aplicação, que é constituído por cinco secções:

- About us é uma secção do tipo HTML
- Catalog é uma secção do tipo coleção dinâmica
- Team é uma secção do tipo coleção dinâmica
- News é uma secção do tipo Bing
- Contact us é uma secção do tipo Menu, que contém ações de menus.

No máximo podem ser definidas seis secções para a aplicação, e depois de definida cada secção é possível editar, apagar ou mover de posição de forma a ordenar da melhor forma.

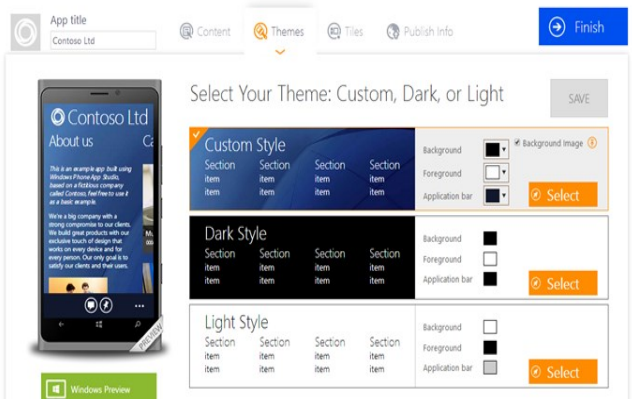
Para cada sessão é possível escolher uma das seguintes opções:

- Rss – adiciona a informação baseada no feed fornecido pela referência do rss
- HTML - permite adicionar uma página estática que incluiu código HTML permitindo criar conteúdo personalizado
- Youtube – lista de vídeos com base no identificador do utilizador no Youtube ou numa palavra de pesquisa
- Flickr – lista um conjunto de fotografias fornecidas com base no Flickr UserId ou numa palavra de pesquisa
- Bing – lista um conjunto de referências web
- Facebook – lista o feed da página do facebook com base no identificador da página
- Menu – permite criar uma lista de ações de menu que podem recorrer a outras aplicações para ações diversas (por exemplo: indicar a morada da empresa e usar o Here Maps, indicar o número de telemóvel para efetuar chamadas)
- Collection – permite definir dados estáticos ou dinâmicos. No caso de dados dinâmicos é permitido possível importar/exportar dados de um ficheiro CSV.

Uma futura funcionalidade será a fonte de dados twitter, que neste momento não está disponível no App Studio, mas que foi mencionada na sessão sobre o tema que decorreu no // Build/.

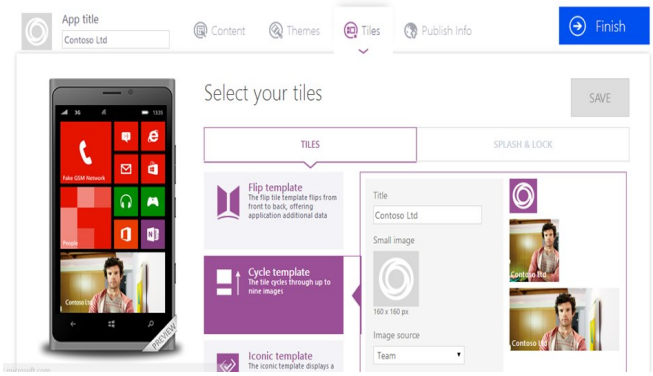
O App Studio já apresenta uma boa lista de fontes de dados, permitindo o utilizador criar aplicações interessantes, no entanto, o facto de se basear em feeds existe algumas limitações no número de dados disponíveis e na qualidade da apresentação dos mesmos.

Tema (Theme)



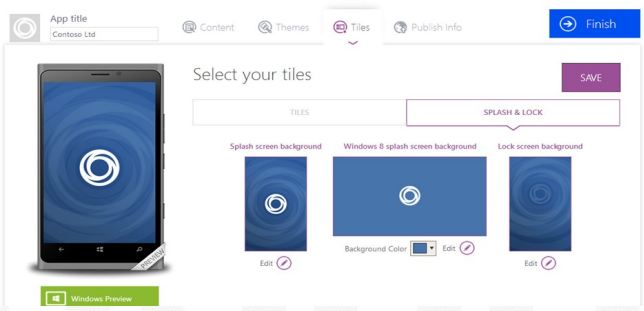
Nesta página podemos definir a imagem ou cor de fundo da aplicação, definir a cor do texto e a cor da "Application bar". Existindo três opções, customizado (usando a imagem de fundo), a preto (mais conhecido por Dark Style) ou a branco (mais conhecido por Light Style). O botão "Select" permite definir o estilo pretendido e depois disso é preciso fazer "Save" para guardar as configurações. Note-se que ao lado do texto "Background image" existe um botão que permite fazer o upload da imagem de fundo.

Mosaicos (Tiles)



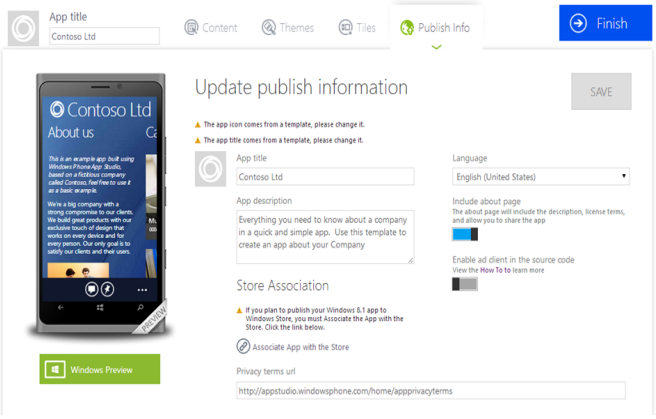
No separador Tiles, podemos definir que tipo de mosaicos (tiles) pretendemos para a aplicação, existindo três tipos: Flip Template, Cycle Template e Icon Template. Neste caso foi selecionado o Cycle Template e definido as imagens e texto pretendido.

No separador Splash & Lock, é definido as imagens para o splash screen das aplicações para Windows Phone 8.1 e para Windows 8.1 e o lock screen do Windows Phone.



Apesar de esta página não mencionar o tamanho das imagens, quando fazemos editar aparece um ecrã que nos informa sobre o tamanho possível de cada imagem.

Publicar (Publish)



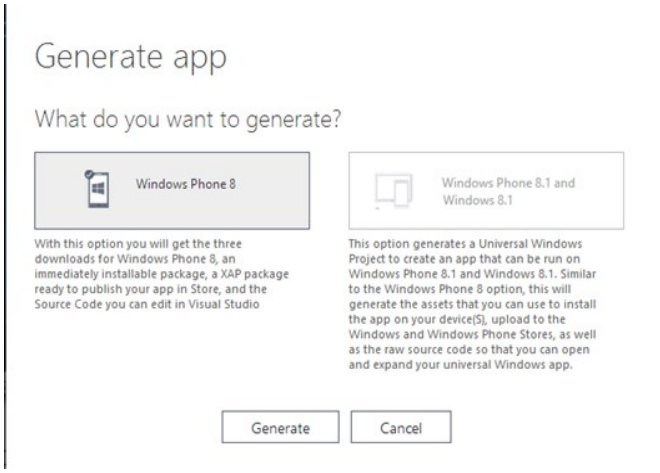
Nesta página é definido a língua usada pela aplicação, atualmente só é possível definir uma. É possível definir o titulo, a descrição da aplicação, permitir incluir a página Acerca de (About) e incluir publicidade na aplicação (que obriga a configurações finais no código). E ainda é possível associar a aplicação à Loja do Windows, que por sua vez requer que seja reservado um nome para a aplicação na Loja.

Depois de todas as configurações é possível finalizar a aplicação clicando no botão "Finish".

Finalizar (Finish)

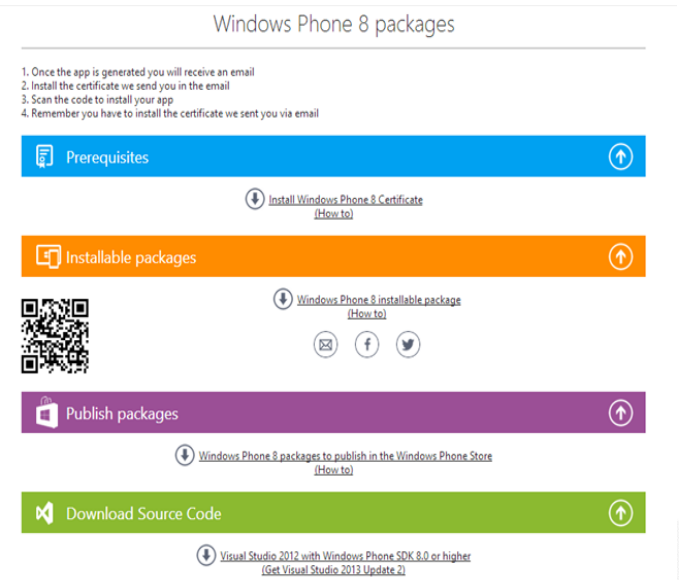
Nesta página podemos visualizar as aplicações para cada target (não sendo possível navegar) e é possível gerar a aplicação para assim obtermos o pacote da aplicação para instalar em cada dispositivo ou então obter o código fonte. Ao

clicar no botão “Generate” obtemos um ecrã onde podemos escolher qual a aplicação que pretendemos gerar.



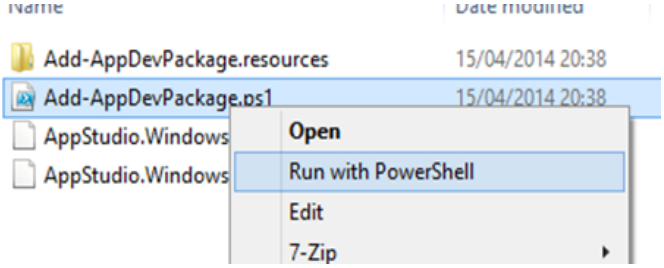
Podemos gerar para ambos os casos, que implica duas gerações diferentes. No entanto, para um utilizador mais experiente e que esteja a usar o Visual Studio para testar a sua aplicação, recomendo que seja feita a geração para Windows 8.1 porque o código gerado é uma solução “Universal app” que é constituído pelo projeto da aplicação Windows Phone 8.1, pelo projeto da aplicação Windows 8.1, por um projeto partilhado entre os projetos anteriores e neste caso ainda contém um projeto do tipo “portable class libray” com toda a estrutura de dados.

Depois de gerado temos disponível o pacote da aplicação que nos permite testar a aplicação no dispositivo ou obter o código fonte e testar no simulador ou no dispositivo usando o Visual Studio. Note-se que um utilizador que não tenha conta de “Developer” e por essa razão não tem o dispositivo desbloqueado, terá que instalar o certificado fornecido e só depois poderá instalar a aplicação no dispositivo usando para tal o QRCode da aplicação ou a referência web fornecida. É possível partilhar a aplicação nas redes sociais e por e-mail.



Ao gerar a aplicação para Windows 8.1 obtemos o seguinte ecrã:

Onde podemos obter o pacote da aplicação Windows 8.1 e depois usando PowerShell podemos instalar a aplicação num PC com Windows 8.1.



Depois desta geração, ao obter o código fonte vamos obter uma solução “Universal app”, que terá a seguinte estrutura:

Com o código fonte cada uma das aplicações pode ser extendida de forma a adicionar novas funcionalidades, tais como:

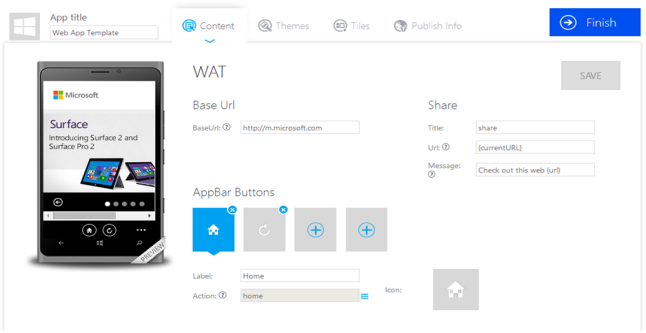
- Permitir mais do que uma língua
- Permitir publicidade na aplicação
- Suporte para twitter
- Alterar a interface com o utilizador

Entre outras…

Web App Template

Com já foi referido, este template só está disponível para aplicações de Windows Phone e trata-se uma aplicação Web. Vejamos então na prática como usar este template.

Ao selecionar o Web App Template iremos ter a seguinte página

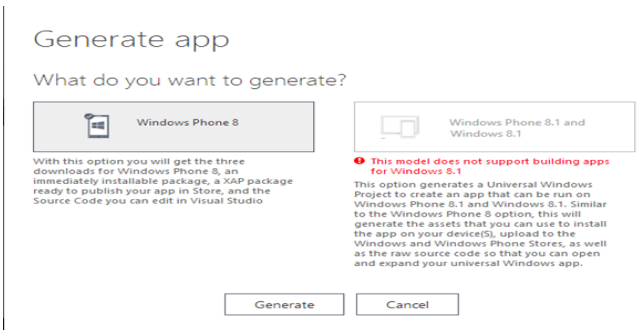


Como se pode ver, o conteúdo desta aplicação resume-se á referência de base da aplicação, que não é mais que um website que supostamente foi desenhado para dispositivos móveis.

Neste exemplo, está-se a usar o endereço m.microsoft.com mas poderíamos utilizar por exemplo: booking.com, m.sapo.pt, m.vodafone.pt, entre outras referências….

As definições para **Theme**, **Tiles** e **Publish** são iguais ao que foi apresentado no *template Contoso Ldt*.

Ao clicar em "Finish" iremos para a página de geração do pacote da aplicação e do código fonte. E ao clicar em "Generator" iremos ter o seguinte ecrã



Com já tinha referido, apenas iremos gerar a aplicação para Windows Phone. E como podemos ver no template do Contoso Ltd, neste caso também vamos ter disponível o certificado para instalar no dispositivo, a referência web e o QRCode do pacote da aplicação, partilha nas redes sociais e via e-mail e o código fonte.

O resultado final da aplicação, usando a referência de base m.sapo.pt, é

As aplicações geradas pelo App Studio, devem ser testadas em vários dispositivos para garantir que a aplicação está a funcionar correctamente e está de acordo com o que o utilizador definiu. Só depois de terminarem os testes é que a aplicação deve ser submitida na loja. De relembrar que na página de Publish tinhamos a opção de ligar a aplicação à loja, e na página final depois da geração da aplicação é nos fornecido a referência dos pacotes para submeter na loja. As aplicações geradas pelo App Studio são submitidas ao mesmo processo de cerfiticação que uma aplicação desenvolvida por um programador.

Os utilizadores mais avançado, que pretendam extender a aplicação dando novas funcionalidades não tem como "submeter" essas alterações no site do App Studio, para que numa nova versão da aplicação no App Studio essas alterações sejam tidas em conta. A solução passa por fazer um "merge" entre o código de fonte alterado e a nova versão da aplicação. E apartir do momento que o código fonte é alterado, o utilizador é que é responsável pela criação do pacotes da aplicação utilizado o Visual Studio.

Para terminar, deixo uma referência sobre o este tema que podem consultar em Building apps without code using AppStudio, onde é possível encontrar um conjunto de artigos, vídeo e fórum sobre o App Studio.

Em conclusão, conclui-se que o App Studio apresenta uma solução rápida no desenvolvimento de aplicações tanto para Windows Phone 8.1 como para o Windows 8.1, sendo possível extender as funcionalidades da aplicação através do código fonte gerado. De salientar que apesar de todo o automatismo associado, existem algumas limitações que devem ser trabalhadas de futuro para que as aplicações geradas pelo App Studio apresentem ainda mais qualidade.

AUTOR

Escrito Por Sara Silva

Licenciada em Matemática – Especialidade em Computação, pela Universidade de Coimbra e é Microsoft Certified Profissional Developer. Atualmente o seu foco de desenvolvimento incide em Windows Phone e Windows 8 Store Apps. O seu Blog **é** www.saramgsilva.com e o twitter é @saramgsilva.

No Code

O Windows Phone 8.1 e a atualização do Windows 8.1

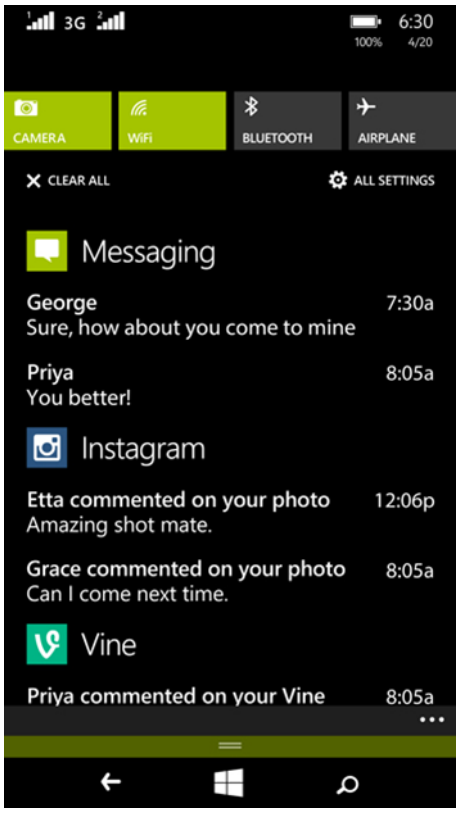
O Windows Phone 8.1 e a atualização do Windows 8.1

Este artigo tem como objetivo apresentar as principais novidades do Windows Phone 8.1 e da atualização do Windows 8.1.

No passado dia 2 de Abril, foi apresentado pela Microsoft na Keynote do //Build/, o Windows Phone 8.1 e uma atualização para o Windows 8.1. Para aqueles que não poderam assistir em direto no dia do evento, o vídeo da Keynote está disponível em http://bit.ly/1oxd0fg.

Windows Phone 8.1

O Windows Phone 8.1 apresenta uma grande novidade, que na opinião dos especialistas, vem melhorar a experiência de utilização do dispositivo, refiro-me ao Action Center!



O Action Center vai permitir o utilizador ficar a pare de todas as notificações das várias aplicações, assim como de chamadas não atendidas, SMS recebidas e que ainda não foram lidas e ainda é possível de forma rápida ativar o Modo de Avião ou ligar/desligar o acesso à internet.

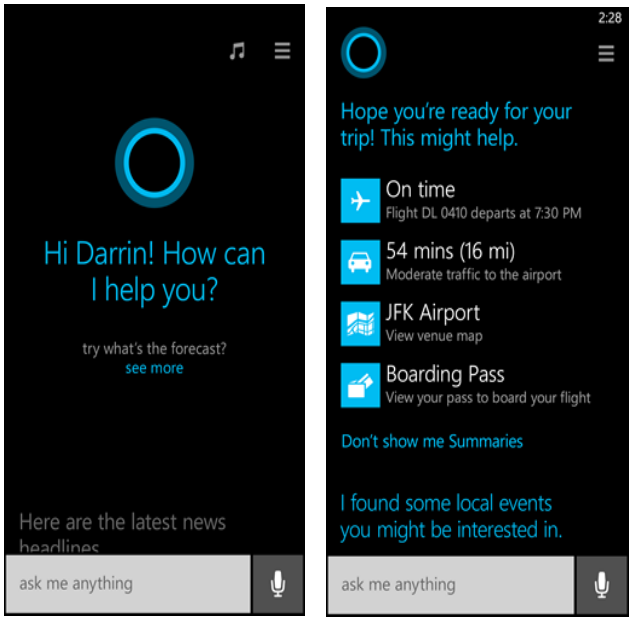
De salientar, que o menu rápido é personalizável e o Action Center está acessível mesmo que o dispositivo esteja bloqueado (LockScreen).

O ecrã de bloqueio apresenta uma alteração significativa, sendo possível configurar o mesmo. Exemplo disso é a possibilidade de manter o resumo do calendário e notificações das aplicações ou então escolher a apresentação semanal, destacando o dia da semana atual.

O ecrã principal (start screen) também apresenta novidades. A partir de agora é possível escolher uma fotografia como imagem de fundo, tornando o dispositivo ainda mais pessoal. De salientar que os programadores terão que ter o cuidado de lançar as aplicações com os mosaicos (tiles) sem cor de fundo para que esta imagem seja visível.

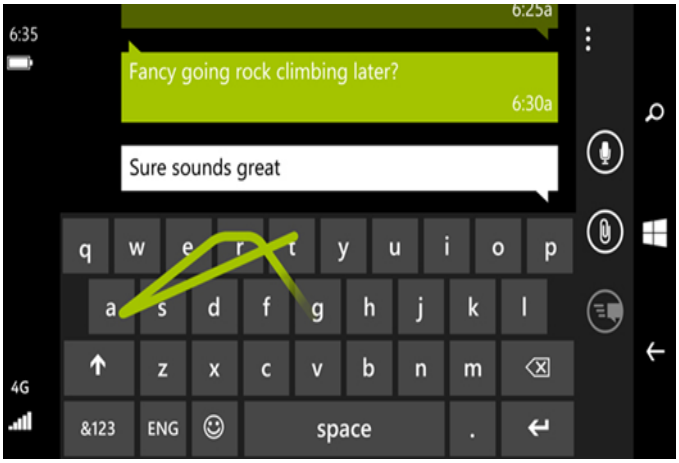
Cortana foi a funcionalidade mais falada no evento, redes socais e noticias, e é a nova assistente digital da nova versão do Windows Phone com base no Bing. Esta assistente começa por aprender tudo sobre o utilizador, guardando toda a informação num "bloco de notas", é possível interagir verbalmente ou por escrito. Tem a capacidade de responder às questões mais mirabolantes do dia-a-dia, e tem a capacidade de analisar os emails do próprio dispositivo (depois de previa autorização do utilizador) e com base nisso ajudar o utilizador no seu dia-a-dia. Esta funcionalidade é

extensível e poderá interagir com outras aplicações, como por exemplo o Skype, Facebook ou Twitter.



Esta funcionalidade só ficará disponível em Portugal no ano de 2015. E será disponibilizada brevemente nos USA como versão beta e no segundo semestre de 2014 será publicado nos USA, UK e China.

É introduzido um novo teclado inteligente, Word Flow Keyboard, que permite descobrir qual a palavra que estás a escrever e com isso diminuir o tempo necessário para escrever uma mensagem. E quanto mais se escrever usando este teclado, mais ele saberá sobre o utilizador e assim irá "adivinhar" mais rapidamente a palavra que se vai escrever. Atualmente é suportado por 16 línguas.



A aplicação calendário do Windows Phone foi redesenhada e agora apresenta uma nova vista semanal que já há algum tempo era requisitada pelos utilizadores, tendo agora um aspeto semelhante ao Outlook para PC e tem integrado a informação sobre o tempo. É possível sincronizar com o calendário do Google e ter múltiplos calendários.

A aplicação Fotografia & Câmara apresenta as fotografias mais recentes, organizando-as por data e localização e ao nível da Câmara apresenta melhorias ao nível da interface com o utilizador.

A aplicação Pessoas passa a estar diretamente ligada a outras aplicações relacionadas com redes sociais, permitindo assim acesso a todas as funcionalidades das aplicações instaladas.

A aplicação Loja apresenta uma nova organização na apresentação das aplicações e apresenta sugestões consoante a localização. É possível definir atualizações automáticas das aplicações ou apenas quando se está ligado ao Wi-Fi, instalar aplicações ou jogos em cartões SD e caso o programador permita será possível fazer cópias de segurança (backups) das aplicações e dos jogos na OneDrive. De referir, que todas as aplicações de Windows Phone 7.8 que já correm no Windows Phone 8.0 também irão correr em Windows Phone 8.1 e o mesmo acontece com as aplicações de Windows Phone 8.0. No entanto, as aplicações Windows Phone 8.1 não irão correr em dispositivos com Windows Phone 8.0 até estes ser atualizados para a versão Windows Phone 8.1.

Ao nível de envio de email, vai ser possível enviar os mesmos encriptados e será a aplicação nativa de email que terá esta funcionalidade. Os documentos do office poderão vir a ser protegido por password e será possível aceder a documentos corporativos usando uma ligação por VPN.

O utilizador vai ter a possibilidade de obter mais feedback no uso do seu dispositivo através do Data Sense, Wi-Fi Sense,

Storage Sense, and Battery Saver, salientando que este irá ter acesso à informação de quais as aplicações que estão a consumir mais recursos. Poderá editar a sua playlist e gerir a sua coleção online de forma fácil e eficiente, e poderá comprar ou alugar vídeos através do Xbox Music e finalmente é possível subscrever podcasts através de RSS feeds ou através de pesquisa no Bing em todos os países em que o Windows Phone está disponível.

Esta versão do Windows Phone tem a aplicação do Skype integrada com o sistema de chamadas permitindo assim que o utilizador mude de uma chamada normal para uma chamada de vídeo no Skype.

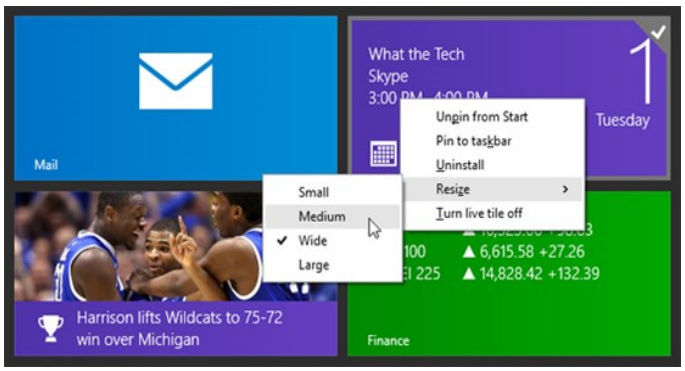
O Internet Explorer 11 é disponibilizado no Windows Phone 8.1 e permite ver todos os separadores abertos, assim como os favoritos. Tem ainda as funcionalidades de guardar e lembrar as passwords do utilizador, um modo de leitura de escrita manual, fazer o upload de documentos e suporte para WebGL.

Por fim, termino com uma nova funcionalidade, que já há muito era requisitada por muitos, e que apenas era possível "dentro" da Microsoft, finalmente é possível projetar o conteúdo do Windows Phone num PC, Tablet usando um cabo USB ou através da TV (com suporte de Wireless Miracast).

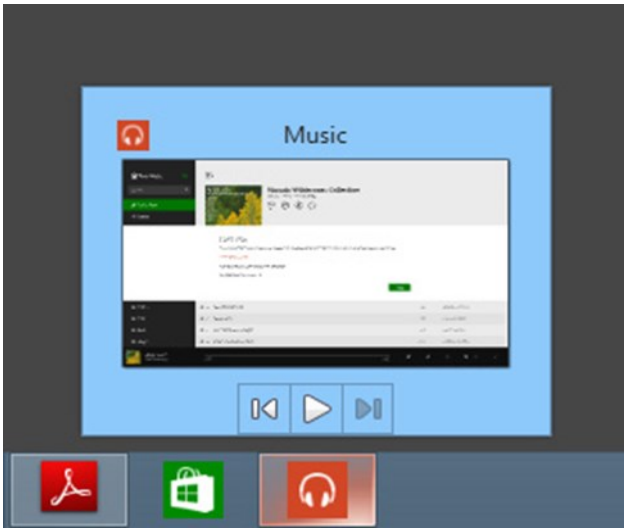
Os utilizadores que sejam detentores de conta de programador e que queriam ter acesso à versão Windows Phone 8.1 Preview (disponível apartir de 10 de Abril) consultem o seguinte artigo: Windows Phone Preview for Developers.

Atualização do Windows 8.1

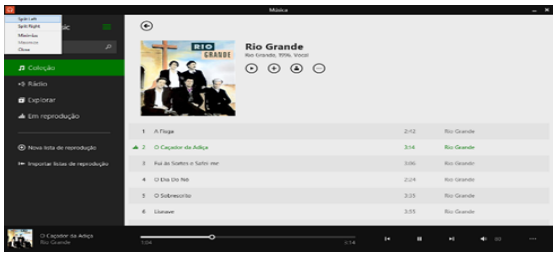
A atualização do Windows 8.1 é focada na utilização do rato e teclado, permitindo ao utilizador ter uma melhor experiência de utilização, isto é, quando estamos perante o ecrã principal e estamos perante uma utilização de rato e teclado é possível clicar no mosaico com o botão do lado direito e assim abrir um menu de contexto que nos possibilita: remover do ecrã principal, fixar na barra de tarefas, desinstalar a aplicação, mudar o tamanho do mosaico e ligar/desligar as notificações da aplicação. Desta forma a "appbar" não irá aparecer como acontecia antes, note-se que para uma utilização com *touch* a usabilidade mantém-se.



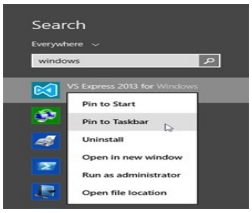
As aplicações modernas passam a estar acessível a partir do desktop, isto é, é possível afixar aplicações modernas na barra de tarefas e quando estas estão a correr a aplicação fica disponível também na barra de tarefas permitindo ao utilizar uma rápida troca entre as várias aplicações, assim como uma fácil interação com as aplicações, é o caso da aplicação Música que apresenta um "quick player".

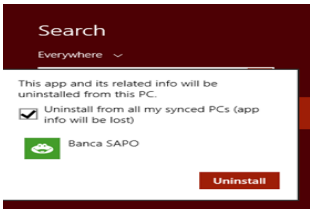


Perante uma utilização de rato e teclado passamos a ter disponível uma barra no topo, que aparece quando o rato se aproxima do topo do ecrã, permitindo ao utilizador minimizar, fechar ou então definir que quer a aplicação aberta do lado esquerdo ou do lado direito do ecrã (split left/split right), opção esta que surge no lado superior esquerdo.

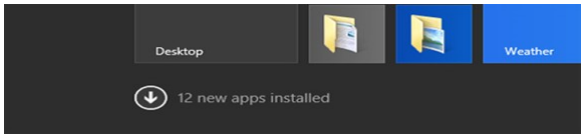


A pesquisa do Windows passa a estar integrada com a Loja, permitindo que as pesquisas procurem por aplicações que satisfaça o termo inserido. Se no resultado da pesquisa aparecer uma aplicação que pretendemos desinstalar, é possível faze-lo através do menu de contexto e é possível definir se pretende remover a aplicação de todos os PC sincronizados pela atual conta.

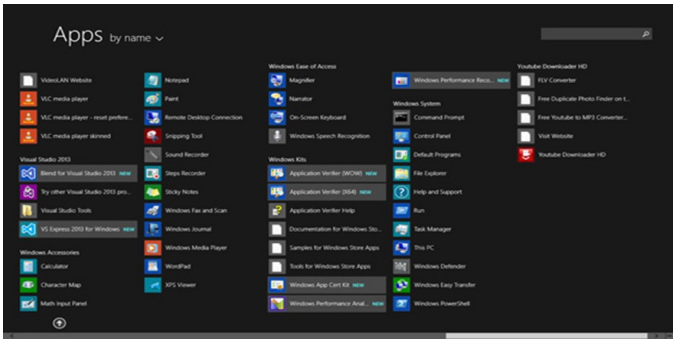




Passamos a ter notificação de quantas aplicações novas foram instaladas e quando consultamos a lista de aplicações instaladas, as aplicações novas aparece destacadas para que o utilizador consiga se aperceber quais são.



Os PC Settings passam a estar disponíveis a partir de um mosaico, facilitando assim o acesso às definições e passa a ter disponível o acesso ao CPanel Control no menu, opção que anteriormente só era possível aceder a partir das definições do Desktop. Outra novidade é a possibilidade de ver o espaço ocupado em disco e a possibilidade de consultar o espaço ocupado por cada aplicação moderna.



Na Keynote do //Build/ foi apresentado o regresso do botão iniciar com uma interface que inclui "Live Tiles" e foi ainda apresentado aplicações modernas a correr dentro de janelas, no entanto estas novidades ainda não estão disponíveis nesta atualização.

A atualização do Windows 8.1 está disponível via MSDN a partir do dia 2 de Abril e apartir do dia 8 de Abril para o público em geral através do Windows Update.

A conta Microsoft vai permitir que um Windows Phone 8.1 e um Windows 8.1 estejam sempre sincronizados de forma a que se alterar o tema do Windows Phone 8.1 esta alteração se irá refletir no Windows 8.1. Ao nível da Loja e do consumo de aplicações o utilizador apenas irá pagar apenas uma vez, isto é, ao comprar uma aplicação na Loja do Windows Phone o utilizador terá acesso à mesma aplicação na Loja Windows sem ter que pagar mais por isso, o contrário é recíproco. Outras definições como por exemplo o acesso ao Wi-Fi também será sincronizado entre dispositivos.

Por fim, gostaria de sugerir algumas demonstrações disponíveis no Youtube pela equipa de Windows Phone e do Windows:

- Introducing Windows Phone 8.1
- Live Demo: New Windows Phone Start Screen and Cortana
- Meet Cortana: The New Windows Phone 8.1 Personal Assistant
- Introducing the Windows 8.1 Update

Para quem quiser ler mais notícias sobre este tema consultem as seguintes referências:

- Novidades sobre a atualização do Windows 8.1 e Windows Phone 8.1 (Em Português)
- Windows Phone 8.1 and Windows 8.1 News (Em inglês)

Em conclusão, o Windows Phone 8.1 apresenta-se com mais funcionalidades que permite uma experiência mais rica e pessoal, e a convergir para uma utilização mais global uma vez que a cada vez mais o Windows Phone e o Windows estão sincronizados entre si. Por sua vez a atualização do Windows 8.1 vem facilitar a vida ao utilizador que diariamente usa rato e teclado, aumentando assim a usabilidade do próprio SO.

AUTOR

Escrito Por Sara Silva

Licenciada em Matemática – Especialidade em Computação, pela Universidade de Coimbra e é Microsoft Certified Profissional Developer. Atualmente o seu foco de desenvolvimento incide em Windows Phone e Windows 8 Store Apps. O seu Blog **é** www.saramgsilva.com e o twitter é @saramgsilva.

Elege o melhor artigo desta edição

Revista PROGRAMAR

http://bit.do/ProgramarED45_V

Veja também as edições anteriores da Revista PROGRAMAR

44ª Edição - Fevereiro 2014

43ª Edição - Dezembro 2013

42ª Edição - Setembro 2013

41ª Edição - Junho 2013

40ª Edição - Abril 2013

39ª Edição - Fevereiro 2013

e muito mais em ...

www.revista-programar.info

DUVIDAS?

IDEIAS?

AJUDAS?

PROJECTOS?

portugal-a-programar.org