

PROGRAMAR

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

EDIÇÃO #41 - JUNHO 2013

ISSN 1647-0710



COLONAS

OBSERVABLE VS. TASK **C#**

OPERADOR OVERLOADING **VISUAL(NOT)BASIC**

A PROGRAMAR

PROGRAMAÇÃO EM C CONST

JBOSS APPLICATION SERVER 7

COMUNIDADES

TELERIK RADCONTROLS **NETPONTO**

ANÁLISES

ANDROID INTRODUÇÃO AO DESENVOLVIMENTO DE APLICAÇÕES

NO CODE

GAME SALAD
PROJECTO EM DESTAQUE NA COMUNIDADE POP **LITTLE BITS**

EQUIPA PROGRAMAR

Coordenador
António Santos

Editor
António Santos

Design
Sérgio Alves
Twitter: [@scorpion_blood](https://twitter.com/scorpion_blood)

Redacção
António Pedro Cunha
José Marques
Maurício Magnani Jr
Paulo Morgado
Ricardo Perre
Rita Peres
Sara Silva
Sérgio Ribeiro

Staff
Ana Barbosa
António Pedro Cunha
António Santos
António Silva
Fábio Domingos
Jorge Paulino
Sara Santos

Contacto
revistaprogramar@portugal-a-programar.org

Website
<http://www.revista-programar.info>

ISSN
1 647-071 0

JMP EAX

Mais uma edição, desta feita a 41ª edição da Revista, que trazemos até vós.

Muito aconteceu desde as edições anteriores, e muito irá acontecer, pois a tecnologia está em constante evolução e a programação em constante desenvolvimento.

No passado dia 25 de Maio realizou-se no Auditório da Microsoft em Lisboa o primeiro Evento Presencial da Comunidade Portugal-a-Programar, sobre o qual poderão encontrar um artigo mais à frente na revista. Em parte por causa de todo o esforço necessário para que se realiza-se o evento, aconteceram alguns atrasos no lançamento desta edição, pelo que peço desculpas a todos os leitores, autores, colaboradores e parceiros que nos têm acompanhado, ao longo destes anos que a Revista PROGRAMAR, já “carrega nos ombros”.

Ainda na sequência do evento, no qual a revista também esteve representada, decorre um concurso de aplicações para Windows 8, pelo que se tem uma ideia e consegue fazer a aplicação, recomendamos que leia a página sobre o concurso, que poderá encontrar nesta edição!

Num outro registo, congratulo-me pelo crescimento da revista e agradeço a todos quantos nela trabalham e colaboram com o seu crescimento! Sem vocês seria impossível, mas não me posso esquecer de todos os leitores, pois é para vós que a revista é feita! É para os leitores que bimestralmente, uma equipa dedicada de autores, revisores, designers e demais staff, traz uma nova edição, com novos artigos, novos temas, tentando ir sempre de encontro ao que esperam encontrar nesta publicação.

Nesta edição damos continuidade a algumas séries de artigos de autores que são verdadeiros “residentes”, e introduzimos novos temas e tecnologias, bem como uma nova série de artigos, no sentido de vos agradar mais e melhor.

Para aqueles que de certa forma foram privilegiados ao ter vivido na geração das 5.25” e até antes com as cassetes e os 8 bits que faziam maravilhas, esperamos continuar a trazer-vos um pouco daquilo que eram as publicações de programação que fizeram de vós e alguns de nós, os programadores que agora somos, mas com os conteúdos actualizados e de acordo com as vossas expectativas. Aos demais leitores que não viveram essa “mágica experiência” de fazer muito com pouco e copiar código do papel para o computador, digitando cada instrução como se fosse uma “palavra mágica”, para no fim se mandar compilar ou interpretar e ver “magia” acontecer, espero que a revista continue a ir de encontro aquilo que vos agrada, sempre com a isenção que a caracteriza, e a pluralidade pela qual se pauta.

Até à próxima edição, *jmp EAX*.

António Santos

A revista PROGRAMAR é um projecto voluntário sem fins lucrativos. Todos os artigos são da responsabilidade dos autores, não podendo a revista ou a comunidade ser responsável por alguma imprecisão ou erro.

Para qualquer dúvida ou esclarecimento poderá sempre contactar-nos.

TEMA DE CAPA

- [7](#) Introdução ao Java EE e Java Web (**José Marques**)

A PROGRAMAR

- [22](#) Programação em C: const (**António Pedro Cunha**)
- [24](#) Jogo da Vida (**Rita Peres**)
- [34](#) JBoss Application Server 7 (**Mauricio Magnani Jr**)
- [34](#) PHP: Uma framework “from scratch” (Parte 1) (**Ricardo Perre**)

COLUNAS

- [39](#) **C#** - Observable vs. Task (**Paulo Morgado**)
- [42](#) **Visual(not)Basic** - Operator Overloading (**Sérgio Ribeiro**)

EVENTOS

- [46](#) | **Evento Presencial Programar**

ANÁLISES

- [48](#) Android – Introdução ao Desenvolvimento de Aplicações (**José Marques**)
- [49](#) C# 5.0 com Visual Studio 2012 (**Ricardo Perre**)

COMUNIDADES

- [52](#) NetPonto - Telerik RadControls - Rápida implementação da página Sobre para Windows Phone Apps (**Sara Silva**)

NO CODE

- [65](#) Game Salad (**Rita Peres**)
- [74](#) Projecto em Destaque na Comunidade P@P: Little Bits

EVENTOS

20 de Junho	XXXVI Encontro da Comunidade SQLPort
22 de Junho	39ª Reunião Presencial da Comunidade NetPonto em Lisboa
21 e 22 Junho	Agile & Scrum Portugal 2013
11 a 22 Junho	CISTI 2013 - 8ª Conferência Ibérica de Sistemas e Tecnologias de Informação

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

ARM desenvolve chip anti-pirataria

A ARM anunciou o lançamento de um chip anti-pirataria que vai permitir controlar a utilização indevida de conteúdos protegidos por direitos de autor.

A novidade é o resultado de uma parceria com Hollywood, que pretende prevenir a pirataria nos conteúdos de vídeo em alta definição numa próxima geração de smartphones e tablets. Vai agir de forma complementar às tecnologias de Digital Rights Management, ao nível do hardware.

O Mali V500 vai tirar partido da tecnologia TrustZone que a fabricante britânica usa já há vários anos como mecanismo de segurança para transações e software, mas que aqui será utilizada pela primeira vez com o objetivo de prevenir a utilização de conteúdos protegidos por direitos de autor.

A disponibilização de conteúdos multimédia de alta definição em sistemas móveis tem sido endereçada com algumas reticências pelos detentores de direitos sobre esses conteúdos, que não encontram grandes mecanismos de segurança nessas plataformas, sobretudo no que se refere aos sistemas operativos móveis de código aberto como é o caso do Android.

Citada pelo Financial Times, a ARM defende que a sua tecnologia será especialmente útil em países como a China, onde a indústria consegue fazer um nível mínimo de receitas com os filmes depois destes saírem das salas de cinema.

Na Computex a ARM lançou ainda o Cortex A12, um processador dirigido ao mercado de gama média desenvolvido a 28 nanómetros. A fabricante garante que o chip assegura melhorias de 40% ao nível da performance face ao antecessor Cortex A9.

Também a Intel aproveitou a feira asiática para apresentar uma nova geração de processadores Core i3, i5 e i7. A eficiência energética, garante a fabricante, é o incremento mais relevante nesta nova geração de chips.

Escrito ao abrigo do novo Acordo Ortográfico

Fonte: Tek Sapo

JP Inspiring Knowledge cria kit portátil de ensino para professores

A tecnológica portuguesa quer democratizar o ensino e coloca a figura do professor como agente principal do objetivo. O Kit Inspired Teacher é uma escola ambulante.

A JP Inspiring Knowledge, empresa responsável pelos computadores Magalhães, está a lançar em vários mercados internacionais um kit portátil de ensino que se destina aos profes-

sores. O conjunto é composto por um computador, por um rato com capacidade de digitalização, por um projetor de pequenas dimensões e por um quadro digital que torna qualquer superfície plana num ambiente de trabalho.

Todos os dispositivos podem ser guardados numa mala trolley resistente a choques e a água, sistema que reforça o carácter de mobilidade do kit de ensino.



O Kit Inspired Teacher tenta conjugar o fator prático ao lado económico. Mesmo sem revelar o preço do conjunto, o pacote de "ensino digital" deve ficar mais barato do que montar uma estrutura semelhante a uma sala de aula com todos os componentes como um PC, projetor e quadro.

A ideia da tecnológica portuguesa é diminuir ao máximo o número de casos de crianças que não têm acesso ao ensino pela falta de material escolar qualificado e atualizado. O professor é ao mesmo tempo a sala de aula, ideia que tem sido bem recebida internacionalmente.

"O conceito já foi apresentado em vários países, com uma resposta muito positiva, sobretudo do público-alvo principal: os professores. Agora, com o lançamento desta solução no mercado, estamos em condições de levar a educação digital a qualquer parte do mundo", escreve em comunicado o presidente da JP Inspiring Knowledge, Jorge Sá Couto.

A empresa responsável pela gama de equipamentos Magalhães não vai disponibilizar o kit em Portugal e a pedido do TeK também não adiantou uma estimativa para o volume de vendas até ao final do ano.

Escrito ao abrigo do novo Acordo Ortográfico

Fonte: Tek Sapo

Portugal acolhe rede internacional de centros de inclusão social

O Centro de Inclusão Digital, um projeto de intervenção social apoiado pela Microsoft, está a chegar a Portugal. No primeiro ano de atividade a organização quer chegar a 1.200 jovens.

A ação do CDI centra-se no desenvolvimento de competências tecnológicas, junto de populações provenientes de contextos socioeconómicos desfavorecidos com o objetivo de criar condições de empregabilidade.

A organização, fundada no Brasil em 1995, também atua apoiando iniciativas empresariais empreendedoras. Em Portugal pretende criar uma rede nacional de centros de inclusão digital. O primeiro será instalado no bairro da Belavista, em Setúbal. Uma segunda estrutura está também já prevista e vai localizar-se em Vale de Cambra, no Porto.

O CDI conta com 780 espaços de inclusão social em 12 países. De acordo com os números avançados pela organização as suas iniciativas já tiveram impacto na vida de um milhão de pessoas.

A iniciativa conta com o apoio da Microsoft que hoje a apresentou no âmbito do 7º Encontro de Parceiros Sociais e que doou 100 mil dólares à instituição para o arranque do projeto.

Escrito ao abrigo do novo Acordo Ortográfico

Fonte: Tek Sapo

Android Studio: An IDE built for Android

Hoje na Google I / O, anunciámos uma nova IDE que é construída com as necessidades dos programadores de Android em mente. É chamado Android Estúdio, é gratuito, e está agora disponível para você experimentar como uma pré-visualização de acesso antecipado.

Para desenvolver o Estúdio Android, colaborámos com JetBrains, os criadores de uma das mais avançadas IDEs Java disponíveis hoje. Baseado no poderoso e extensível IntelliJ IDEA Community Edition, adicionámos recursos que são projetados especificamente para o desenvolvimento Android, que simplificam e otimizam o seu fluxo de trabalho diário.



Ferramentas de construção extensíveis

Sabemos que é preciso construir um sistema que se adapte aos seus requisitos de projeto, mas que se estenda ainda ao seu ambiente maior de desenvolvimento. O Android Studio usa um novo sistema de construção baseado em Gradle que proporciona flexibilidade, sabores de construção personalizados, resolução de dependências, e muito mais. Este novo sistema de montagem permite-lhe construir os seus projetos no IDE, bem como em seus servidores de integração contínua. A combinação permite-lhe gerir facilmente as configurações complexas de construção nativamente, através do seu fluxo de trabalho, em todas as suas ferramentas. Confira a documentação de pré-visualização para ter uma ideia melhor do que o novo sistema de construção pode fazer.



Edição de código poderoso

O Android Studio inclui um editor de código poderoso. Ele é baseado no editor IntelliJ IDEA, que suporta recursos como edição inteligente, refatoração de código avançado e análise aprofundada de código estático.

Os recursos de edição inteligentes, tais como pesquisas de recursos em linha facilitam a leitura do código, dando-lhe acesso instantâneo a editar o código dos recursos de apoio. A refatoração avançada de código dá-lhe o poder de transformar seu código através de todo o projeto, com rapidez e segurança.

Adicionámos análise de código estático para o desenvolvimento Android, ajudando a identificação de erros mais rapidamente. Além das centenas de inspeções de código que o IntelliJ IDEA oferece, adicionámos inspeções personalizadas. Por exemplo, nós adicionamos metadados para as APIs do Android, que assinalam quais os métodos que podem retornar nulos ou não, que constantes são permitidas para que métodos, e assim por diante. Android Studio usa esses dados para analisar seu código e encontrar possíveis erros.

Fonte: Android Developers Blog

TEMA DE CAPA

Introdução ao Java EE e Java Web

Introdução ao Java EE e Java Web

Java Enterprise Edition

A plataforma Java Enterprise Edition ou Java EE é uma plataforma da Oracle que fornece uma API adicional ao Java SE para o desenvolvimento e execução de aplicações java do tipo empresarial focadas nos serviços de rede e Web.

Vindo do Java Professional Edition, o nome que foi anunciado em Maio de 98, foi a partir dos finais de 1999 que ficou com o nome de Java EE incluindo tecnologias como Java Database Connectivity (JDBC), Java Servlet, JavaServer Pages (JSP), Enterprise JavaBeans, JavaMail, etc.

CONCEITOS

Aplicação Web Java

Uma aplicação Java Web gera páginas Web interactivas contendo vários tipos de markup languages (HTML, XML, etc.) e conteúdo dinâmico. É tipicamente composto por elementos Web tais como JavaServer Pages (JSP), Servlets e JavaBeans para modificar e temporariamente armazenar dados, interagir com bases de dados e serviços Web gerando conteúdo em resposta aos pedidos do cliente.

JSP (JavaServer Pages)

JavaServer Pages é uma tecnologia usada no desenvolvimento de aplicações Web Java semelhante ao PHP. Possui compatibilidades com outras soluções Java como a Servlet, corre em servidores como o Apache Tomcat e o GlassFish

Ao contrário dos típicos jar's os arquivos deste tipo de implementações são *.ear ou *.war (Web Archive).

As páginas Web dinâmicas java, aceitam HTML, CSS, Javascript, tudo que possam incluir no um ficheiro normal HTML, não necessitam de compilação prévia isso fica ao cargo do servidor Web de traduzir as linhas de código em Java para o browser compreender.

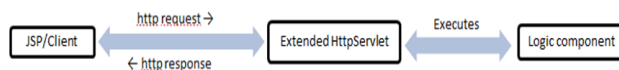
Qualquer código Java inserido nos ficheiros JSP deve ficar entre as tags:

- `<% *Code* %>` (para inserção de código)
- `<%=var %>` (para a expressão de uma variável)

- `<%@page %>` (para importar, `@include` para incluir uma página JSP ou HTML externa)

Servlet

É uma tecnologia Java que permite gerar dados em HTML e XML. Incluída na biblioteca `javax.servlet`, permite processar pedidos e respostas vindas de JSP e serve como uma extensão do servidor Web. Neste caso vamos usar a `HttpServlet` onde a sua função será processar pedidos das JSP resolver os pedidos e enviar uma resposta.



XML

XML (Extensible Markup Language) subtipo da SGML, foi criada pela W3C com o objectivo de ser usado como formato de configuração entre várias linguagens através da internet.

Ex.:

```
<?xml version="1.0" encoding="iso-8859-1"
      encoding="iso-8859-1"?>
<receita nome="pão" tempo_de_preparacao="5 min
      tempo_de_cozedura="1hora">
  <ingredientes>
    <ingrediente quantidade="4"
      unidade="chavenas">Farinha</ingrediente>
    <ingrediente quantidade="7"
      unidade="gramas">Fermento</ingrediente>
    <ingrediente quantidade="1.5"
      unidade="chavenas">Agua</ingrediente>
    <ingrediente quantidade="1"
      unidade="colheres">Sal</ingrediente>
  </ingredientes>
  <instrucoes>
    <passo>Misturar tdos os ingredientes.</passo>
    <passo>Cobrir com um pano e deixar repousar
      uma hora à temperatura ambiente</passo>
    <passo>Misturar novamente e colocar ao forno
      até ao fim da cozedura</passo>
  </instrucoes>
</receita>
```

Neste caso em particular, o ficheiro XML que vamos criar/ editar, na configuração de uma aplicação Java Web, é o **web.xml** que tem como *tags* mais usa das as seguintes:

- `<? ?>` à define comentários
- `<web-app id version xmlns >` à define o tipo do XML, a versão...
- `<display-name>` à define o nome
- `<servlet>` à define uma servlet

TEMA DA CAPA

INTRODUÇÃO AO JAVA EE E JAVA WEB

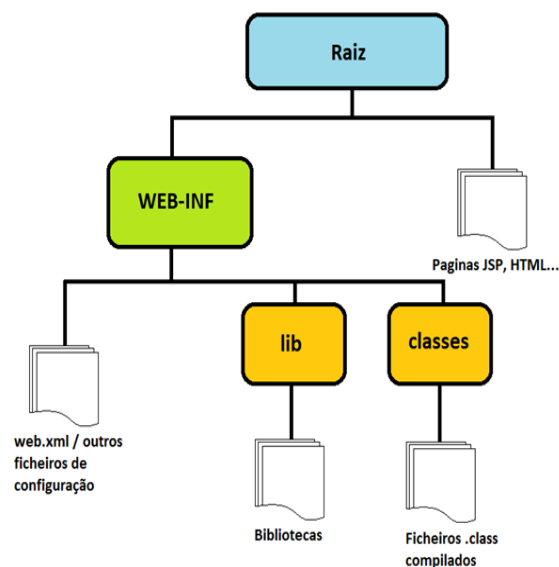
- `<servlet-name>` à define o nome da *servlet*
- `<servlet-class>` à aponta o caminho da *servlet* na aplicação
- `<load-on-startup>` à quando o valor for 1 a *servlet* será inicializada no início da aplicação
- `<servlet-mapping>` à caso a *servlet* efectue operações requeridas por elementos externos
- `<servlet-name>` à define o nome terá de ser o mesmo que se encontra na tag `<servlet>`
- `<url-pattern>` à define um url para a *servlet*
- `<filter>` à semelhante à tag `<servlet>` com a diferença de que esta permite enquadrar uma classe de filtro na aplicação que permita a existência de sessões na aplicação Web ou qualquer segmento de código que necessite de ser constantemente executado
- `<filter-name>` à define o nome da classe filtro
- `<filter-class>` à aponta o caminho da class na aplicação
- `<filter-mapping>` à sempre necessário definir para uma classe filtro
- `<filter-name>` à define o nome terá de ser o mesmo que se encontra na tag `<filter>`
- `<url-pattern>` à para possuir o efeito de filtro o url será todos os existentes ou seja `/*`
- `<error-page>` à tag existente para definir páginas de erro que sejam geradas pelo web server e que possam ocorrer na aplicação
- `<error-code>` à define o tipo de código de erro gerado pelo web server (404-page not found, 500-internal server error...)
- `<location>` à localiza a página de erro personalizada dentro da aplicação.

Estrutura de uma aplicação Web Java

O exemplo que vai ser aqui apresentado será montado num Web Archive (ficheiro *.war) que poderá depois ser colocado num servidor Web sendo os mais conhecidos o Apache Tomcat e o Oracle Glassfish. Ambos disponíveis gratuitamente e no pacote de Java EE do IDE NetBeans que os instala em modo desenvolvimento na máquina local.

Nota: As aplicações necessárias para a criação deste exemplo são o Java Runtime Environment (JRE), o Java Development Kit (JDK) e o IDE NetBeans.

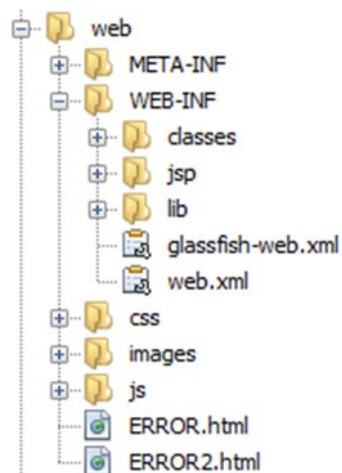
A estrutura normal de uma aplicação java desta natureza tem, por norma, a seguinte estrutura.



O ficheiro WAR criado, se o abrirmos, tem uma estrutura de pastas/ficheiros orientada ao modelo de desenvolvimento MVC (Model-view-controller) onde a View está incluída diretamente na raiz do projeto onde se pode encontrar as páginas JSP e HTML acompanhadas pelos devidos ficheiros JavaScript e CSS necessários. Caso haja ausência de um Deployment Descriptor (ficheiro de configuração referente a uma determinada aplicação que neste caso será o web.xml) o servidor Web procura sempre por um index.jsp como página inicial.

Dentro da pasta WEB-INF estão todas as classes, bibliotecas, ficheiros de configuração e servlets necessárias para processar a informação mostrada pelas JSPs.

Em suma ter-se-á então algo deste género:



TEMA DA CAPA

INTRODUÇÃO AO JAVA EE E JAVA WEB

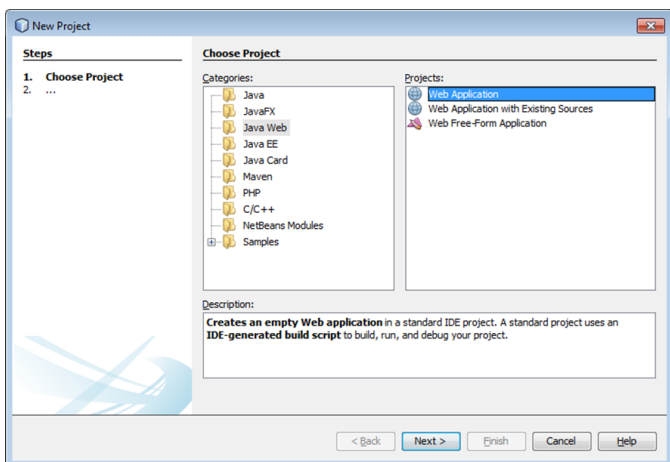
Nota: Nesta estrutura optou-se por colocar as páginas dinâmicas debaixo do WEB-INF pois uma vez que podemos usar as servlets para controlar o que se deve ou não ser mostrado perante os pedidos vindos do cliente. Isto torna a aplicação mais segura na questão de mapeamento de URLs.

Desenvolvimento

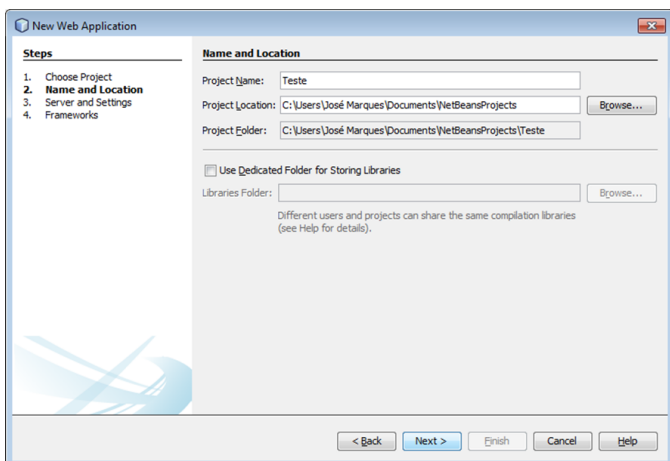
Neste exemplo vai-se usar o IDE NetBeans pois é aquele que a Oracle recomenda, encontra-se disponível para Windows, Linux, Solaris e Mac OS X sendo mais simples de configurar. O NetBeans encontra-se disponível para download no próprio site e no site da Oracle como complemento do Java Development Kit 7 (Ver bibliografia).

Primeiros passos

Para dar início à construção da nossa aplicação Java Web no NetBeans basta ir a **File – New Project** e aparece-nos a seguinte janela.

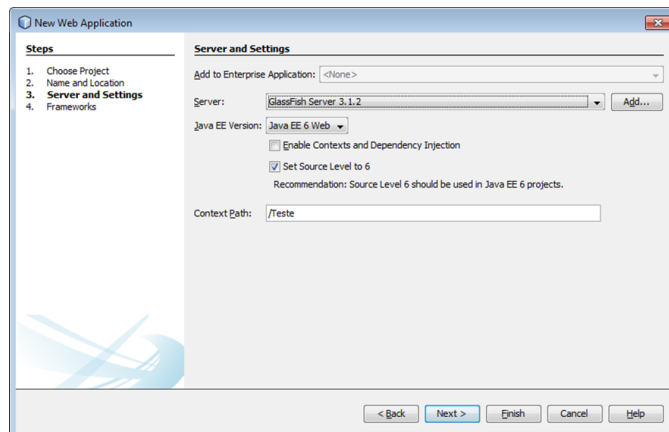


E vamos selecionar **Java Web – Web Application**. Em seguida definimos a localização e o nome da nossa aplicação.



Dando o nome de **Teste** a esta aplicação, no passo seguinte o nome que foi atribuído é automaticamente dado como

Context Path, ou seja, a aplicação irá ser identificada no *Deployment descriptor* como **Teste** e uma vez colocada no servidor terá o seguinte URL: <http://hostname:port/Teste>.

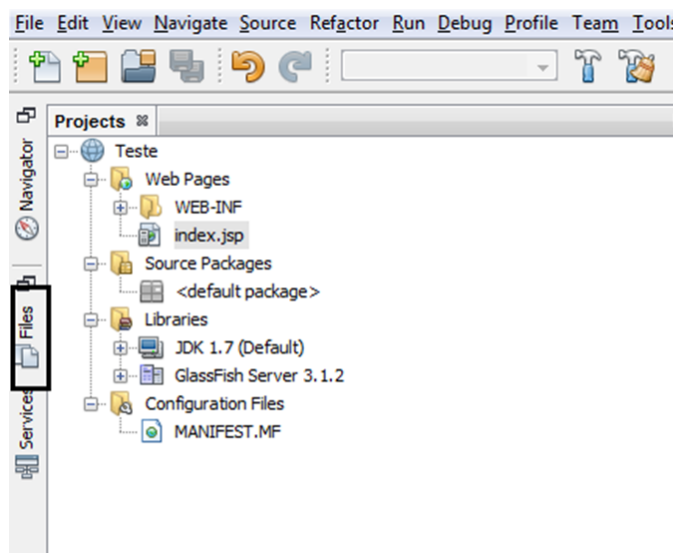


Neste exemplo vamos usar para o servidor GlassFish da Oracle, a versão 6 do Java EE e o IDE Netbeans.

O servidor escolhido pode ser posteriormente alterado caso queira-se testar a aplicação noutros ambientes, contudo é desencorajado a alteração da versão sem primeiro verificar a existência de possíveis incompatibilidades com código e bibliotecas que estejam a ser usadas. O último será ignorado uma vez que este exemplo que vamos criar não vai possuir nenhum framework.

Uma vez concluído todos os passos verificamos que automaticamente o IDE criou-nos uma estrutura com uma página *index.jsp* na raiz, uma pasta *WEB-INF* vazia e atalhos para a pasta de sources, bibliotecas e de ficheiros de configuração com o tradicional *MANIFEST.MF* criado.

A estrutura visível não corresponde à estrutura real de diretorias para se visualizar isso basta ir ao separador *Files* situado é esquerda, como mostra a imagem seguinte.

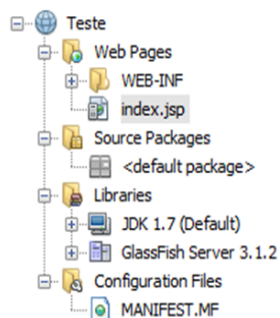


TEMA DA CAPA

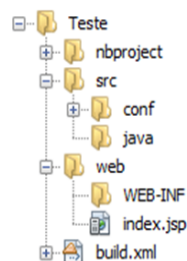
INTRODUÇÃO AO JAVA EE E JAVA WEB

Caso não esteja criado o atalho pode-se sempre aceder indo à opção **Window – File** abrindo o separador e recolocando o atalho.

Estrutura do Projecto no NetBeans



Estrutura de directorias criadas



Numa breve comparação pode-se verificar que o NetBeans cria determinados ficheiros e directorias referentes ao projeto (bluid.xml e nbproject) e onde na estrutura do projeto a pasta Web Pages tem referência à pasta web e a Source Packages tem referência à pasta src. Todos os ficheiros de configuração ficam dentro da pasta WEB-INF e no caso da pasta Libraries o seu conteúdo é um conjunto de atalhos cuja sua localização pode ser referenciada a quaisquer pastas externas ao projeto. No entanto recomenda-se criar uma pasta lib debaixo de WEB-INF com as bibliotecas a ser usadas pela aplicação, para evitar erros originados por utilização do mesmo ficheiro de biblioteca por mais do que um projeto e para facilitar a passagem do projeto para outro ambiente ou máquina pois assim basta só passar a pasta com o projeto sem haver preocupações com importação de bibliotecas.

Criação de uma JSP

Se abrirmos a página JSP criada pelo IDE ela vai ter um aspecto semelhante a este:

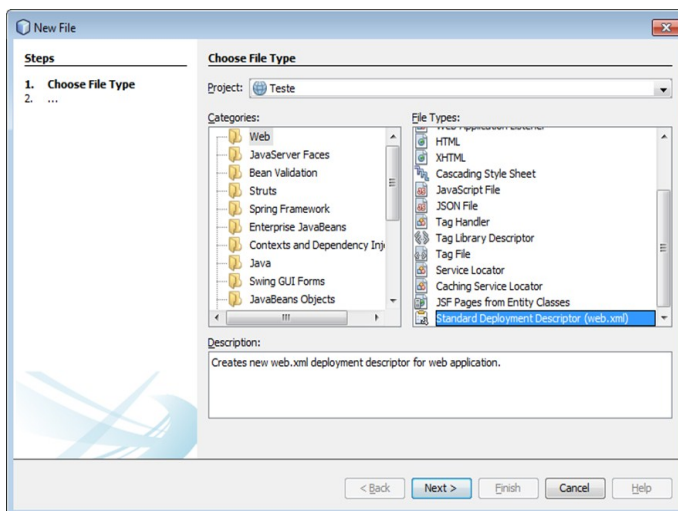
```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type"
          content="text/html; charset=UTF-8">
    <title>JSP Page</title>
  </head>
  <body>
    <h1>Hello World!</h1>
  </body>
</html>
```

Uma vez que esta página tem o nome de **index** caso seja feito o *deploy* da aplicação a mesma vai apresentar uma página de fundo branco com o título **JSP Page** e a frase **Hello World!**

Para realizar o *deploy* da aplicação basta carregar sobre o botão de play situado no painel de topo do IDE Netbeans ou então ir a **Run – Run Project**.

Caso o nome da página seja alterado e não exista nenhuma página com o nome **index** o servidor não consegue encontrar nenhuma página inicial. Então em seguida vamos alterar o nome da nossa página inicial para **home.jsp** e vamos configurar o nosso **web.xml** de maneira a criarmos um filtro de sessão que vai definir que o servidor no arranque da mesma redirecionar-se para uma determinada *servlet*.

Para tal se formos à opção de **File – New File...** na secção Web selecionar o tipo de ficheiro **Standard Development Descriptor (web.xml)**



Uma vez criado o ficheiro terá o seguinte conteúdo por defeito.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
         version="3.0">
  <session-config>
    <session-timeout>
      30
    </session-timeout>
  </session-config>
</web-app>
```

Temos agora o ficheiro **web.xml** onde poderemos colocar as nossas referências às *servlets* que vamos criar atribuindo-lhes um *url pattern* a fim de poderem ser acedidas.

Para não termos trabalho o IDE por si só resolve todas as *tags* que necessitamos de colocar no **web.xml** na altura em que criamos uma *servlet*, caso não seja resolvido pelo IDE. Para mapearmos a *servlet* **WelcomeServlet** no *package* main.servlet com o *url pattern* **Welcome** o que deveria de ser colocado seria algo deste género (ver as *tags* fornecidas):

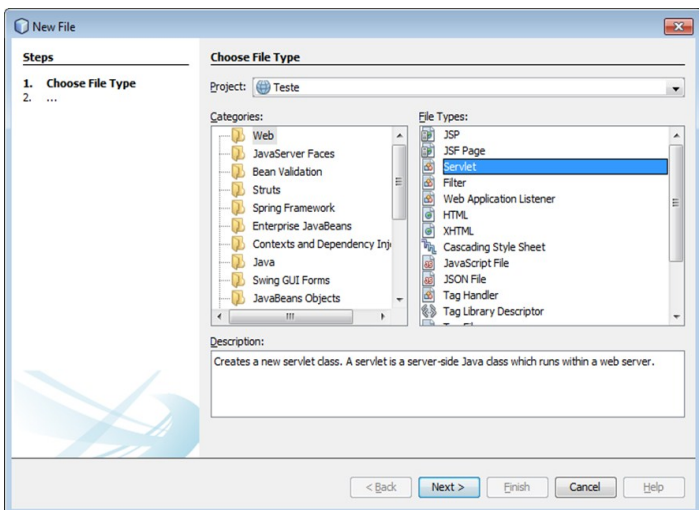
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="3.0" xmlns="http://java.sun.com/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="
```

TEMA DA CAPA

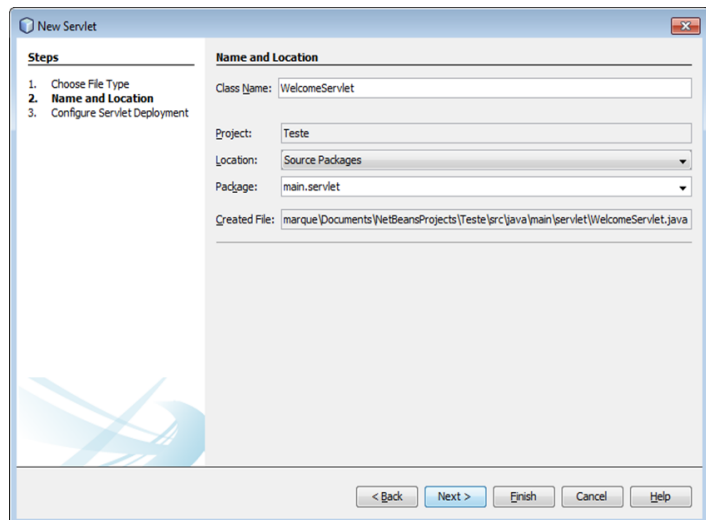
INTRODUÇÃO AO JAVA EE E JAVA WEB

```
"http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/
web-app_3_0.xsd">
<servlet>
  <servlet-name>WelcomeServlet</servlet-name>
  <servlet-class>main.servlet.WelcomeServlet
    </servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>WelcomeServlet</servlet-name>
  <url-pattern>/Welcome</url-pattern>
</servlet-mapping>
<session-config>
  <session-timeout>
    30
  </session-timeout>
</session-config>
</web-app>
```

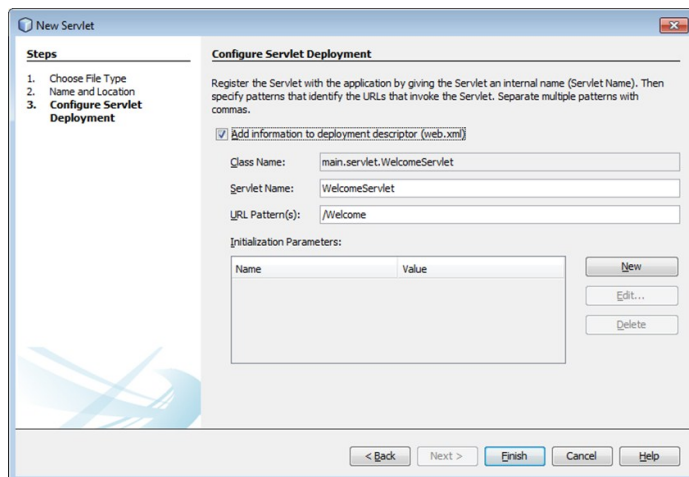
Mas não é necessário escrever estas *tags* até porque o NetBeans não permite a criação de *servlets* que já possuam referências no **web.xml** como tal o que podemos fazer é ir a opção **File – New File...** e dentro do Web selecionar **Servlet**.



Em seguida o IDE vai recomendar a não criar classes na raiz dos *packages* e como tal vamos preencher o campo **Package** com **main.servlet** e como nome **WelcomeServlet**.



Dado o passo seguinte o IDE automaticamente cria a estrutura dos *packages* introduzidos e apresenta o formulário de configuração da *servlet*.



Carregando sobre a check box e alterando o **URL Pattern** para **/Welcome** teremos automaticamente criado as *tags* pretendidas no **web.xml**.

Uma vez criada a *servlet*, vem com os métodos herdados de *HttpServlet* **doGet()** e **doPost()** implementados e com um método chamado **processRequest()**, que envia código HTML. O que vamos fazer é apagar este método e colocar a vazio o método **doPost()** e introduzir a seguinte linha no método **doGet()**:

```
getServletContext().getRequestDispatcher("/
home.jsp").forward(request, response);
```

Com esta linha criada e uma vez chamada a *servlet*, que por defeito responde por GET, o pedido vai ser reencaminhado para a nossa página criada **home.jsp**.

Fazendo *deploy* coloca-se o URL <http://localhost:8080/Teste/Welcome> e verifica-se que a nossa *servlet* está a fazer o reencaminhamento para a nossa página inicial.

Como próximo passo vamos enviar variáveis para uma *servlet* onde a mesma irá processar e enviar uma resposta. Então o que se pretende é tomando os mesmos passos de criação da **WelcomeServlet** vamos criar a **SomaServlet** com o seguinte código:

```
package main.servlet;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class SomaServlet extends HttpServlet {

    public static final String TEXTO = "texto";
    public static final String VALOR_1 = "valor1";
    public static final String VALOR_2 = "valor2";
    public static final String RESULTADO =
        "resultado";
```

TEMA DA CAPA

INTRODUÇÃO AO JAVA EE E JAVA WEB

```
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    request.setAttribute(TEXT0, "Insira 2
    numeros e submeta para obter o resultado");
    getServletContext().getRequestDispatcher
        ("/soma.jsp").forward(request, response);
}

@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    int resultado;
    try {
        int val1 = Integer.parseInt
            (request.getParameter(VALOR_1));
        int val2 = Integer.parseInt
            (request.getParameter(VALOR_2));
        resultado = val1 + val2;
        request.setAttribute(RESULTADO,
            ""+resultado);
    } catch (NumberFormatException e) {
        request.setAttribute(RESULTADO,
            "Valores não validos");
    }
    getServletContext().getRequestDispatcher
        ("/resultado.jsp").forward(request, response);
}
```

E para interagirmos com a *servlet* vamos criar as páginas **soma.jsp** e **resultado.jsp** onde mais uma vez vamos ao menu **File – New File...**, selecionamos JSP e preenchemos o nome sem a extensão no campo **File Name**.

soma.jsp

```
<%@page import="main.servlet.SomaServlet"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>
<% String texto = (String) request.getAttribute
    (SomaServlet.TEXT0);%>

<html>
<head>
<meta http-equiv="Content-Type"
    content="text/html; charset=UTF-8">
<title>Soma</title>
</head>
<body>
<h2><%=texto%></h2>

<form action="Soma" method="POST">
    Valor 1:<input type="text" name="
        <%=SomaServlet.VALOR_1%>">
    Valor 2:<input type="text" name="
        <%=SomaServlet.VALOR_2%>">
    <br>
    <input type="submit" name="submit">
</form>
</body>
</html>
```

Com o objectivo de evitar erros, a *servlet* possui constantes que permitem que haja uma coerência entre as variáveis enviadas e recebidas, com o objectivo de evitar erros. Pode-se observar que a *servlet* uma vez iniciada envia uma variável através de **request.setAttribute()**. Uma vez chamado o **setAttribute()** requer 2 parâmetros:

request.setAttribute(NomeDoAtributo, UmObjectoQQ);

O primeiro parâmetro, como descrito, atribui um nome, sendo esse o nome que deve ser usado nas páginas JSP para poderem capturar a variável enviada que se vai encontrar descrita no segundo parâmetro que, por sua vez, é sempre convertida da *servlet* para *object*. Como tal teve que ser efectuado o *cast* do objecto à chegada para o tipo de variável que tinha antes de ser enviada.

String atributo = (String) request.getAttribute(NomeDoAtributo);

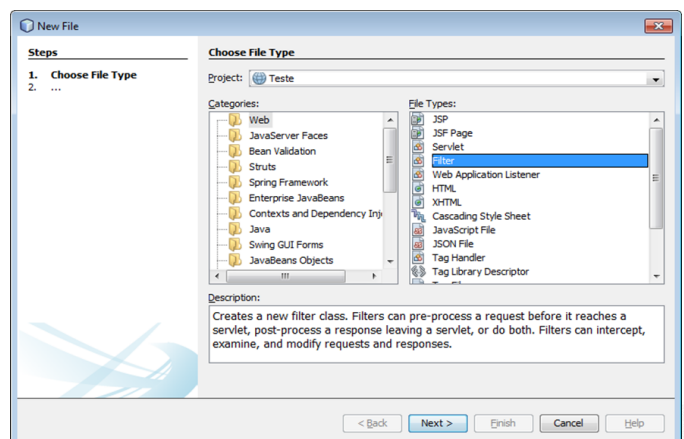
(Recepção do atributo)

Caso o que se pretenda seja o contrário enviar variáveis no sentido JSP à *Servlet* a página JSP deve de conter um formulário HTML com o método que se pretende: **POST** ou **GET** e na *action* o *URL pattern* da *servlet* em questão. E os nomes dos *inputs* são as referências que a *servlet* usa para receber as variáveis das páginas JSP, variáveis essas quem vêm como *String* e como tal devem de ser convertidas para o tipo desejado. (Como demonstrado no código acima referido). Vamos então testar acedendo ao URL: <http://localhost:8080/Teste/Soma>.

Porem a aplicação funciona mas apenas mostra páginas caso as mesmas sejam escritas no URL, pois uma vez que renomeamos o **index.jsp** para **home.jsp** não existe de momento nenhuma página inicial, ou seja, caso escrevessemos <http://localhost:8080/Teste/> a aplicação não vai mostrar nenhuma página.

Para solucionarmos esse problema vamos então criar um filtro que remeta para uma determinada *servlet*, filtro esse que vai aguardar que uma determinada variável se altere de estado para desbloquear as restantes páginas. Com este tipo de classes podemos definir níveis de acesso dentro da nossa aplicação restringindo a mesma quando certas variáveis não forem validadas ou não se encontram com o valor pretendido (por exemplo um determinado nome de utilizador e palavra chave)

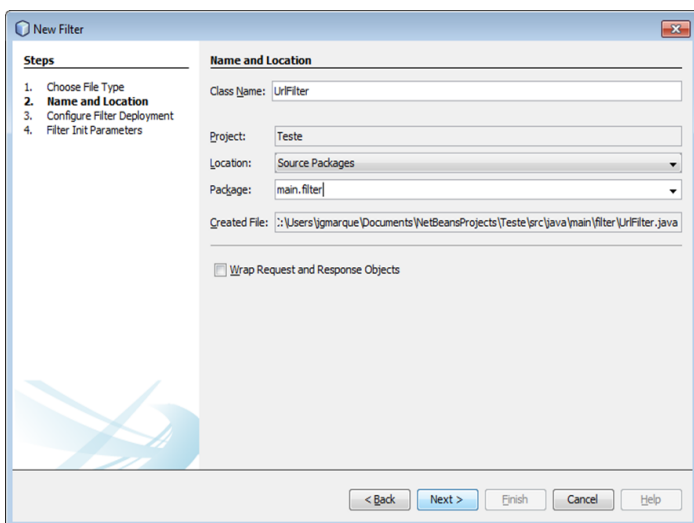
Vamos então novamente a **File – New File...** e selecionamos *Filter*



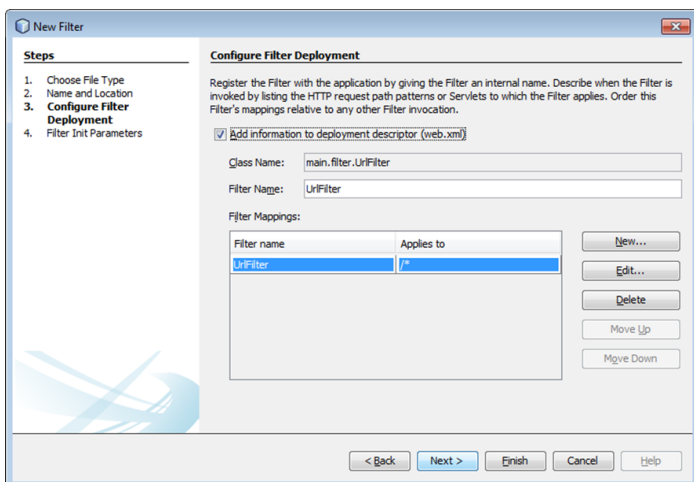
TEMA DA CAPA

INTRODUÇÃO AO JAVA EE E JAVA WEB

No seguinte passo vamos atribuir o nome de **UrlFilter** e colocá-lo num novo **package main.filter**.



Semelhante ao passo de criação do mapeamento das *servlets* no **web.xml** o IDE igualmente cria automaticamente um mapeamento do filtro no ficheiro XML, contudo o código gerado é ligeiramente diferente (Ver referência de tags do **web.xml**).



Após dar seguinte neste passo e não preencher nada no próximo, o IDE gera uma classe filtro por defeito e gera o seu mapeamento no **web.xml** semelhante a:

```
<filter>
  <filter-name>UrlFilter</filter-name>
  <filter-class>main.filter.UrlFilter
  </filter-class>
</filter>
<filter-mapping>
  <filter-name>UrlFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

Muito semelhante ao que se vai encontrar num mapeamento de uma *servlet* apenas com a diferença do nome das *tags* de *servlet* para *filter* e o seu mapeamento será múltiplo para todos os URLs que se deseja correr um determinado filtro antes de haver o redirecionamento. Úteis para criar políticas de autenti-

cação e restrição de acessos os filtros permitem a execução de certas linhas de código que mantêm estados numa aplicação e é o sítio ideal para colocar a parte do código da aplicação que manipula variáveis de sessão.

Após criar-se a classe do tipo *Filter*, ignorando o código gerado, o filtro deverá possuir o seguinte código:

```
package main.filter;

import java.io.IOException;
import javax.servlet.*;
import javax.servlet.http.*;

public class UrlFilter implements Filter {

    FilterConfig filterConfig;
    public static final String SESSION = "session";

    @Override
    public void doFilter(ServletRequest req,
        ServletResponse resp, FilterChain chain)
        throws IOException, ServletException {
        if (req instanceof HttpServletRequest) {
            String path = ((HttpServletRequest)
                req).getServletPath();
            HttpSession session =
                ((HttpServletRequest) req).getSession(false);
            boolean sessionValidator = false;
            if (session != null) {
                String validSession = (String)
                    session.getAttribute(SESSION);
                if (validSession != null) {
                    sessionValidator = true;
                }
            }
            if (!path.equals("/Welcome") && !
                sessionValidator) {
                ((HttpServletResponse)
                    resp).sendRedirect(filterConfig.getServletContext()
                        ().getContextPath() + "/Welcome");
            } else {
                chain.doFilter(req, resp);
            }
        }
    }

    @Override
    public void init(FilterConfig filterConfig) {
        this.filterConfig = filterConfig;
    }

    @Override
    public void destroy() {
    }
}
```

Uma vez criado o filtro este vai redirecionar todos os pedidos à aplicação para a **WelcomeServlet** enquanto a variável **validSession** estiver a NULL, como tal na *servlet WelcomeServlet* coloca-se a seguinte linha:

```
request.getSession().setAttribute(UrlFilter.SESSION,
    "gotSession");
```

Assim que o filtro passar pela **WelcomeServlet** a variável **validSession** vai deixar de estar a NULL colocando o filtro a fazer um encadeamento com os restantes pedidos que aparecerem.

TEMA DA CAPA

INTRODUÇÃO AO JAVA EE E JAVA WEB

Após implementado, quando se aceder à aplicação pelo URL apontado para a raiz (<http://localhost:8080/Teste/>) o servidor com conhecimento da existência de um filtro vai chama-lo para cada vez que o cliente acede a qualquer *servlet* ou JSP.

Para finalizar este exemplo vamos só criar um link na página **home.jsp** que nos vai apontar para a página **soma.jsp** através da **SomaServlet**.

```
<a href="Soma">Somar</a>
```

Dentro da aplicação os *URL pattern* que foram definidos no **web.xml** servem igualmente para os links HTML chamando sempre o método **doGet()** da *servlet*.

Conclusão

Este artigo mostra numa forma introdutória como se pode criar uma aplicação Java com uma interface Web tendo em conta que as capacidades do Java EE e das suas bibliotecas, nomeadamente Servlets, Filters e as configurações necessárias para criar uma aplicação Web básica com conceitos adicionais de XML(no seu geral), JSP e MVC.

Bibliografia/Referências

<http://www.oracle.com/technetwork/java/javase/downloads/jre7-downloads-1880261.html>

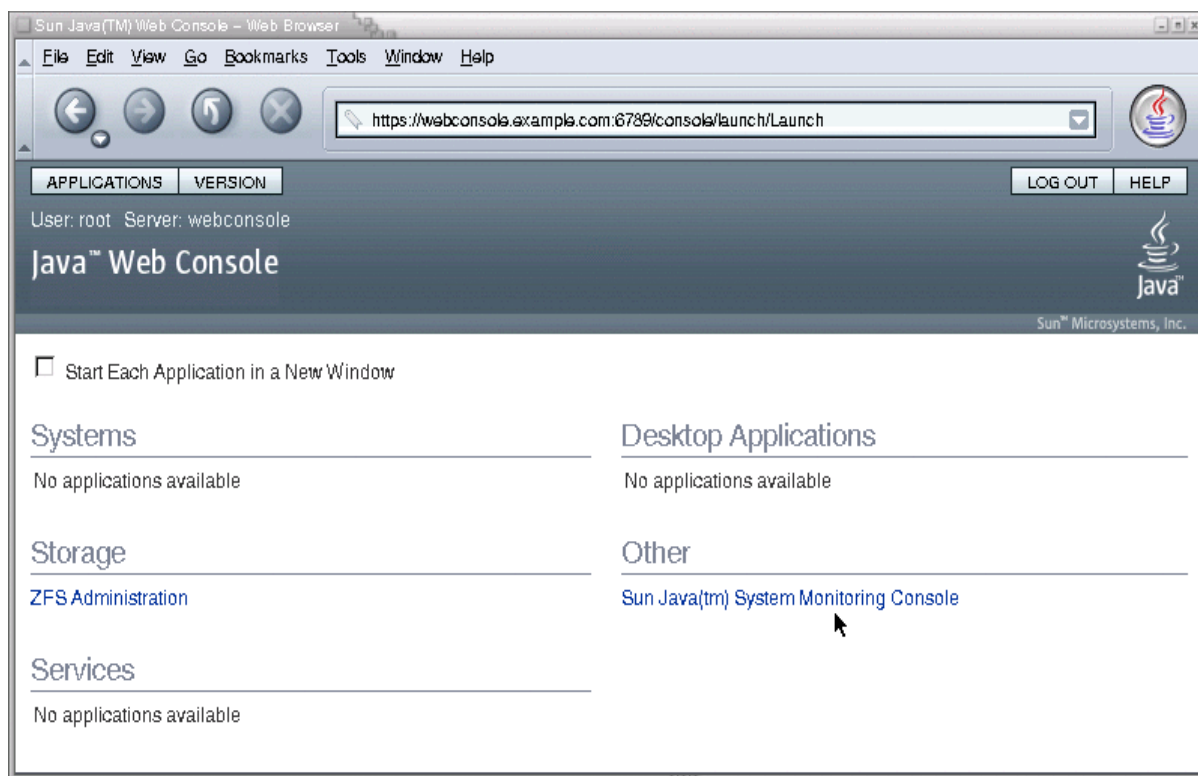
<http://www.oracle.com/technetwork/java/javaee/downloads/index.html>

<https://netbeans.org/features/index.html>

http://en.wikipedia.org/wiki/Java_EE_version_history

<http://www.jsptut.com/>

<https://developers.google.com/appengine/docs/java/config/webxml>



AUTOR



Escrito por **José Marques**

Natural de Coruche, formado em Gestão e programação de sistemas informáticos e técnico especialista de gestão de redes. Membro do P@P desde Abril de 2013.

A PROGRAMAR

Programação em C: const

Jogo da Vida

JBoss Application Server 7

PHP: Uma framework “from scratch” (Parte 1)

A PROGRAMAR

Programação em C: const

Uma das “recentes” adições à linguagem C foi o qualificador `const`. Ao contrário do que se possa pensar, utilizar `const` não cria uma constante mas sim uma variável só-de-leitura. O que queremos dizer com isto? Vejamos o seguinte exemplo:

```
1 const int i = 42;
2 int *ptr = (int*)&i;
3
4 *ptr = 73;
```

Qual o valor de `i` após execução da linha 4? Embora possa parecer estranho, `i` vai ter o valor 73, apesar de termos declarado `i` como constante. Por este motivo, variáveis qualificadas com `const` não são constantes verdadeiras em C (são no C++) e não podem por este motivo ser utilizadas em certos lugares, como a definição do tamanho de arrays ou em cases de um `switch` (excepto com extensões dos compiladores).

De que nos serve então um qualificador `const` se não assegura que a variável permanece inalterada? Bem, a utilização de `const` para qualificar variáveis impede, de facto, a sua alteração na maioria dos casos. Não podemos alterar `i` directamente, por exemplo (i.e. `i = 73`; emite um erro). No exemplo apresentado acima enganámos o compilador (linha 2) com o cast de `&i` para o tipo `int*`; sem cast, um compilador bem configurado emitirá pelo menos um aviso de que o qualificador `const` é descartado na atribuição de `&i` a `ptr`, um pequeno sinal de alarme para o programador (é só estar atento!).

Podemos, como referido acima, qualificar uma variável com `const` e impedir que seja alterada directamente. Se tentarmos apontar um apontador a essa variável, o compilador avisa-nos que a qualificação de `const` será perdida e podemos alterar a variável através do apontador. Haverá maneira de impedir que isto aconteça? Na realidade, a questão é outra: *como utilizar o qualificador `const` com apontadores?*

Existem 4 formas de qualificar um apontador com `const`, obtendo-se diferentes níveis de acesso de leitura e/ou escrita da variável apontada através do mesmo. Para que não existam dúvidas, é importante reter que o qualificador `const` qualifica uma variável e não um local na memória. Por outras palavras (e recorrendo ao exemplo inicial), qualificar `i` como `const` só impedir a alteração do valor *através da variável* `i`. Quando apontamos um apontador ao endereço de `i` podemos fazer o que quisermos com o valor lá armazenado, independentemente de como `i` foi declarado.

A primeira forma de usar `const` num apontador é, surpreendentemente, não usar `const` num apontador. Na realidade (obviamente) esta não é uma utilização do qualificador `const`,

mas este artigo não estaria completo sem se falar dela. Um apontador simples, de forma tipo `*p` permite-nos fazer várias coisas: podemos apontá-lo a qualquer local da memória que queiramos e, desde que apontado a um endereço válido, podemos alterar o valor aí armazenado com o operador de indirectão (eg. `*p = 3`). Este tipo de apontador não apresenta restrições nenhuma (por esse motivo é que o compilador emite um aviso quando tentamos apontá-lo a uma variável `const`, excepto na presença de casts).

Segue-se então a criação de um apontador para uma variável constante. Este tipo de apontador apresenta uma particularidade; embora seja possível apontá-lo a qualquer endereço de memória, não é possível utilizar este apontador para alterar o conteúdo do endereço apontado. Logicamente, este era o tipo de apontador que precisávamos no exemplo inicial; se temos uma *variável constante*, queremos apontar para ela com um apontador para uma *variável constante*:

```
1 const int i = 42;
2 const int *ptr = &i;
3
4 *ptr = 73; /* erro! */
```

Se lermos a declaração de `ptr` da direita para a esquerda, tudo faz sentido: `ptr` é um apontador (`*`) para um `const int`. E de facto `i` é um `const int`, tudo parece bem! Quando tentamos modificar o valor apontado por `ptr` o compilador emite um erro, queixa-se que estamos a tentar atribuir um valor a uma variável/endereço só de leitura. Este era o comportamento que procurávamos inicialmente.

Passemos agora para a terceira forma de usar `const` com apontadores. Uma espécie de “inverso” de um apontador para uma variável constante, é um *apontador constante* para uma variável (não constante). A diferença do anterior para este é que agora passamos a conseguir alterar o valor apontado (por outras palavras, `*ptr = 73` passa a ser possível), mas não nos é possível apontar `*ptr` para outra variável:

```
1 const int i = 42;
2 int * const ptr = &i;
3
4 *ptr = 73; /* ok */
5 ptr = NULL; /* erro */
```

Uma vez mais, da direita para a esquerda, `ptr` é um apontador constante (`* const`) para um `int`. Podemos utilizar `*ptr` à vontade para modificar o valor de 42 para 73, mas é-nos impossível alterar o endereço para o qual `ptr` aponta. Este tipo de apontadores é útil quando nos queremos assegurar de que não há alterações acidentais do local para onde um apontador aponta. Um pormenor importante: sendo `ptr` um apontador cujo endereço não pode ser alterado, devemos

apontá-lo para onde queremos no momento da sua declaração, sob pena de ficar apontado para um local aleatório sem possibilidade de alteração.

A quarta (e última) forma de utilizar `const` num apontador é, como devem ter imaginado, a combinação das duas últimas: um apontador constante para uma variável constante. Este é o tipo de apontador mais limitado de todos, uma vez que a única coisa que podemos fazer com ele após a sua declaração é consultar o valor para o qual aponta. Não podemos alterar `i` se `ptr` for deste tipo, nem podemos apontar `ptr` para `NULL` ou outro endereço qualquer:

```
1  const int i = 42;
2  const int * const ptr = &i;
3
4  *ptr = 73; /* erro */
5  ptr = NULL; /* erro */
```

Desta vez, utilizamos `const` em dois locais: à esquerda do asterisco para qualificar a variável apontada, e à direita do asterisco para qualificar o apontador. Neste caso nenhuma das operações funcionará; o compilador assegura-se que não utilizaremos `ptr` para alterar o valor para o qual aponta, nem tentaremos apontar `ptr` para outro endereço.

Neste artigo cobrimos algumas utilizações do qualificador `const` na linguagem C (volto a lembrar, estamos a falar de C e não de “C/C++”), e foi também referido como utilizar este qualificador para melhorar o código que escrevemos de forma a prevenir asneiras relativamente comuns quando se lida com apontadores.

Falta, no entanto, mencionar outra utilidade do `const`: documentação. Quando o programador utiliza `const` de uma forma pensada está a documentar o seu código! Se virmos uma função que aceita como argumento `const void*` podemos imediatamente imaginar que o endereço passado nunca será alterado pela função em causa, enquanto que um simples `void*` não nos informa sobre as intenções da API. Naturalmente, se o programador da API declarar um argumento com `const` e depois utilizar casts para se livrar da qualificação, então aí *all bets are off* e nada é garantido. Mas assumamos que isso é a exceção e não a regra.

Existem ainda múltiplos aspectos que não foram cobertos neste artigo, nomeadamente a conversão implícita de `T` para `const T` e `T*` para `const T*`. Esta conversão é bastante útil

quando queremos passar um valor do tipo `T` a uma função que requer `const T` como argumento. Se pensarmos bem, faz sentido: em qualquer lugar onde é necessário um `const T`, podemos utilizar um `T` perfeitamente (que nunca será alterado). Por outro lado, a conversão inversa (de `const T` ou `const T*` para `T` e `T*`, respectivamente) não é permitida, e também isso faz sentido: se uma função pede um valor do tipo `T*` não faz sentido aceitar um `const T*` porque isso implicaria potenciais alterações ao valor apontado (que são proibidas pelo qualificador `const`).

As conversões implícitas relativas ao `const` terminam por aqui. Em C, não há conversão de `T**` para `const T**`, o que significa que numa função que aceite um argumento `const T**` será emitido um aviso se tentarmos passar um `T**`. Neste caso, temos que recorrer a um cast para silenciar o compilador. Nada disto é um erro, no entanto; os casts surgiram porque são necessários na linguagem C, e recusar a sua utilização é negar a linguagem. Há que saber quando e como fazer as coisas da forma correcta.

“ **Ao contrário do que se possa pensar, utilizar `const` não cria uma constante mas sim uma variável só de leitura.** ”

Para terminar, recomendo uma olhadela às manpages da biblioteca standard de C: já utilizam `const` abundantemente, nota-se bem o seu valor como documentação dentro do código. Além deste aspecto da documentação, a utilização de `const` leva à criação de código mais previsível e seguro, e serve também como dica para possíveis optimizações feitas pelo compilador. Usem `const`. Informem e encorajem a utilização de `const`, especialmente aos iniciados, porque vale a pena.

AUTOR



Escrito por António Pedro Cunha (pwseo)

Médico natural de Guimarães, formado na Universidade do Minho.

Programador autodidacta em parte dos tempos livres, inscrito no fórum desde 2006.

Website: <http://pwseo.aloj.net>

Jogo da Vida

Nesta edição vamos propor ao leitor um olhar sobre um famoso algoritmo da década de 70.

O seu inventor, John Horton Conway, nasceu em Liverpool em Dezembro de 1937 e é um dos matemáticos mais conhecidos da nossa era em todo o mundo. Estudou na Universidade de Cambridge e ainda hoje é uma mente ativa nas mais variadas teorias como por exemplo a teoria dos códigos e a teoria combinatória dos jogos.

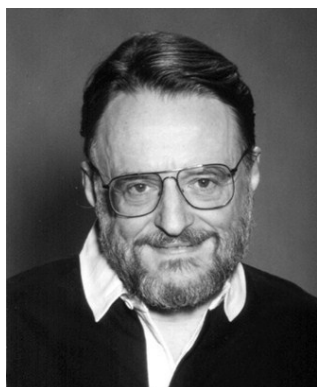


Ilustração 1 - John Conway

O algoritmo escolhido para esta edição foi criado em 1970 por Conway – o Jogo da Vida – sendo este um jogo para zero jogadores. Este jogo tornou-se mundialmente famoso através de uma coluna na revista Scientific American. Em 1971 foi tema de capa e tinha então relançado uma área da Matemática: a dos autómatos celulares, que são estruturas matemáticas úteis em simulações de processos físicos e biológicos, e que a um nível teórico podem comportar-se como computadores.

Um autómato celular é um modelo discreto e cada autómato deste tipo é representado por uma grelha de células, podendo essas células assumir um número finito de estados que variam de acordo com cada modelo. Outra característica a considerar é que o tempo de evolução é também discreto; por exemplo, o estado de uma célula no tempo t é sempre obtido tendo em consideração o estado dessa mesma célula (e respetiva vizinhança) no tempo $t-1$. Cada vez que todas as regras determinísticas são aplicadas a toda a grelha, uma nova geração é produzida.

Isto foi um dos motes de partida para que Conway criasse o Jogo da Vida. Este jogo é considerado um autómato celular bidimensional em que são simulados processos de evolução de células biológicas. Estudos já efetuados provam que este é um autómato computacional universal, isto é, é capaz de simular qualquer sistema evolucionário possível (reprogramando as respetivas regras determinísticas se isso for necessário).

Devido às analogias com o aumento, redução e alteração de um sistema de supostos organismos vivos (as células), este jogo pertence à classe dos “jogos de simulação” pois recriam processos do mundo-real.

Voltando de novo ao tema principal do nosso artigo, é importante recordar ao leitor que este jogo é para zero jogadores; o utilizador é apenas responsável pelos valores de inicialização do sistema. Depois desses valores serem introduzidos, as gerações seguintes serão sempre calculadas com base nas regras determinísticas do algoritmo.

Neste jogo, num dado instante, cada célula tem dois estados possíveis: “viva” ou “morta”.

As regras do jogo da vida são simples, ora vejamos:

- Uma célula viva com menos de dois vizinhos vivos morre por solidão.
- Uma célula viva com mais de três vizinhos vivos morre por sobrepopulação (por escassez de recursos).
- Uma célula viva com exatamente dois ou três vizinhos vivos permanece viva.
- Uma célula morta com exatamente três vizinhos vivos torna-se viva.

Ao contrário do artigo anterior desta série, a solução proposta para este problema (na linguagem C), propõe-se a ser algo extremamente simples, escrita de forma quase elementar.

Implementação do Algoritmo

Para simular o sistema de jogo foi usada uma matriz, e para que o cálculo das regras fosse simples, foi considerado que todas as células dessa matriz teriam 8 vizinhos. A forma mais direta encontrada para isso foi inserir a matriz do utilizador (que neste artigo será a matriz secundária) numa matriz com dimensão $[\text{linhas}+2][\text{colunas}+2]$ (matriz principal) para garantir que até as células dos “cantos” teriam os tais 8 vizinhos.

De forma a simplificar a leitura da nossa implementação definimos que cada célula viva terá o valor 1 (representada pela macro CELVIVA) e que cada célula morta terá o valor 0 (representada pela macro CELMORTA).

No início da função `jogo()`, são pedidos ao utilizador os valores para o número de linhas e colunas do sistema de jogo e o número de gerações a simular.

Após termos esses valores é então criada a matriz principal com todos os campos inicializados a -1 para que seja efetua-

da uma “limpeza” da memória utilizada. Seguidamente, e através desses valores, é inicializada dentro na matriz principal, a submatriz (matriz secundária) que é usada para calcular os valores das células em cada geração. Neste caso, é inicializada a matriz com a dimensão que o utilizador pediu com o valor de CELMORTA em todos os campos.

Após gerar a matriz secundária, é perguntado ao utilizador quantas células vivas pretende ter no jogo. Seguidamente são pedidas as coordenadas dessas mesmas células. Quando o programa recebe esses valores (da linha e da coluna pretendida) então a essas posições da matriz é atribuído o valor CELVIVA.

Resumindo: para representar uma célula viva é usado o valor 1, e para representar uma célula morta é usado o valor 0. Todas as outras posições da matriz principal permanecem inalteradas com o valor -1.

Assim, o que o algoritmo faz é contabilizar quantas células vivas estão nos 8 vizinhos. Ou seja, por outras palavras, para cada posição (i,j) da matriz, o programa vai verificar as posições: (i-1, j-1), (i-1, j), (i-1, j+1), (i, j-1), (i, j+1), (i+1, j-1), (i+1, j) e (i+1, j+1). Caso o valor de cada uma dessas células seja igual a 1 (CELVIVA, i.e. célula viva), a variável de contagem (cvivas) é incrementada uma unidade.

Após a verificação de todas as posições, é verificado se a célula da posição (i, j) é uma célula viva ou morta e são verificadas as regras do jogo, para que se possa decidir se a célula (i,j) permanece viva ou se morre (tomando, respetivamente, o valor CELVIVA e CELMORTA na submatriz).

Usando a variável i como auxiliar para as linhas e a variável j como auxiliar para as colunas, os cálculos iniciam-se na posição (1,1) da matriz principal, pois é nestas coordenadas que se inicia a matriz secundária (i.e., o sistema de jogo do utilizador). Os cálculos são feitos em todas as posições (i,j) apenas da matriz secundária, daí que a condição de execução dos ciclos for seja enquanto o i e o j forem menores que o valor das linhas e das colunas introduzidas pelo utilizador. Ou seja, a matriz principal não é toda processada nos cálculos dos vizinhos vivos, para que não sejam gastos recursos de computação desnecessariamente.

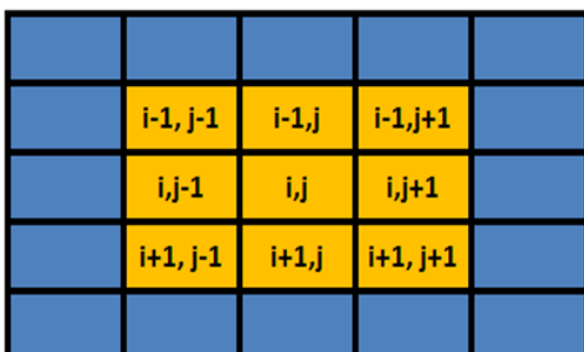


Ilustração 2 - Exemplo de verificação de Células

Célula a célula, a matriz atual é processada, sendo que após esse processamento, o resultado do cálculo (se a célula vive ou morre) é colocado na posição correspondente na matriz nova (que é uma matriz com as mesmas características da matriz atual, isto é, tem também uma matriz principal e uma matriz secundária).

Após esse cálculo é mostrada no ecrã a matriz secundária da matriz nova e a informação das células da matriz nova é copiada para a matriz atual, de forma a que na próxima iteração do for principal (que executa o mesmo numero de vezes, quantas gerações o utilizador peça ao programa) a informação volte a ser processada de forma análoga.

Para que fosse mais simples ao utilizador o uso do nosso programa, foi tomada em consideração, para que não se tornasse aborrecido estar sempre a iniciar a execução do programa caso o utilizador quisesse continuar a jogar, que o programa apenas terá fim quando o utilizador escolher sair do programa neste caso introduzindo o valor 0 no menu principal do mesmo.

Se o utilizador preferir jogar novamente então basta carregar em 1 e o programa volta a gerar uma nova jogada, gerando um novo sistema de gerações consoante os valores introduzidos pelo utilizador. Na nossa implementação, quando o programa mostra o sistema de jogo, as células vivas são representadas pelo símbolo *.

Código do Programa

Este é apenas um excerto da função que calcula a próxima geração tendo como base uma matriz que represente a geração anterior. O código completo poderá ser consultado no fórum do P@P:

```
void jogo(void) {  
    ...  
    for (int g = 0; g < geracoes; g++) {  
        printf("\n\nGERACAO: %d ", g + 1);  
  
        for (int i = 1; i <= linhas; i++) {  
            for (int j = 1; j <= colunas; j++) {  
                int cvivas = 0;  
  
                if (matrizActual[i-1][j-1] == CELVIVA)  
                    cvivas++;  
  
                if (matrizActual[i-1][j] == CELVIVA)  
                    cvivas++;  
  
                if (matrizActual[i-1][j+1] == CELVIVA)  
                    cvivas++;  
  
                if (matrizActual[i][j-1] == CELVIVA)  
                    cvivas++;  
                if (matrizActual[i][j+1] == CELVIVA)  
                    cvivas++;  
  
                if (matrizActual[i+1][j-1] == CELVIVA)  
                    cvivas++;  
  
                if (matrizActual[i+1][j] == CELVIVA)
```

A PROGRAMAR

JOGO DA VIDA

```
    cvivas++;
    if (matrizActual[i+1][j+1] == CELVIVA)
        cvivas++;
    if (isAlive && cvivas < 2)
        DIE;
    else if (isAlive && cvivas > 3)
        DIE;
    else if (isAlive && (cvivas == 2 || cvivas
                        == 3))
        LIVE;
    else if (isDead && cvivas == 3)
        LIVE;
    else
        DIE;
}
}
```

Como o leitor pode verificar nas linhas acima, este é um algoritmo bastante simples e de rápida implementação. É de referir ainda que o código do artigo desta edição foi implementado segundo o standard C99 (suportado pelos compiladores GCC, Clang e Pelles C), que nos permite programar de uma forma portátil e ainda assim confortável (comentários começados com //, variable-length arrays, tipo de dados bool, entre outros).

Caso o leitor queira compilar o código deste artigo em Linux deve utilizar a linha de comando: gcc -Wall -Wextra -std=c99 -pedantic jogo-da-vida jogo-da-vida.c

O objetivo principal deste algoritmo é então que o utilizador possa decidir qual a sua população inicial e que após isso possa observar o comportamento do seu sistema. Ao longo dos anos muitas configurações foram testadas e experimentadas com este jogo. O jogo da vida convida a experimentar diversas configurações, sendo que à maior parte das configurações iniciais escolhidas bastam apenas três ou quatro gerações para a população desaparecer completamente. Contudo há algumas configurações já conhecidas como configurações de “Vida Eterna” por haver sempre células vivas ao longo de muitas gerações (e até mesmo de gerações infinitas) como mostra, por exemplo, imagem seguinte (obtida com a execução do nosso programa).

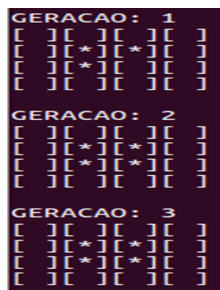


Ilustração 3 - Exemplo configuração “Vida Eterna” (Bloco).

Existem ainda diferentes configurações que podem ocorrer no Jogo da Vida (configurações essas que o utilizador pode facilmente encontrar na internet relativas a vários estudos já feitos sobre este algoritmo). Os exemplos mais simples são mostrados abaixo, com as células vivas em preto, e as células mortas em branco.



Há configurações que são alvo de vários estudos científicos, por exemplo, se dois gliders são colocados em direção a um bloco, o bloco vai aproximar-se da fonte dos gliders, mas por outro lado se três gliders são colocados no mesmo lugar, o bloco vai afastar-se desses gliders. Este facto é conhecido como a “memória de bloco deslizante e pode ser usada para simular um contador.

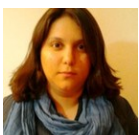
É possível construir portas lógicas AND, OR e NOT usando gliders, tal como é possível construir uma imagem que aja como uma máquina de estado finito conectada a dois contadores.

“ a matriz principal não é toda processada (...) para que não sejam gastos recursos de computação ”

O que significa que esta disposição possui o mesmo poder computacional de uma máquina de Turing universal, ou seja, Jogo da Vida, pode ser tão poderoso quanto qualquer computador com memória ilimitada dando origem ao Turing completo, como já vários estudos comprovam esse mesmo feito.

Mais uma vez recorro ao leitor que pode experimentar sem qualquer restrição as configurações iniciais que mais lhe agradarem. Este artigo é o terceiro de uma série de artigos de programação tendo como base principal a linguagem C, que esperamos que siga atentamente.

AUTOR



Escrito por Rita Peres

Natural de Castelo Branco, licenciou-se em Engenharia Informática pela Universidade da Beira Interior. Membro do P@P desde Janeiro de 2010.

JBoss Application Server 7

Introdução

Atualmente, a plataforma Java é uma das mais utilizadas no mundo, e muito disso se deve à capacidade da plataforma suportar outras linguagens como por exemplo JRuby, Groovy e Scala. Milhares de aplicativos para Web e Mobile são desenvolvidos a cada mês utilizando esta plataforma como base. A plataforma Java mudou muito desde seu início, e provavelmente vai continuar evoluindo nos próximos anos.

Em paralelo com essas mudanças, visualizamos os servidores de aplicação que cada vez mais oferecem recursos de alta complexidade, como componentes de balanceamento de carga inteligente, implementação de autenticação (JAAS), troca de mensagens assíncronas (JMS), controle de transação (JTA), persistência de objetos (JPA), componentes para criação de aplicações distribuídas e clusterizadas (EJB), API Java para processamento de arquivos XML e Webservices (JAX-WS e JAX-B) e muitas outras tecnologias facilitando o desenvolvimento e padronização de aplicações. Eles também disponibilizam uma infraestrutura estável e escalonável para aplicações de missão crítica. Um dos grandes desafios dos servidores de aplicação é acoplar serviços de grande complexidade, deixando-os estáveis e flexíveis.

Nas versões iniciais a plataforma ainda conhecida como Java2EE não estava madura e seus recursos nos servidores de aplicação eram carregados de forma hierárquica consumindo muitos recursos e tornando o startup altamente custoso.

A arquitetura dos principais servidores de aplicação foi redesenhada para que seus recursos possam ser iniciados de modo concorrente ou sob demanda. As melhorias também podem ser notadas no consumo de memória, onde em uma simples máquina desktop, pode-se facilmente configurar serviços de alta disponibilidade e escaláveis para realização de testes em tempo de desenvolvimento.

Se você deseja um ambiente com alta disponibilidade, escalonável e que seja independente de fabricante (de forma a evitar o vendor lock-in) o Java EE e seus servidores de aplicação foram feitos para si.

JBoss Application Server 7

JBoss Application Server 7 ou apenas JBoss AS 7 é um servidor de aplicação de código aberto que é 100% compatível com a especificação Java EE 6. Em versões anteriores do Java EE, mesmo utilizando somente algumas tecnologias para o desenvolvimento da aplicação, éramos obrigados a lidar com todos os recursos implementados no servidor. Para resolver esse problema no Java EE 6 foi inserido o conceito

de profile que tem como objetivo criar configurações com fins específicos como por exemplo o Web Profile que possui tecnologias para o desenvolvimento web.

Até ao momento, a versão atual do JBoss AS foi baixada mais de 150 mil vezes. Para mais informações visite o site do projeto <http://www.jboss.org/jbossas>.

Recentemente o projeto recebeu um novo nome e futuramente se chamará WildFly. Mais informações podem ser encontradas em: <http://www.wildfly.org/faq/>

Requisitos

Como todo servidor de aplicação o JBoss AS 7 requer um ambiente com JDK 1.6/JDK 1.7 devidamente configurado e instalado. O Java Development Kit (JDK) é um conjunto de utilitários para criação de softwares para plataforma Java. Existem várias implementações cada qual com a sua finalidade.

Para baixar o JDK 7 navegue até a página de download da Oracle em <http://www.oracle.com/technetwork/java/javase/downloads/index.html> e escolha a opção Java Platform (JDK). Em seguida você será direcionado para outra página onde deverá aceitar a License Agreement. Baixe a versão `jdk-7u21-linux-x64.rpm`.

Para instalar o JDK execute o seguinte comando:

```
$ sudo rpm -Uvh jdk-7u21-linux-x64.rpm
```

Execute o comando `java --version` para verificar se o JDK está instalado:

```
$ java --version
```

```
java version "1.7.0_21"
```

```
Java(TM) SE Runtime Environment (build 1.7.0_21-b11)
```

```
Java HotSpot(TM) 64-Bit Server VM (build 23.21-b01, mixed mode)
```

Duvidas na instalação? Consulte o link:

<http://docs.oracle.com/javase/7/docs/webnotes/install/index.html>

Instalando o JBoss AS 7

O JBoss AS 7 pode ser baixado gratuitamente no site da comunidade: <http://www.jboss.org/jbossas/downloads/> Para instalar basta descompactar o arquivo `jboss-as-7.1.1.Final.zip` utilizando um utilitário de descompressão.

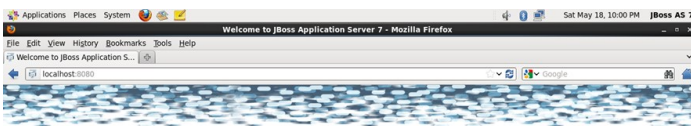
Boas Práticas

Nunca inicie o JBoss utilizando o usuário root, pois a plataforma Java oferece APIs para execução de códigos nativos do sistema operacional e mecanismos de gerenciamento remoto. No Linux crie um usuário com privilégios de root para iniciar o serviço do JBoss. Já no Windows crie um usuário com poderes administrativos, mas com privilégios reduzidos.

Uma vez instalado, é fortemente aconselhável iniciar o JBoss AS para verificar se existe alguma incompatibilidade com a arquitetura do JDK utilizado ou até mesmo se a memória disponibilizada é suficiente.

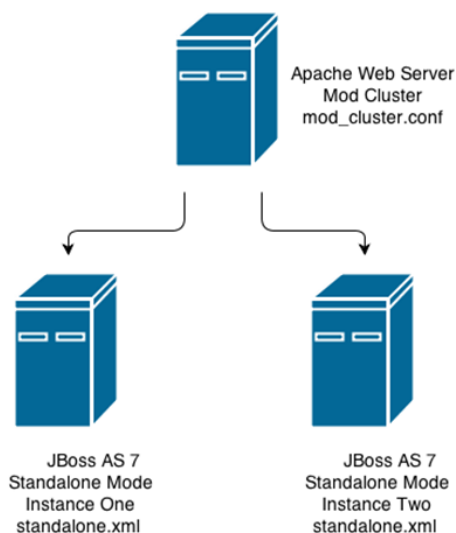
Execute uma simples instância standalone e acesse a url `http://localhost:8080` para verificar se o JBoss foi iniciado corretamente.

```
$ ./jboss-as-7.1.1.Final/bin/standalone.sh
```



Domain Mode x Standalone Mode

O JBoss AS 7 possui dois modelos de trabalho conhecidos como Standalone Mode e Domain Mode. No Standalone Mode podemos trabalhar com uma única instância muito semelhante a versões anteriores do JBoss AS 5 e JBoss AS 6. Para utilizar o Standalone mode basta iniciar o script `JBOSS_HOME/standalone.sh` no Linux.



Agora você deve estar se perguntando se é possível criar um ambiente de alta disponibilidade utilizando o Standalone Mode. A resposta é Sim! É possível criar um ambiente com recursos clusterizados, replicação de sessão e tudo mais. Se você já utilizou outras versões do JBoss deve estar pensando em utilizar o famoso Farm Deployment onde a aplicação era replicada para todos os nós do cluster. Esse recurso não existe no Standalone Mode, pelo que você terá que realizar o deploy em todos os nós um por um. Isso não é um problema se forem 3 ou 4 instâncias, mas imagine se forem umas 20 instâncias.

Uma solução para esse problema seria a utilização de ferramentas como o RHQ (<http://www.jboss.org/rhq>) para realização do deploy em múltiplas instâncias o que acarretaria consumo de mais recursos e tempo. Para solucionar esse problema temos uma solução bem simples já embutida no servidor de aplicação o Domain Mode e é sobre isso que vamos falar no próximo tópico.

Introdução ao Domain Mode

O Domain Mode permite iniciar várias instâncias e também oferece uma maneira centralizada de gerenciamento dos recursos facilitando a administração das instâncias JBoss. O Domain Mode pode ser visto como uma unidade de instâncias que compartilham recursos e configurações e são administradas por um processo chamado Domain Controller.

Para iniciar o JBoss AS 7 de modo domain execute o script `JBOSS_HOME/domain.sh` no Linux ou `JBOSS_HOME/domain.bat` no Windows.

Quando iniciamos o JBoss em Domain Mode na configuração default temos no mínimo quatro processos: um Host Controller, um Process Controller e dois Servers.

Domain Controller: Ele é quem controla o gerenciamento do domain. Nele estão as configurações que são compartilhadas entre as instâncias que estão nesse domain.

Process Controller: Ele é de grande importância pois ele é responsável pela criação das instâncias e também do Host Controller que vamos falar a seguir. O Process Controller não deve ser confundido com uma instância, ele é simplesmente um processo na JVM.

Host Controller: Como Domain Controller o Host Controller também coordena as instâncias do domain. Ele é o responsável por fazer algo semelhante ao Farm Deployment (não existe nessa versão), ou seja, ele distribui o arquivo deployado para todas as instâncias do domain.

Servers: São as instâncias em si, onde estão as aplicações deployadas. Um ponto importante é que cada server é um processo Java.

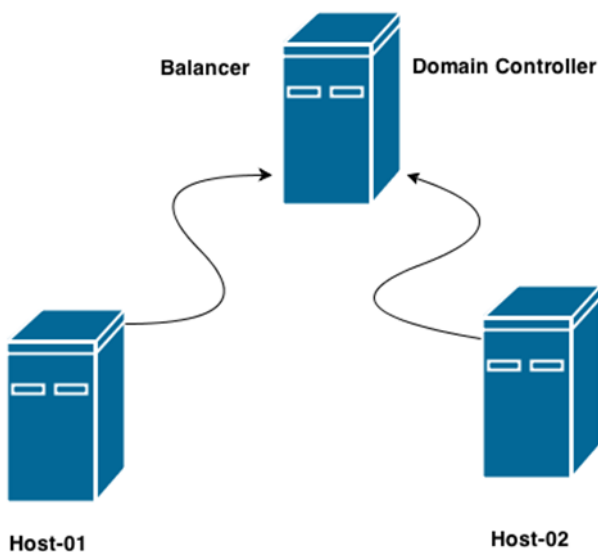
Entre os benefícios da utilização do Modo Domain que nós podemos citar estão:

- Gestão Centralizada;

- Configuração Centralizada;
- Deploy Centralizado;
- Manutenção Centralizada;

Preparando a Infra Estrutura

Nesse artigo estamos utilizando Três servidores. Dois deles serão utilizados pelas as instâncias JBoss e Um será utilizado Pelo Domain Controller e Apache Web Server para realizar o Balanceamento de carga.



Crie um grupo e adicione um usuário para ser utilizado pelo JBoss AS 7 em todos os servidores:

```
$ sudo groupadd jboss
$ sudo useradd -s /bin/bash -d /home/jboss -m -g jboss
jboss
```

Faça também o download do JBoss AS 7 no diretório tmp:

```
$ cd /tmp
$ wget http://download.jboss.org/jbossas/7.1/jboss-as-7.1.1.Final/jboss-as-7.1.1.Final.zip
```

Crie a estrutura de diretórios para armazenar o JBoss:

```
$ sudo mkdir /usr/local/jboss
$ chown jboss:jboss /usr/local/jboss
$ su jboss
$ mkdir /usr/local/jboss/
$ cd /usr/local/jboss/
$ unzip /tmp/jboss-as-7.1.1.Final.zip
```

Finalmente o JBoss AS 7 está instalado em /usr/local/jboss/jboss-as-7.1.1.Final em todos os servidores.

Configurando Alta Disponibilidade

Para aplicações com muitas solicitações, a busca pela melhora do desempenho é quase constante. Um dos meios mais utilizados é o Balanceamento de Carga, que consiste em distribuir a carga das solicitações em vários servidores ou instâncias proporcionando a melhora no tempo de reposta.

O JBoss AS 7 possui um componente nativo conhecido como mod_cluster que foi criado para atender a mecanismos de balanceamento de carga alinhados ao conceito de Cloud. O Mod Cluster vem com algoritmos de balanceamento de carga mais avançados, que se baseiam na carga da aplicação, ou seja, quantidades de sessões, conexões abertas, entre outros. Ele pode ser customizado conforme a necessidade da aplicação visando um ambiente elástico. Um das vantagens é também o descobrimento automático de novas instâncias JBoss (utilizando Multicast), não havendo a necessidade de configurações extras.

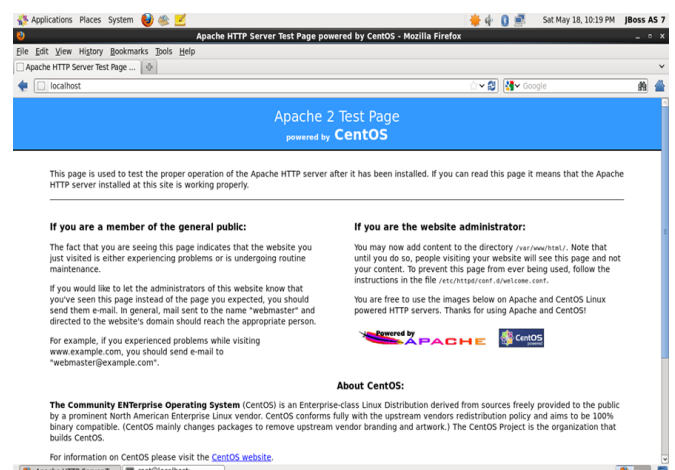
As configurações do mod_cluster devem ser realizadas no JBoss AS 7 e no Apache Web Server que será utilizado como Proxy reverso.

Instalando Apache Web Server

No Servidor Balancer instale o Apache Web Server. Basta executar o seguinte comando:

```
$ sudo yum install httpd -y
```

Inicie o serviço e verifique se página de testes é carregada:



```
$ sudo service httpd start
```

Instalando Mod Cluster

Ainda no Servidor Balancer instale o Mod Cluster. Ele pode ser baixado gratuitamente no site da comunidade: http://www.jboss.org/mod_cluster/downloads/1-2-0-Final/ Para instalar basta

descompactar o arquivo `mod_cluster-1.2.0.Final-linux2-x64-ssl.tar.gz` utilizando um utilitário de descompressão e copiar os módulos para o diretório de módulos do Apache Web Server:

Para baixar e extrair o arquivo execute:

```
$ cd /tmp
$ wget http://downloads.jboss.org/
mod_cluster//1.2.0.Final/mod_cluster-1.2.0.Final-
linux2-x64-ssl.tar.gz
$ tar -zxvf mod_cluster-1.2.0.Final-linux2-x64-
ssl.tar.gz
```

Copie os módulos para o Apache:

```
$ sudo cp /tmp/opt/jboss/httpd/lib/httpd/modules/
mod_advertise.so /etc/httpd/modules/
$ sudo cp /tmp/opt/jboss/httpd/lib/httpd/modules/
mod_manager.so /etc/httpd/modules/
$ sudo cp /tmp/opt/jboss/httpd/lib/httpd/modules/
mod_proxy_cluster.so /etc/httpd/modules/
$ sudo cp /tmp/opt/jboss/httpd/lib/httpd/modules/
mod_slotmem.so /etc/httpd/modules/
```

Edite o arquivo `/etc/httpd/conf/httpd.conf` e comente a linha `LoadModule proxy_balancer_module modules/mod_proxy_balancer.so`, para não ocorrer um conflito com `LoadModule proxy_cluster_module modules/mod_proxy_cluster.so`:

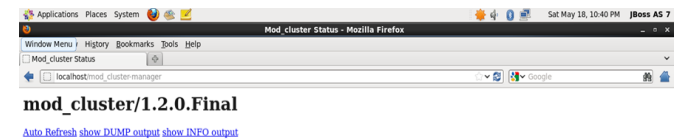
```
#Content Mod Cluster
LoadModule slotmem_module modules/
mod_slotmem.so
LoadModule manager_module modules/
mod_manager.so
LoadModule proxy_cluster_module modules/
mod_proxy_cluster.so
LoadModule advertise_module modules/
mod_advertise.so
<VirtualHost *:80>
<Directory />
Order deny,allow
Allow from all
</Directory>
KeepAliveTimeout 60
ManagerBalancerName mycluster
MaxKeepAliveRequests 0
ServerAdvertise On
EnableMCPMReceive On
</VirtualHost>
<Location /mod_cluster-manager>
SetHandler mod_cluster-manager
Order deny,allow
Allow from all
</Location>
```

```
#LoadModule proxy_balancer_module modules/
mod_proxy_balancer.so
```

Crie o arquivo `mod_cluster.conf` e adicione a seguinte configuração:

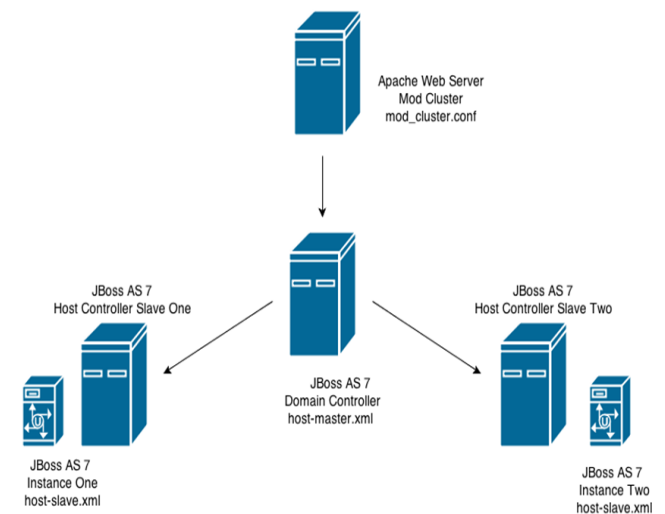
```
$ sudo vim /etc/httpd/conf.d/mod_cluster.conf
```

Reinicie o Apache:



```
$ sudo service httpd restart
```

Acesse o `mod_cluster-manager` e verifique se as informações sobre a versão do `mod_cluster` estão aparecendo:



Arquitetura dos Servidores em Modo Domain

Para realizar os testes do balanceamento de carga com o `mod_cluster`, vamos subir o Domain Controller e depois vamos conectar dois Hosts Controller com uma Instância JBoss em cada.

Criando os Perfis dos Servidores JBoss

No Servidor Balancer crie o perfil chamado master que será o Domain Controller:

```
$ cp -Rap /usr/local/jboss/jboss-as-7.1.1.Final/
domain/ /usr/local/jboss/jboss-as-7.1.1.Final/master
```


No Servidor Host-01 crie o perfil chamado `host01` que será um dos Hosts Controllers:

```
$ cp -Rap /usr/local/jboss/jboss-as-7.1.1.Final/domain/  
usr/local/jboss/jboss-as-7.1.1.Final/host01
```

No Servidor Host-02 crie o perfil chamado `host02` que também será um dos Hosts Controllers:

```
$ cp -Rap /usr/local/jboss/jboss-as-7.1.1.Final/domain/  
usr/local/jboss/jboss-as-7.1.1.Final/host02
```

Configurando Toda a Arquitetura

Como vimos anteriormente todas as configurações e gerenciamento são realizadas de forma centralizada no Domain Controller. Quando for necessário adicionar um novo grupo de servidores, configurar logs, criar datasources, alterar portas entre outras coisas, tudo isso deverá ser feito no `domain.xml` do Domain Controller que nesse caso é chamado de master. Sendo assim todas as nossas instâncias JBoss usufruirão das configurações realizadas para o Domain ou para o Server Group em questão.

A primeira coisa a ser feita é definir qual o conjunto de tecnologias (subsystems) serão utilizadas na aplicação. Isso pode variar conforme os requisitos de cada sistema. Um conjunto de tecnologias (subsystems) no JBoss 7 é chamado de profile.

```
<profile name="full-ha">
```

O JBoss AS 7 possui 4 profiles por padrão: "default", "full", "ha", "full-ha", mas nada impede de nós copiarmos qualquer um deles e renomeá-los conforme nossos requisitos. No Java EE 6 foi introduzido o conceito de Profiles, onde se pode criar um subconjunto de tecnologias presentes na spec Java EE ou até mesmo adicionar novas definidas pela JCP (Java Community Process).

Mais informações:

<https://community.jboss.org/wiki/JavaEE6UtilizandoWebProfileOuFullProfileNoJBossAS7>

Nós vamos utilizar o profile `ha`, pois nele os recursos para clusterização estão disponíveis.

Depois de definir o profile o próximo criar um Grupo de Servidores: Server Group. Um server group nada mais é que um agrupamento de instâncias JBoss. Nessa arquitetura vamos utilizar apenas um server group. Isso pode ser definido na tag `<server-groups>` no `domain.xml` do master (Domain Controller).

Crie um Server Group chamado `apps` que utilize o profile `ha`:

```
$ vim /usr/local/jboss/jboss-as-7.1.1.Final/master/  
configuration/domain.xml
```

```
<server-group name="apps" profile="ha">  
  <jvm name="default">  
    <heap size="1303m" max-size="1303m"/>  
    <permgen max-size="256m"/>  
  </jvm>  
  <socket-binding-group ref="ha-sockets"/>  
</server-group>
```

Os outros Server Groups podem ser removidos.

No profile `ha` especificamente no subsystem `web` adicione o atributo `instance-id`:

```
<subsystem xmlns="urn:jboss:domain:web:1.2" de-  
fault-virtual-server="default-host" instance-  
id="{jboss.server.name}"  
native="false">
```

O `instance-id` será passado como parâmetro na inicialização do Host Controller.

O próximo passo é configurar o atributo `proxy list` no subsystem `modcluster`:

```
<subsystem  
  xmlns="urn:jboss:domain:modcluster:1.0">  
<mod-cluster-config advertise-socket="modcluster"  
  proxy-list="192.168.238.186:80">  
<dynamic-load-provider>  
  <load-metric type="busyness"/>  
</dynamic-load-provider>  
</mod-cluster-config>  
</subsystem>
```

Nesse atributo está configurado o IP e porta do Apache Web Server.

No Servidor Host-01 edite o arquivo `/usr/local/jboss/jboss-as-7.1.1.Final/host01/configuration/host-slave.xml` para criar a instância JBoss.

Na tag `<servers>` é onde estão de fato as nossas instâncias JBoss. Quando criamos um novo `<server>` estamos criando uma nova instância JBoss.

Conforme foi definido na arquitetura o Host Controller 01 conterá apenas uma instância JBoss chamada `instance-one`.

Deixe o `<servers>` como abaixo:

```
<servers>  
  <server name="instance-one" group="apps">  
  </server>  
</servers>
```

Definimos que a instância JBoss `instance-one` fará parte do Server Group `apps`.

Para finalizar na tag `<host>` adicione o nome `host01`.

```
<host name="host01" xmlns="urn:jboss:domain:1.2">
```

A PROGRAMAR

JBOSS APLICATION SERVER 7

Repita os procedimentos realizados no servidor Host-01 em Host-02 alterando apenas o nome do servidor que será instance-two e do <host> que será host02.No ambiente em que realizei os testes o sevidores possuem os seguintes IPs:

Domain Controller+Apache WebServer: 192.168.238.186

Host Controller 01: 192.168.238.187

Host Controller 02: 192.168.238.188

Inicie o Domain Controller (master) utilizando os seguintes parâmetros:

```
./usr/local/jboss/jboss-as-7.1.1.Final/bin/domain.sh
-Djboss.domain.base.dir=/usr/local/jboss/jboss-as-7.1.1.Final/master/
-Djboss.host.default.config=host-master.xml
Djboss.bind.address=192.168.238.186
Djboss.bind.address.management=192.168.238.186
```

Verifique o log:

JBoss AS 7.1.1.Final "Brontes" (Host Controller) started in 15505ms - Started 11 of 11 services (0 services are passive or on-demand)

O Domain Controller foi iniciado com sucesso. Conecte primeiro Host Controller 01 (Host-01) ao Master. Inicie o Perfil host01 utilizando os seguintes parâmetros:

```
./usr/local/jboss/jboss-as-7.1.1.Final/bin/domain.sh
-Djboss.domain.base.dir=/usr/local/jboss/jboss-as-7.1.1.Final/host01/
-Djboss.host.default.config=host-slave.xml
-Djboss.domain.master.address=192.168.238.186
-Djboss.bind.address.management=192.168.238.187
-Djboss.bind.address=192.168.238.187
-Djboss.server.name=host-01
```

Agora observe os logs no Domain Controller e perceba que o Host Controller 01 se conectou ao Domain:

JBAS010918: Registered remote slave host "host01", JBoss AS 7.1.1.Final "Brontes"

Observe também o Mod Cluster Manager em http://192.168.238.186/mod_cluster-manager e veja que a nossa primeira instância já apareceu:



Inicie o Host Controller 02 (slave02) utilizando os seguintes parâmetros:

```
./usr/local/jboss/jboss-as-7.1.1.Final/bin/domain.sh
-Djboss.domain.base.dir=/usr/local/jboss/jboss-as-7.1.1.Final/host02/
-Djboss.host.default.config=host-slave.xml
-Djboss.domain.master.address=192.168.238.186
-Djboss.bind.address.management=192.168.238.188
-Djboss.bind.address=192.168.238.188
-Djboss.server.name=host-02
```

Observando novamente o log Domain Controller e verifique que o Host Controller 02 se também conectou:

JBAS010918: Registered remote slave host "host02", JBoss AS 7.1.1.Final "Brontes"

No Mod Cluster Manager em http://192.168.238.186/mod_cluster-manager já estão as duas instâncias:



Para testar o Balanceamento vamos fazer o deploy da aplicação SystemProps que pode ser baixada <http://goo.gl/yjynK>. Para fazer o deploy conecte do domain controller utilizando o CLI:

```
[jboss@localhost jboss]$ sudo ./jboss-as-7.1.1.Final/bin/jboss-cli.sh -c controller=192.168.238.186:9999
```

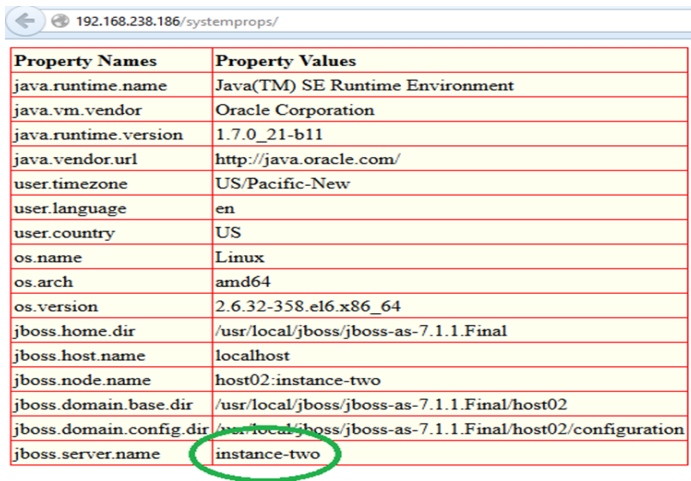
```
[domain@192.168.238.186:9999 /] deploy /tmp/systemprops.war --server-groups=apps
```

A aplicação foi deployada para as instâncias do server group apps. Acesse a url do balanceador e verifique se a aplicação está disponível: <http://192.168.238.186/systemprops/>

Property Names	Property Values
java.runtime.name	Java(TM) SE Runtime Environment
java.vm.vendor	Oracle Corporation
java.runtime.version	1.7_0_21-b11
java.vendor.url	http://java.oracle.com/
user.timezone	US/Pacific-New
user.language	en
user.country	US
os.name	Linux
os.arch	amd64
os.version	2.6.32-358.el6.x86_64
jboss.home.dir	/usr/local/jboss/jboss-as-7.1.1.Final
jboss.host.name	localhost
jboss.node.name	host01:instance-one
jboss.domain.base.dir	/usr/local/jboss/jboss-as-7.1.1.Final/host01
jboss.domain.config.dir	/usr/local/jboss/jboss-as-7.1.1.Final/host01/configuration
jboss.server.name	instance-one

Perceba que estamos na instance-one. Então para testarmos o funcionamento do balanceamento de carga vamos parar a Instância instance-one e quando tentarmos um novo acesso será redirecionado para o instance-two:

No CLI execute o seguinte comando para parar o instance-one:



Property Names	Property Values
java.runtime.name	Java(TM) SE Runtime Environment
java.vm.vendor	Oracle Corporation
java.runtime.version	1.7.0_21-b11
java.vendor.url	http://java.oracle.com/
user.timezone	US/Pacific-New
user.language	en
user.country	US
os.name	Linux
os.arch	amd64
os.version	2.6.32-358.el6.x86_64
jboss.home.dir	/usr/local/jboss/jboss-as-7.1.1.Final
jboss.host.name	localhost
jboss.node.name	host02:instance-two
jboss.domain.base.dir	/usr/local/jboss/jboss-as-7.1.1.Final/host02
jboss.domain.config.dir	/usr/local/jboss/jboss-as-7.1.1.Final/host02/configuration
jboss.server.name	instance-two

```
[domain@192.168.238.186:9999 /] /host=host01/server-config=instance-one:stop
{
  "outcome" => "success",
  "result" => "STOPPING"
}
```

Acesse novamente a aplicação <http://192.168.238.186/systemprops/>

Perceba que agora estamos no instance-two. Concluímos assim que o balanceamento de carga está funcionando.

Conclusão

O JBoss AS 7 trouxe a evolução arquitetural que era esperada como pode ser vista nesse artigo. Carregamento de módulos dinâmico, gerenciamento centralizado e tempo de startup reduzido são apenas

algumas das inúmeras melhorias trazidas nessa versão.

“**Milhares de aplicativos para Web e Mobile são desenvolvidos a cada mês utilizando esta plataforma como base. A plataforma Java mudou muito desde seu início e provavelmente vai continuar evoluindo nos próximos anos.**”

Espera-se que a plataforma JBoss continue evoluindo trazendo sempre mais benefícios a todos que utilizam os produtos desse ecossistema. Cada membro da comunidade também pode fazer a sua parte seja colaborando com artigos ou participando de discussões para encontrar melhorias para a plataforma. Cabe a nós aceitar o convite de melhorar cada vez mais essa tecnologia.

AUTOR



Escrito por **Mauricio Magnani Jr.**

Consultor JBoss com 6 anos de experiência no mercado. Trabalhou na RedHat como Engenheiro de Suporte JBoss atendendo clientes da América Latina e EUA. Criador e mantenedor do blog <https://jbossdivers.wordpress.com/> e membro ativo do Grupo de Usuários JBoss do Brasil. Hoje é Senior Java Software Engineer no Groupon Brasil. Possui as certificações Red Hat Certified JBoss Administrator on JBoss EAP 5 e JBoss EAP 6.

PHP: Uma framework “from scratch” (Parte 1)

Para o desenvolvimento de aplicações Web existem um sem número de frameworks gratuitas e mais ou menos bem conceituadas. Contudo temos vários problemas associados que com o crescer de uma aplicação podem trazer vários recuos no seu desenvolvimento. Um dos maiores é o óbvio: são gratuitas e por isso mesmo o código é lido por milhares de pessoas em todo o mundo tornando as falhas de segurança bastante visíveis e fáceis de explorar. Outro grande problema é o facto de trazerem funcionalidades específicas em que muitas vezes não as usamos, tornando-se em alguns casos um “canhão para matar um mosquito”. Pegando na abordagem das correntes frameworks, podemos nós próprios construir algo sólido o suficiente para uma grande aplicação.

Vamos nesta série de três artigos, construir uma framework passo-a-passo começando pelo básico mas o mais importante: o routing.

.htaccess

```
RewriteEngine on
RewriteCond $1 !^(allow\.php|index\.php|content)
RewriteRule ^(.*)$ /index.php/$1 [L]
<Files .htaccess>
order allow,deny
deny from all
</Files>
```

Tirando partido do serviço Apache, delegamos o enclausuramento do nosso sistema de ficheiros à regras htaccess. Não permitimos a listagem de ficheiros nem a execução de outros scripts PHP à excepção do index.php, sendo este o responsável pelo routing da framework. Para garantirmos que estas regras são cumpridas e que o index.php seja sempre chamado, redireccionamos todos os pedidos de execução de scripts para o mesmo index.php dando-lhe todos os argumentos do URL originalmente fornecidos. Desta forma o utilizador não sabe a que ficheiros ou pastas está a aceder. A esta regra temos de adicionar uma excepção: a pasta content onde temos todos os conteúdos como imagens, ficheiros CSS, vídeos etc. No fundo todos aqueles ficheiros que o browser do utilizador vai descarregar. Para mantermos o enclausuramento da nossa aplicação, não é aconselhado não colocarmos nessa pasta nenhum tipo de script, visto que pode ser executado arbitrariamente do exterior: o contrário do que queremos.

config.php

```
define('DS', DIRECTORY_SEPARATOR);
define('BASE_PATH', 'http://localhost/');
define('DEFAULT_PAGE', 'hello');
define('ROOT', $_SERVER['DOCUMENT_ROOT']);
```

Usaremos este ficheiro como auxilio para definições de caminhos para pastas e configurações por defeito.

- `define('DS', DIRECTORY_SEPARATOR);`

Esta definição é uma redundância, é definida simplesmente por conveniência na escrita do código.

- `define('BASE_PATH', 'http://localhost/');`

Aqui definimos o caminho URL para a nossa aplicação.

- `define('DEFAULT_PAGE', 'hello');`

Esta é a nossa página por defeito, caso nenhuma tenha sido requerida.

- `define('ROOT', $_SERVER['DOCUMENT_ROOT']);`

Aqui é definido o caminho para a pasta da aplicação.

Uma vez definidas as configurações, passamos à parte mais importante: o routing.

Visto que temos todos os requests redireccionados para o index.php, agora temos de fazer sentido dos argumentos que são passados pelo URL.

```
function routeURL(){
    $urlArray = array();
    $urlArray = explode("/", $_SERVER
        ["REQUEST_URI"]);
    array_shift($urlArray);
    $url = array();
    if (!isset($urlArray[0]) || $urlArray[0]==='')
    {
        $url['page'] = DEFAULT_PAGE;
        $url['args'] = array();
    } else {
        $url['page'] = $urlArray[0];
        array_shift($urlArray);
        if(count($urlArray)>0){
            $url['args'] = $urlArray;
        }else {
            $url['args'] = array();
        }
    }
    return $url;
}
```

Aqui separamos os argumentos em duas partes: a página que queremos aceder e os argumentos para esta página. O seguinte pedido:

http://localhost/hello/foo/bar

resulta em:

```
array (size=2)
  'page' => string 'hello' (length=5)
  'args' =>
    array (size=2)
      0 => string 'foo' (length=3)
      1 => string 'bar' (length=3)
```

De seguida, encaminhamos para a página requerida.

```
function callHook() {  
  
    $queryString = array();  
    $url = routeURL();  
  
    $page = $url['page'].".php";  
  
    if ( file_exists("pages". DS .$page) ) {  
        include_once("pages". DS .$page);  
    } else {  
        header('HTTP/1.0 404 Not Found');  
        exit;  
    }  
}
```

Aqui verificamos se a página que é pedida existe, no case de existir importamos o ficheiro caso contrario enviamos o header de erro e terminamos a aplicação.

Index.php

```
<?php  
require_once ('config.php');  
function routeURL(){  
    $urlArray = array();  
    $urlArray = explode("/",$_SERVER  
        ["REQUEST_URI"]);  
    array_shift($urlArray);  
    $url = array();  
    if (!isset($urlArray[0]) || $urlArray[0]=='')  
{  
        $url['page'] = DEFAULT_PAGE;  
        $url['args'] = array();  
    } else {  
        $url['page'] = $urlArray[0];  
        array_shift($urlArray);  
        if(count($urlArray)>0){  
            $url['args'] = $urlArray;  
        }else {  
            $url['args'] = array();  
        }  
    }  
    return $url;  
}  
function callHook() {  
  
    $queryString = array();  
    $url = routeURL();  
  
    $page = $url['page'].".php";  
  
    if ( file_exists("pages". DS .$page) ) {  
        include_once("pages". DS .$page);  
    } else {  
        header('HTTP/1.0 404 Not Found');  
        exit;  
    }  
}
```

```
callHook();  
?>
```

- ROOT
 - Index.php
 - Config.php
 - .htaccess
 - Pages
 - Hello.php

Com esta micro-framework temos um sistema de routing que nos irá permitir nos dois próximos artigos alargar os horizontes, passando a uma programação OOP e implementando MVC. Finalmente iremos implementar um pequeno driver para MySQL e um debugger.

“ (...)micro-framework temos um sistema de routing que nos irá permitir nos dois próximos artigos alargar os horizontes, passando a uma programação OOP e implementando MVC. ”

Dependências:

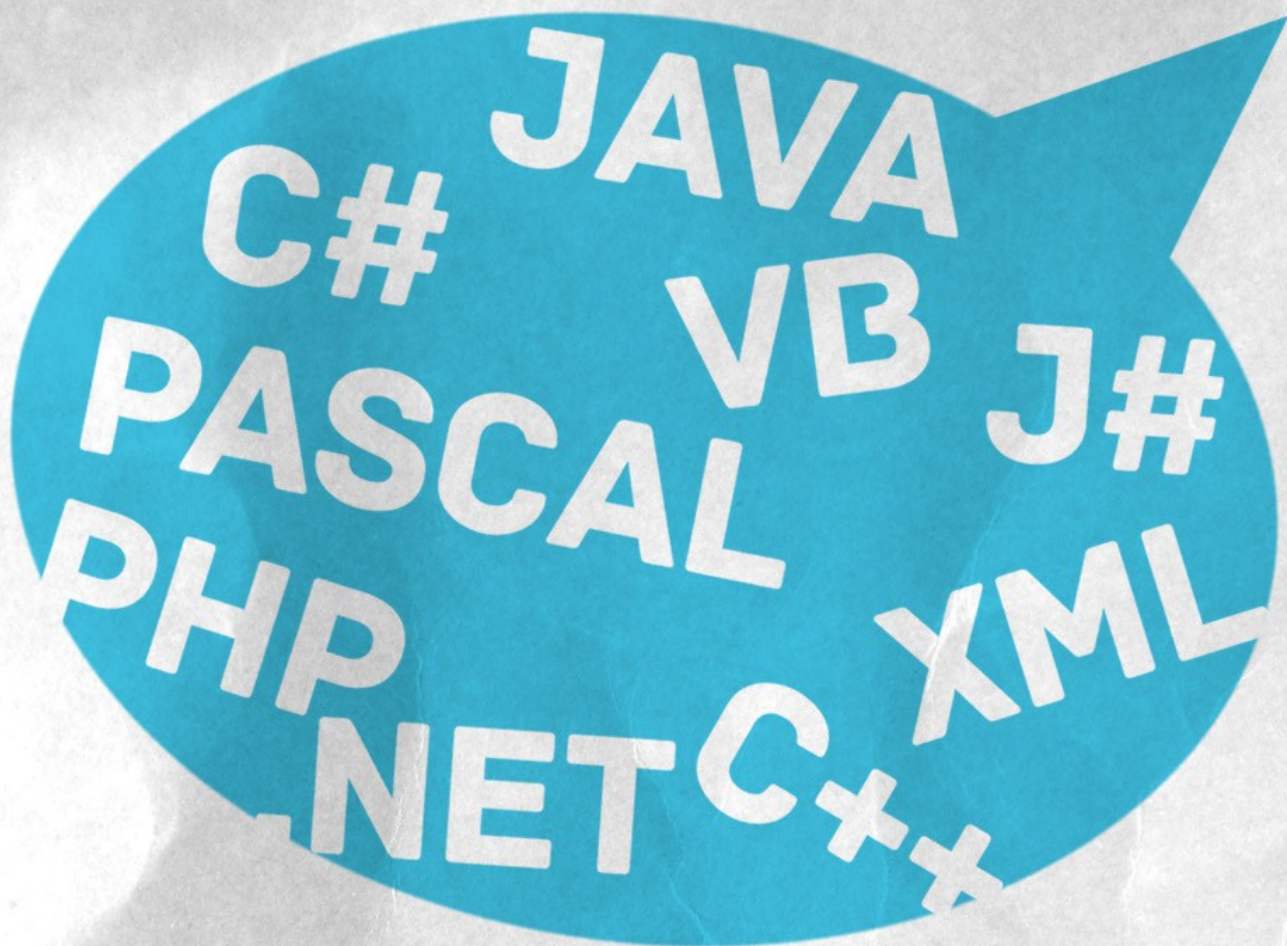
Apache com mod_rewrite e PHP.

AUTOR



Escrito por Ricardo Perre

Estuda Eng. da Computação Gráfica e Multimédia no Instituto Politécnico de Viana do Castelo, é programador focado em web, tendo bastante experiência em PHP. Foca-se em aplicações, e escalonamento das mesmas. Caso se resumisse seria: descobrir, inventar e fazer.



ENTÃO, SÓ FALAS
EM CÓDIGO?

TEMOS O REMÉDIO CERTO PARA TI!



portugal-a-programar.pt

A MAIOR COMUNIDADE PORTUGUESA DE
PROGRAMAÇÃO, APARECE!

COLUMNAS

C# - Observable vs. Task

Visual (Not) Basic - Operator Overloading

C# - OBSERVABLE VS. TASK

Nas edições anteriores vimos como transformar uma API baseada assíncrona em eventos numa API baseada em observáveis (observables) ou numa API baseada em tarefas.

A questão que se levanta é: quando usar qual?

Quando Usar Observáveis?

A programação baseada em observáveis é também conhecida como programação reativa e começou a ser usada como alternativa a assíncrona baseada em eventos porque a programação assíncrona baseada em tarefas ainda não estava disponível (ou não estava disponível em todas as plataformas).

Na verdade, o campo de batalha da programação reativa são as correntes de eventos (event streams). É aqui que a utilização de eventos se torna também mais natural.

Uma API baseada em eventos em que o evento apenas é disparado uma vez é, na verdade, uma API baseada em chamadas de resposta (callback). O facto de ser implementada com o mecanismo de eventos da plataforma .NET não faz dela uma API baseada em eventos.

A existência da necessidade de combinar várias correntes de eventos e relacioná-los entre si é também uma boa indicação para a utilização de observáveis.

Quando Usar Tarefas?

Como o nome indica, uma tarefa é algo que tem um início e um fim. Assim sendo, todas as API assíncronas baseadas num mecanismo de chamada de resposta (callback) são candidatas a ser transformadas numa API assíncrona baseada em tarefas. Mesmo que o mecanismo de chamada de resposta sejam eventos .NET.

Matriz De Aplicabilidade

Baseando-nos nas considerações anteriores, e porque nem sempre tudo é assíncrono, chegamos à seguinte matriz de aplicabilidade:

	Síncrono	Assíncrono
Um único valor	-	Tarefa
Múltiplos valores	Enumerável	Observável

Conclusão

No final não existe uma resposta mágica. O que numa situação pode ser um observável noutra pode ser uma tarefa.

“ o campo de batalha da programação reativa são as correntes de eventos (event streams) ”

Recursos

[O GeoCoordinateWatcher Como Um Serviço Reativo - Revista PROGRAMAR - 39ª Edição - Fevereiro 2013](#)

[Trazendo Async E Await ao Serviço de Contactos do Windows Phone - Revista PROGRAMAR - 40ª Edição - Abril 2013](#)

[Extensões Reativas \(Rx\)](#)

[Programação assíncrona com async e await \(C# e Visual Basic\)](#)

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

VISUAL (NOT) BASIC

OPERATOR OVERLOADING

“Não podes somar batatas com elefantes!” Frases como a anterior, ou semelhantes, já vos passaram por a consideração em alguma altura da vossa vida. Quer tenha sido nos vossos primeiros passos a matemática, ou ainda na semana passada em qualquer implementação de software mais rebuscada, alguém, algures transmitiu dessa forma que não é possível operar aquelas duas variáveis. De facto, batatas e elefantes não são tipicamente operáveis a menos que a resposta de que estamos à procura seja diferente.

Saber como operar

A solução da soma de batatas com elefantes reside precisamente na resposta que procuramos obter. Eu posso dizer que Elefante + Batata é igual a 4, se for especificado que a soma de seres-vivos opera aritmeticamente o número de pernas do ser. Não posso? Claro que posso! Só precisamos de definir que os seres-vivos são operáveis assim, e de que forma. É aqui que entra o “overload” de operadores.

Operadores?

Operadores, sim. Quando se falam de operadores, referem-se todos aqueles que estão provavelmente a passar-vos na cabeça agora, tanto unários como binários, mas nem todos:

Unários: + - Not IsTrue IsFalse CType

Binários: + - * / \ & Like Mod And Or Xor ^ << >> = <> > < >= <=

Estes operadores fazem sentido nos usos comuns, onde já sabemos para que servem e que resultados vão surtir. À luz da especificação anterior, podemos afirmar com segurança que Elefante > Batata, porque uma batata não tem pernas, e porque conhecemos o operador > (maior que). A implementação do operador pode basear-se em qualquer aspecto dos seus operandos, mas não é boa ideia contradizer a lógica do operador que se está a definir. Por razões óbvias, não é boa ideia, por exemplo, implementar overloads de tal forma que Batata - Elefante = -4 ao mesmo tempo que Batata > Elefante = TRUE. Não faz sentido porque não estamos a usar o mesmo critério. Se Batata - Elefante é menor que zero, a Batata deveria ser menor que o Elefante e não o contrário.

Não é que seja impossível, mas vai tornar o código muito mais difícil de ler, o que vai contra a facilidade do overload de operadores.

Overload?

A ideia não é inventar novos operadores. O que queremos realmente indicar é qual o procedimento a seguir para conseguir aplicar o análogo do operador para determinado tipo ou

determinados tipos. Assim, o operador +, por exemplo, terá de continuar a fazer sentido na aritmética ao mesmo tempo que faz sentido na soma de seres-vivos: se tentarmos somar números, entra o operador aritmético; se tentarmos somar batatas com elefantes, entra a nossa implementação do operador.

Portanto, e em tom de resumo, da mesma forma que definimos overloads de métodos, também se definem overloads de operadores, com base na sua assinatura.

Já chega de elefantes e batatas, não?

Efectivamente já. Não convém esticar a analogia, até porque nem todos os operadores fazem sentido com uma classe de seres vivos. As coisas começam a ficar mais claras com exemplos de implementação.

Vamos considerar a seguinte classe para exemplificar:

```
Public Class Veiculo
    Public Enum TipoVeiculo
        AUTOMOVEL
        MOTO
        PESADO
    End Enum

    Public Enum TipoCombustivel
        GASOLINA
        GASOLEO
        HIBRIDO
        ELECTRICO
        GPL
    End Enum

    Public Property Tipo As TipoVeiculo
    Public Property Marca As String
    Public Property Modelo As String
    Public Property Cilindrada As Integer
    Public Property Combustivel As TipoCombustivel
    Public Property NumeroPortas As Short

    Sub New(Tipo As TipoVeiculo,
            Marca As String,
            Modelo As String,
            Cilindrada As Integer,
            Combustivel As TipoCombustivel,
            NumeroPortas As Short)
        _Tipo = Tipo
        _Marca = Marca
        _Modelo = Modelo
        _Cilindrada = Cilindrada
        _Combustivel = Combustivel
        _NumeroPortas = NumeroPortas
    End Sub
End Class
```

Como será de esperar, se tentarmos por exemplo comparar instâncias desta classe, o Visual Basic não saberá o que fazer para comparar. É um tipo composto que compreende

VISUAL (NOT) BASIC

OPERADOR LIKE

membros que nós mesmos criamos e seria impossível perceber automaticamente quais as características a comparar.

Este código não compila com dois erros em Carro1 = Carro2 e Carro2 = Carro3:

```
Dim Carro1 As New Veiculo(AUTOMOVEL,
    "Ford", "Ka", 1249, GASOLINA, 3)
Dim Carro2 As New Veiculo(AUTOMOVEL,
    "Ford", "Ka", 1250, GASOLEO, 3)
Dim Carro3 As New Veiculo(AUTOMOVEL,
    "Ford", "Ka", 1251, GASOLEO, 3)
Debug.WriteLine("Carro1 = Carro2: " & (Carro1 =
    Carro2))
Debug.WriteLine("Carro2 = Carro3: " & (Carro2 =
    Carro3))
```

Operator '=' is not defined for types 'OperatorOverload.Veiculo' and 'OperatorOverload.Veiculo'.

E tem toda a razão. O operador igual (=) não está nem implementado na classe, nem tem nenhum overload que implique uma operação binária com dois tipos Veiculo. Vamos então implementar um operador para o igual (=) que consiga determinar se estamos na presença do mesmo veículo ou não. Vamos considerar que a cilindrada não é um factor para distinguir dois carros.

Basta acrescentar os seguintes métodos na classe Veiculo:

```
Public Shared Operator =(V1 As Veiculo, V2 As
    Veiculo)
    If V1.Tipo <> V2.Tipo Then Return False
    If V1.Combustivel <> V2.Combustivel Then
        Return False
    If Not V1.Marca.ToLower.Equals
        (V2.Marca.ToLower) Then Return False
    If Not V1.Modelo.ToLower.Equals
        (V2.Modelo.ToLower) Then Return False
    If V1.NumeroPortas <> V2.NumeroPortas Then
        Return False
    Return True
End Operator

Public Shared Operator <>(V1 As Veiculo, V2
    As Veiculo)
    If V1.Tipo <> V2.Tipo Then Return True
    If V1.Combustivel <> V2.Combustivel Then
        Return True
    If Not V1.Marca.ToLower.Equals
        (V2.Marca.ToLower) Then Return True
    If Not V1.Modelo.ToLower.Equals
        (V2.Modelo.ToLower) Then Return True
    If V1.NumeroPortas <> V2.NumeroPortas Then
        Return True
    Return False
End Operator
```

Algumas implementações de operadores implicam que se implementem também a sua negação. No caso do operador igual (=) é necessário não só especificar o que é considerado igual, mas também o que é considerado diferente.

Assim, o código anterior já é correctamente compilado e produz o seguinte output:

Carro1 = Carro2: False

Carro2 = Carro3: True

O que é verdade. Como operações aritméticas não faziam muito sentido com uma classe a representar um veículo, vamos considerar esta nova classe:

```
Public Class Vector3
    Public Property X As Decimal
    Public Property Y As Decimal
    Public Property Z As Decimal

    Sub New(X As Decimal, Y As Decimal, Z As
        Decimal)
        _X = X
        _Y = Y
        _Z = Z
    End Sub

    Public Overrides Function ToString() As String
        Return String.Format("{0},{1},{2}", _X,
            _Y, _Z)
    End Function
End Class
```

Esta classe representa um vector tridimensional. Mantém um valor para X, para Y e para Z. Não adianta realizar operações aritméticas com o Vector, uma vez que, novamente, o Visual Basic não saberia o que fazer com ele. Vamos então acrescentar alguns operadores:

```
Public Shared Operator +(V1 As Vector3, V2 As
    Vector3)
    Return New Vector3(V2.X + V1.X, V2.Y + V1.Y,
        V2.Z + V1.Z)
End Operator

Public Shared Operator -(V1 As Vector3, V2 As
    Vector3)
    Return New Vector3(V2.X - V1.X, V2.Y - V1.Y,
        V2.Z - V1.Z)
End Operator

Public Shared Operator *(V1 As Vector3, V2 As
    Vector3)
    Return (V2.X * V1.X) + (V2.Y * V1.Y) + (V2.Z *
        V1.Z)
End Operator

Public Shared Operator *(V1 As Vector3, E1 As
    Decimal)
    Return New Vector3(E1 * V1.X, E1 * V1.Y, E1 *
        V1.Z)
End Operator

Public Shared Operator &(S1 As String, V1 As
    Vector3)
    Return S1 & V1.ToString
End Operator
```

Assim, já é possível efectuar algumas operações com esta classe:

```
Dim V1 As New Vector3(3, 2, 1)
Dim V2 As New Vector3(1, 2, 3)
Debug.WriteLine("Adição: " & (V1 + V2))
Debug.WriteLine("Subtração: " & (V1 - V2))
Debug.WriteLine("Produto escalar: " & (V1 * V2))
Debug.WriteLine("Multiplicação com escalar: " &
    (V2 * 3))
```

VISUAL (NOT) BASIC

OPERADOR LIKE

O que produz:

Adição: (4,4,4)

Subtração: (-2,0,2)

Produto escalar: 10

Multiplicação com escalar: (3,6,9)

Foi necessário especificar o operador de concatenação, para que o Visual Basic saiba como tratar a concatenação de uma String com o resultado de uma outra operação com a mesma classe.

Neste caso estamos apenas a concatenar a String com a representação String da classe, que definimos anteriormente. Ao analisar os operadores, podemos verificar que existem dois overloads para a multiplicação.

Um implica dois vectores e o outro um vector e um decimal. A nossa implementação dita que se forem fornecidos dois vectores, o resultado será o produto escalar dos dois vectores.

Se for fornecido apenas um vector e um valor escalar, o resultado será outro vector que representa o produto desse vector com um escalar.

Existe com certeza aplicações matemáticas (e não só) mais interessantes, mas as apresentadas demonstram a facilidade

com que implementamos os nossos próprios comportamentos de operadores.

“ Operator overloading permite uma notação mais próxima do domínio alvo. ”

Conclusão

Operator overloading permite uma notação mais próxima do domínio alvo, o que promove a legibilidade e dissipa dúvidas de interpretação. Esta é a sua maior vantagem. No entanto, se as implementações não forem de encontro ao sentido natural dos operadores, o efeito poderá ser o perfeito contrário.

É um caso típico de melhor amigo ou pior inimigo. Não há meio-termo.

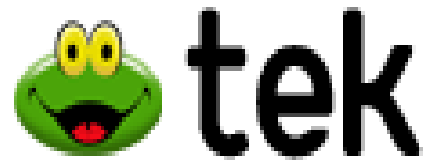
AUTOR



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>

Media Partners da Revista PROGRAMAR



EVENTO PROGRAMAR 2013



No passado dia 25 de Maio de 2013 realizou-se na Microsoft Portugal, no Parque das Nações, o 1º evento presencial da comunidade Portugal a Programar. Ainda antes das 9h começaram a chegar os participantes, crescia o entusiasmo envolto em todo o evento e, pouco a pouco, todos nós fomos dando cara aos nicks a que nos habituámos ao longo dos anos.



Pontualmente, a organização deu início ao evento PROGRAMAR 2013 com Jorge Paulino, David Pintasilgo e Rui Gonçalves a abrir o evento. Brevemente foi-nos dada a conhecer a comunidade P@P, assim como os quatro pontos-chave que a caracterizam:

- Fórum P@P
- Wiki
- Portal de Downloads
- Revista Programar

Parafraseando o anfitrião Jorge Paulino, “este foi um projeto que se iniciou a 28 de Maio de 2005 para dar resposta à necessidade de partilhar informação e experiências nesta área”. Após a apresentação do evento e da comunidade seguiu-se um discurso inspirador de um dos oradores, Rui Delgado, acerca da necessidade de se empreender em Portugal.

Com o limite de inscrições a ser atingido em apenas três dias, o 1º evento do P@P contou com a presença de 19 oradores voluntários que deram origem a 18 apresentações técnicas e a 2 workshops práticos. Cerca de 250 participantes estiveram presentes no edifício da Microsoft Lisbon Experience.



Pelas diversas salas do evento partilharam-se experiências e conhecimento. As **apresentações** decorreram 3 em simultâneo e foram as seguintes:

- Apresentação das novidades de Java EE 7 (Ernest Duarte)
- As novidades do C# 5.0 (Paulo Morgado)
- ASP.NET SignalR – Comunicação em real-time simples e para todo mundo (Glauco Godoi)
- Conquistar o mundo com aplicações feitas à velocidade da luz (Ricardo Marques)
- CRM? Como escolher? De raiz ou sistema já desenvolvido? (Pedro Azevedo)
- Desenvolvimento de Aplicações em Windows Phone 8 (Nuno Silva)
- Desenvolvimento em SharePoint, por onde começar (Rodrigo Pinto)
- Desenvolvimento rápido de sítios web com personalização de Joomla (Rui Guimarães)
- DMVs – Conhece o teu SQL Server (Vitor Pombeiro)
- Empreendedorismo em TI (Fernando Martins)
- HTML5 e CSS3 – rápido e eficaz para o presente (Sérgio Laranjeira)
- Introdução ao Cloud Computing e ao Windows Azure (Vitor Tomaz)
- Microsoft Kinect SDK (Rui Simão)
- Plataforma de desenvolvimento para Windows Store Apps (Nuno Silva)
- Powershell "à minha maneira" (Bruno Lopes)
- O desenvolvimento de aplicações móveis, antes da 1ª linha de código (Alberto Silva)
- SEO – A importância do Search Engine Optimization (Miguel Lobato)
- SQL Server – Performance e Tuning (Pedro Martins)

E ainda os **workshops práticos**:

- Workshop – Integração de CRM Dynamics com Java e .NET (Pedro Azevedo)
- Workshop – Web em Realtime (Sérgio Costa)



EVENTOS



Durante o decorrer das sessões houve ainda oferta de brindes (gentilmente oferecidos pelos patrocinadores) aos participantes.

Outro ponto que marcou este PROGRAMAR 2013 foi o concurso de aplicações Windows Phone e Windows 8, vencido por Jorge Costa com o jogo “Little Bits”, sendo o segundo lugar atribuído a Mauro Castro pela aplicação “Portugal’s Evenings”.



Houve ainda demonstrações do Microsoft Kinect (Rui Simão) e do Leap Motion (Diniz Vieira) a todos os participantes interes-

sados, e todo o dia foi passado num espírito descontraído e animado, aumentando assim os laços entre toda a comunidade.

A toda a organização e a todos os oradores envolvidos deixamos um agradecimento especial, agradecendo também a todos os patrocinadores nomeadamente a Microsoft Portugal, ESET Portugal, FCA Editora, Rumos e Truwind-Chiron. Nunca deixando de referir também os Media Parters que ajudaram a toda a divulgação deste evento.



A todos os elementos da comunidade que fazem com que a Portugal-a-Programar seja cada vez mais conhecida no nosso país e além-fronteiras, a todos deixamos um “Muito Obrigado”, relembrando mais uma vez que este é um projeto de todos e para todos, e que a mais pequena ação pode fazer a diferença.

“ (...) a mais pequena ação pode fazer a diferença. ”

É com satisfação e orgulho que podemos afirmar que o 1º Evento Presencial do P@P foi um sucesso, pois mesmo após as sessões terem terminado, houve ainda quem se mantivesse no local partilhando experiências de um dia cheio de emoções.

E citando um dos oradores do evento, Rui Delgado, “Oh Yes! Muito Bom!”... Venha o PROGRAMAR 2014!

Página do evento: <http://evento.portugal-a-programar.pt/>

Análises

Android – Introdução ao Desenvolvimento de Aplicações

C# 5.0 com Visual Studio 2012

Android – Introdução ao Desenvolvimento de Aplicações

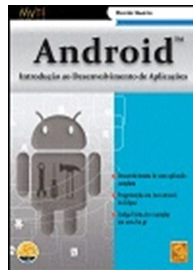
Título: Android – Introdução ao Desenvolvimento de Aplicações

Autor: Ricardo Queirós

Editora: FCA

Páginas: 224

ISBN: 978-972-722-763-1



O Android tornou-se na plataforma de dispositivos móveis mais usada no mundo e, como tal, existe um interesse crescente dos programadores em criarem soluções para esse mercado. Para quem queira dar aquele 1.º passo no desenvolvimento de aplicação em Android, o livro “Android - Introdução ao desenvolvimento de aplicações”, de Ricardo Queirós, é uma boa escolha.

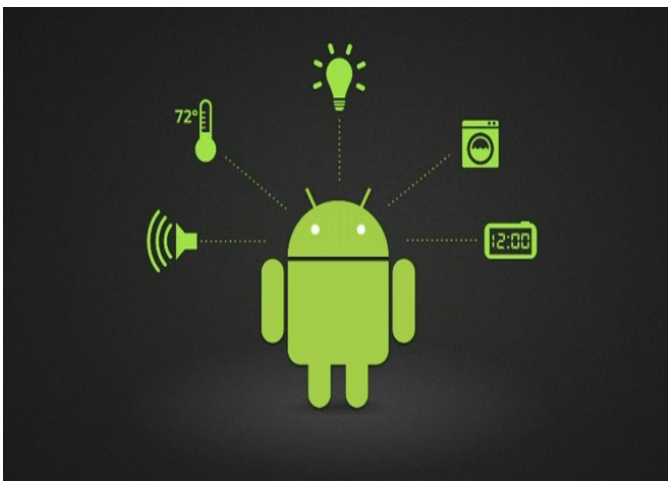
Contém, passo a passo, desde a configuração do ambiente de desenvolvimento, até à concepção de uma aplicação para Android na sua íntegra. Destaca exemplos muito úteis que podem ser reaproveitados noutra aplicação, pois contempla o desenho de uma interface gráfica, a gestão de dados da aplicação através de gestão de ficheiros ou bases de dados, utilização de APIs de localização, mapas e sensores e a criação do jogo do galo.

Infelizmente já é habitual que nos sejam apresentados os típicos tons acinzentados das ilustrações e a ausência de qualquer tipo de realce de sintaxe no código-exemplo, o que por vezes pode-se tornar cansativo ao olhar.

Um factor alheio ao autor do livro, mas que a editora em causa deveria ter em consideração por forma a facilitar a leitura.

“ (...) destaca exemplos muito úteis que podem ser reaproveitados noutra aplicação (...) ”

“Android – Introdução ao Desenvolvimento de Aplicações”, de Ricardo Queirós, é um bom livro para quem já tenha alguma experiência em programação Java e que queira explorar as potencialidades do Android.



Durante a sua leitura pode-se constatar que é um livro rico em imagens explicativas e exemplificativas que nos ajudam a perceber o que devemos fazer e o que devemos alcançar com a construção de todo o código escrito.



AUTOR



Escrito por José Marques

Natural de Coruche, formado em Gestão e programação de sistemas informáticos e técnico especialista de gestão de redes. Membro do P@P desde Abril de 2013.

C# 5.0 com Visual Studio 2012

Título: C# 5.0 com Visual Studio 2012

Autor: Henrique Loureiro

Editora: FCA

Páginas: 608

ISBN: 978-972-722-752-5



Elementos de programação

Abordam-se os temas de programação mais elementar, com referências à programação iterativa passando pela orientada a objecto. O texto está nitidamente orientado para um público que tem noções básicas de programação explicando a lógica de uma função até à herança de classes. Tenta de uma forma sucinta demonstrar na prática, ou seja com code-snippets, como construir de uma forma coerente uma aplicação em C# recorrendo ao Visual Studio.

Usabilidade - É também abordada a importância da usabilidade do user-interface numa qualquer aplicação. São demonstrados os conceitos básicos de Interação Homem-Máquina e como aplica-los aos Windows Forms. Este tema serve de ponte para um tema bem mais extensivo: Windows Forms e programação orientada à GUI, detalhando alguns dos componentes que a .NET traz built-in. Explica a funcionalidade e a interacção com os componentes mais genéricas e mais usuais para uma aplicação que se use de um sistema de janelas como GUI. A explicação embora limitada a poucos elementos é bastante completa, dando ênfase aos events associados a este tipo de modulos explicando assim a programação assíncrona em Windows Forms.

Modelação de dados e LINQ - Temos uma explicação bottom-to-top da noção de modelação de dados. Sendo neste tema um pouco mais extenso explicando toda a noção da Modelação e das Formas Normais. Recorre-se ao SQL Server para fazer toda a explicação na prática dos conceitos, sendo bastante ilustrativo com diagramas e screenshots de um processo de criação e normalização de uma base de dados. Tendo a base de dados criada no SQL Sever, passa à explicação do que é o LINQ e sua sintaxe em C#. Aqui é feita mais uma vez a ponte com Windows Forms, utilizando componentes anteriormente demonstrados mapeando tabelas e manipulações básicas. Outra vez grande ênfase aos events.

Sistema Operativo e Input Output Começando pela base, expõe passo-a-passo como manipular as Tarefas e Processos do Windows, explicando também como colmatar as diferenças entre versões do sistema operativo. Dá exemplos de como manipular ficheiros de texto dando uma introdução à encriptação que vem built-in na .NET. Mostra como se usa o Graphics da .NET explicando os seus conceitos base: desenho e animação, chegando a à impressão destes mesmos gráficos desenhados. Com este tema aprofunda a explicação dos Windows Forms, fazendo a ponte para WPF.

WPF, XML e XAML - Tendo explicado Windows Forms, passa à explicação das diferenças e semelhanças do mesmo com WPF, dando também a sua opinião sobre o futuro das mesmas dando a entender que WPF irá substituir Windows Forms eventualmente. Para o efeito começa por explicar o que é XML dando exemplos de aplicações em C# .NET para manipulação deste tipo de dados, passando para um nível acima: o XAML. Exemplo atrás de exemplo, demonstra como usar as potencialidade do WPF mostrando como usar Graphics com a sintaxe XAML, incluindo animação, WebViews, views modelares e navegação entre diferentes ecrãs numa só janela.

Integração com Microsoft Office e Windows Store - É também mostrada a facilidade com que se integra uma aplicação feita em C# com as principais ferramentas do MS Office: Word, Excel, PowerPoint, Access e Outlook. Demonstra a semelhança de outras ferramentas Microsoft, que há sempre uma Class built-in na .NET para criar e manipular este tipo de documentos da família Microsoft. No caso da Windows Store dá exemplos vocacionados para Windows 8 e como tirar partido do conceito visual Metro e seus layouts genéricos, fazendo aplicações como uma calculadora e uma WebView.

Opinião pessoal - Na minha opinião este livro é ideal no mundo académico, servindo perfeitamente como suporte a um professor para leccionar uma cadeira de programação OOP, ou até para programação 2D. Também não acharia descabido servir como suporte a um aluno para evoluir da programação iterativa para OOP visto que o livro está repleto de exemplos e exercícios, tendo até no final 3 projectos from scratch de aplicações completas explicando todo o processo lógico na elaboração de todos os aspectos da mesma: de base de dados normalizada, interacção Homem-Máquina a comercialização na Windows Store.

AUTOR



Escrito por Ricardo Perre

Estuda Eng. da Computação Gráfica e Multimédia no Instituto Politécnico de Viana do Castelo, é programador focado em web, tendo bastante experiência em PHP. Foca-se em aplicações, e escalonamento das mesmas. Caso se resumisse seria: descobrir, inventar e fazer.

COMUNIDADES

Comunidade NetPonto – Telerik RadControls - Rápida implementação da página Sobre para Windows Phone Apps

TELERIK RADCONTROLS - RÁPIDA IMPLEMENTAÇÃO DA PÁGINA SOBRE PARA WINDOWS PHONE APPS

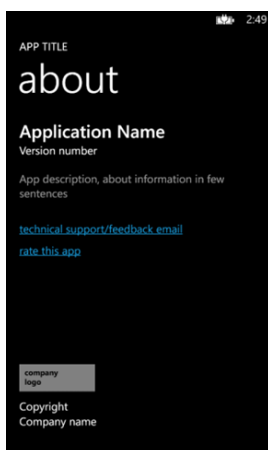
Introdução

A Telerik RadControls disponibiliza um projeto template que permite escolher um conjunto de funcionalidade que serão incluída no projeto, aquando da sua criação. A página "Sobre" é uma delas.

Truque: A Telerik oferece uma versão de teste, que permite explorar os controlos. O programa Nokia Premium Developer permite aos seus membros terem uma licença válida destes!

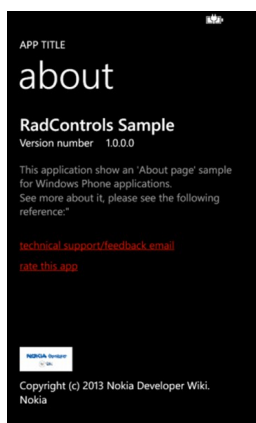
Ecrãs

Para compreender melhor o que iremos fazer, primeiro iremos criar a página "Sobre" que é fornecida pelo projeto template da Telerik, cujo resultado é:



Página "Sobre", gerada por omissão pelo projeto template da Telerik

Depois, iremos customizar a página para que esta tenha a sua própria informação, o resultado será:

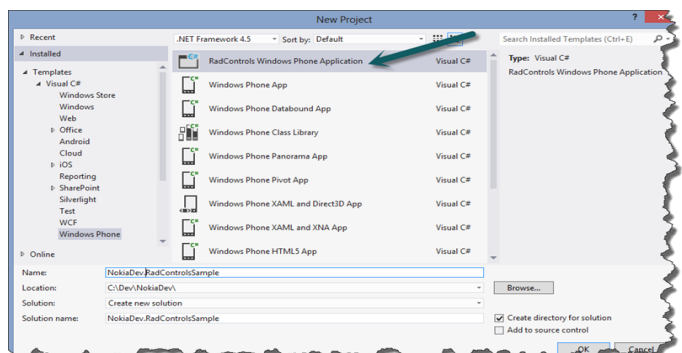


Customização da página "Sobre"

Criação do projeto

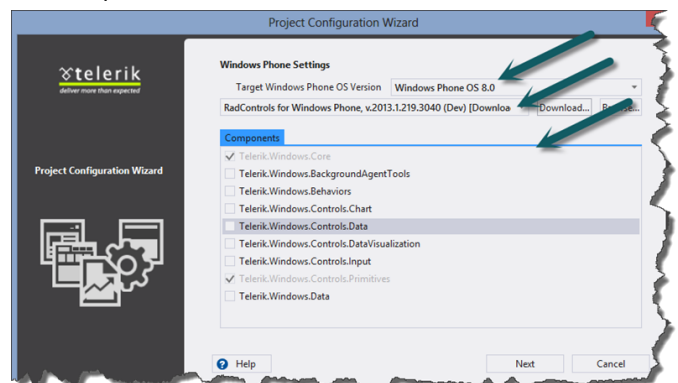
Para começar o projeto é recomendado que se instale os Telerik RadControls (versão de teste).

1. Seleção do projeto que se irá criar:



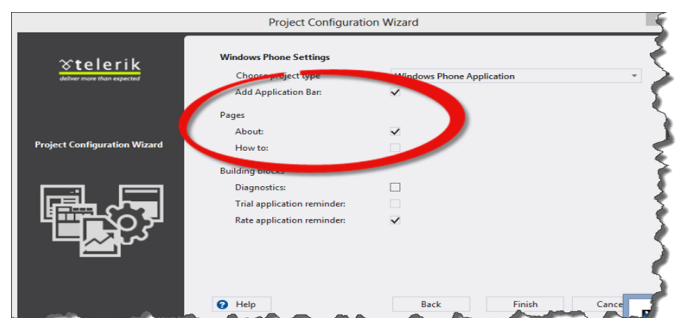
Criando o projeto.

2. A Telerik tem uma interface para a seleção inicial. Selecione a plataforma e referências da Telerik que queremos:



Selecionando a plataforma e as referências.

3. Seleção da funcionalidade página "Sobre":



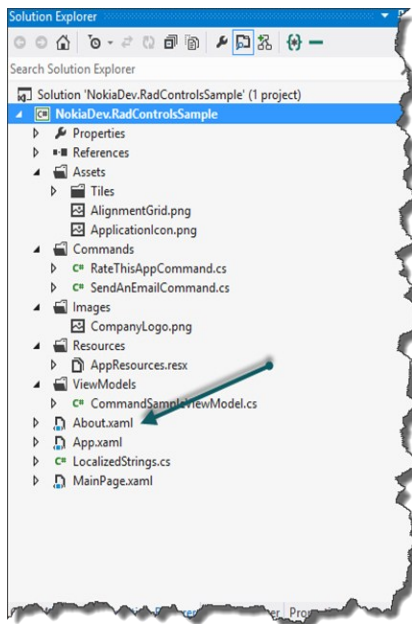
Quando o projeto é criado iremos obter a página por omissão podemos ver nos ecrãs.

COMUNIDADE NETPONTO

<http://netponto.org>

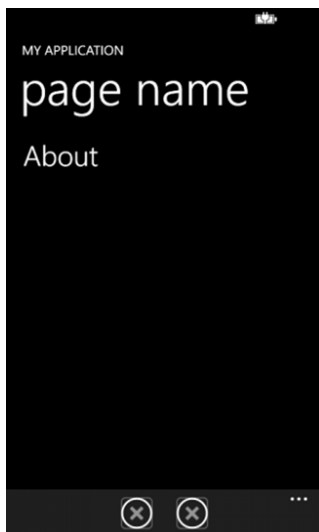
PÁGINA AVANÇADA "SOBRE" PARA APLICAÇÕES DE WINDOWS PHONE

O resultado da estrutura do projeto será:



Customização

Começemos por analisar a página Home Page



Cujo XAML é

```
<phone:PhoneApplicationPage
x:Class="NokiaDev.RadControlsSample.MainPage"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/
```

```
markup-compatibility/2006"

xmlns:phone=
"clr-namespace:Microsoft.Phone.
Controls;assembly=Microsoft.Phone"

xmlns:shell=
"clr-namespace:Microsoft.
Phone.Shell;assembly=Microsoft.Phone"

FontFamily=
"{StaticResource PhoneFontFamilyNormal}"

FontSize=
"{StaticResource PhoneFontSizeNormal}"

Foreground=
"{StaticResource PhoneForegroundBrush}"

Orientation="Portrait"

SupportedOrientations="Portrait"

shell:SystemTray.IsVisible="True"

mc:Ignorable="d">
```

```
<!-- LayoutRoot is the root grid where all
page content is placed -->
```

```
<Grid x:Name="LayoutRoot"
Background="Transparent">

<Grid.RowDefinitions>

<RowDefinition Height="Auto" />

<RowDefinition Height="*" />

</Grid.RowDefinitions>
```

```
<!--
```

LOCALIZATION NOTE:

To localize the displayed strings copy their values to appropriately named

keys in the app's neutral language resource file (AppResources.resx) then replace the hard-coded text value between the attributes' quotation marks with the binding clause whose path points to that string name.

For example:

```
Text="{Binding
Path=LocalizedResources.ApplicationTitle, Source=
{StaticResource LocalizedStrings}}"
```

This binding points to the template's string resource named "ApplicationTitle".

Adding supported languages in the Project Properties tab will create a new resx file per language that can

```

carry the translated values of your
    UI strings. The binding in these examples
will cause the value of the
    attributes to be drawn from the .resx
file that matches the
    CurrentUICulture of the app at run time.
-->

<!-- TitlePanel contains the name of the
    application and page title -->

<StackPanel x:Name="TitlePanel"

    Grid.Row="0"

    Margin="12,17,0,28">

    <TextBlock x:Name="ApplicationTitle"

        Style="{StaticResource
PhoneTextNormalStyle}"

        Text="MY APPLICATION" />

    <TextBlock

        x:Name="PageTitle"

        Margin="9,-7,0,0"

        Style="{StaticResource
PhoneTextTitle1Style}"

        Text="page name" />

</StackPanel>

<!-- ContentPanel - place additional
    content here -->

<Grid x:Name="ContentPanel"

    Grid.Row="1"

    Margin="12,0,12,0">

    <ListBox Margin="14,0,-12,0"

        FontFamily="{StaticResource
PhoneFontFamilySemiLight}"

        FontSize="{StaticResource
PhoneFontSizeExtraLarge}">

        <ListBoxItem Content="About"

            Tap="GoToAbout" />

    </ListBox>

</Grid>

<!-- Uncomment to see an alignment grid to help
ensure your controls are aligned on common bound-
aries. The image has a top margin of -32px to
account for the System Tray. Set this to 0 (or
remove the margin altogether)
if the System Tray is hidden.

Before shipping remove this XAML and the image
itself.-->

<!-- <Image Source="/Assets/AlignmentGrid.png"
VerticalAlignment="Top" Height="800" Width="480"
Margin="0" Grid.Row="0" Grid.RowSpan="2" IsHit-
TestVisible="False" /> -->

</Grid>
<!-- Sample code showing usage of ApplicationBar

```

```

->

<phone:PhoneApplicationPage.ApplicationBar>

    <shell:ApplicationBar IsMenuEnabled="True"

        IsVisible="True">

        <shell:ApplicationBarIconButton

            IconUri="/Images/appbar_button1.png"

            Text="Button 1" />

        <shell:ApplicationBarIconButton

            IconUri="/Images/appbar_button2.png"

            Text="Button 2" />

        <shell:ApplicationBar.MenuItems>

            <shell:ApplicationBarMenuItem

                Text="MenuItem 1" />

            <shell:ApplicationBarMenuItem

                Text="MenuItem 2" />

        </shell:ApplicationBar.MenuItems>

    </shell:ApplicationBar>

</phone:PhoneApplicationPage.ApplicationBar>

</phone:PhoneApplicationPage>
Para navegar para a página "Sobre" é necessário
clicar na palavra "About" (tap gesture), o código
referente é:
<ListBoxItem Content="About" Tap="GoToAbout" />
E em code behind iremos ter:
    /// <summary>

    /// Navigates to about page.

    /// </summary>

    private void GoToAbout(object sender,
        GestureEventArgs e)

    {

        this.NavigationService.Navigate(new Uri
("/About.xaml", UriKind.RelativeOrAbsolute));

    }

Por fim, a página AboutPage será:
O código XAML será:
<phone:PhoneApplicationPage
x:Class="NokiaDev.RadControlsSample.About"

    xmlns="http://
schemas.microsoft.com/winfx/2006/xaml/presentation"

    xmlns:x="http://
schemas.microsoft.com/winfx/2006/xaml"

    xmlns:d="http://
schemas.microsoft.com/expression/blend/2008"

    xmlns:mc="http://
schemas.openxmlformats.org/
markup-compatibility/2006"

    xmlns:phone="clr-namespace:
Microsoft.Phone.Controls;assembly=Microsoft.Phone"

    xmlns:shell="clr-namespace:

```

COMUNIDADE NETPONTO

<http://netponto.org>

PÁGINA AVANÇADA “SOBRE” PARA APLICAÇÕES DE WINDOWS PHONE

```
Microsoft.Phone.Shell;assembly=Microsoft.Phone"

    xmlns:viewModels="clr-namespace:
    NokiaDev.RadControlsSample.ViewModels"
    Name="Root"
    FontFamily=
    "{StaticResource PhoneFontFamilyNormal}"
    FontSize=
    "{StaticResource PhoneFontSizeNormal}"
    Foreground=
    "{StaticResource PhoneForegroundBrush}"
    Orientation=
    "Portrait"
    SupportedOrientations="Portrait"
    d:DesignHeight="768"
    d:DesignWidth="480"
    shell:SystemTray.IsVisible="True"
    mc:Ignorable="d">
<Grid x:Name="LayoutRoot"
    Background="Transparent">
    <Grid.DataContext>
    <viewModels:CommandSampleViewModel />
    </Grid.DataContext>
    <Grid.RowDefinitions>
    <RowDefinition Height="Auto" />
    <RowDefinition Height="*" />
    </Grid.RowDefinitions>

    <StackPanel x:Name="TitlePanel"
    Grid.Row="0"
    Margin="12,17,0,28">
    <TextBlock x:Name="ApplicationTitle"
    Style="{StaticResource
    PhoneTextNormalStyle}"
    Text="APP TITLE" />
    <TextBlock x:Name="PageTitle"
    Margin="9,-7,0,0"
    Style=
    "{StaticResource PhoneTextTitle1Style}"
    Text="about" />
    </StackPanel>
    <Grid x:Name="ContentPanel"
    Grid.RowSpan="2"
    Margin="12,160,12,1">
    <Grid.RowDefinitions>
    <RowDefinition Height="Auto" />
    <RowDefinition Height="Auto" />
    <RowDefinition Height="Auto" />
    <RowDefinition Height="Auto" />
    </Grid.RowDefinitions>
    <Grid.RowDefinitions>
    <StackPanel>
    <TextBlock Margin="12,0,12,0"
    FontFamily=
    "{StaticResource PhoneFontFamilySemiBold}"
    FontSize=
    "{StaticResource PhoneFontSizeLarge}"
    Text="{Binding
    ApplicationName,
    ElementName=Root}" />

    <StackPanel
    Orientation="Horizontal">
    <TextBlock Margin="12,0,12,0"

    FontSize=
    "{StaticResource PhoneFontSizeNormal}"

    Text="Version
    number" />
</Grid>
</phone:PhoneApplicationPage>
```

```
<TextBlock Margin="12,0,12,0"
    FontSize=
    "{StaticResource PhoneFontSizeNormal}"
    Text="{Binding
    Version,
    ElementName
    =Root}" />
</StackPanel>
</StackPanel>
<TextBlock Grid.Row="1"
    Margin="12,24,12,24"
    FontSize="{StaticResource
    PhoneFontSizeNormal}"

    Foreground="{StaticResource
    PhoneSubtleBrush}"

    TextWrapping="Wrap">

    This application show an 'About
    page' sample for Windows Phone applications.
    <LineBreak />
    See more about it, please see the
    following reference:&quot;
</TextBlock>
<HyperlinkButton Grid.Row="2"
    Margin="0,12,0,0"
    HorizontalAlignment="Left"
    Command="{Binding
    SendAnEmailCommand}"
    Content="technical
    support/feedback email"
    FontSize=
    "{StaticResource PhoneFontSizeNormal}"
    Foreground=
    "{StaticResource PhoneAccentBrush}" />
<HyperlinkButton Grid.Row="3"
    Margin="0,12,0,0"
    HorizontalAlignment="Left"
    Command="{Binding
    RateThisAppCommand}"
    Content="rate this
    app"
    FontSize=
    "{StaticResource PhoneFontSizeNormal}"
    Foreground=
    "{StaticResource PhoneAccentBrush}" />
<StackPanel Grid.Row="4"
    Margin="12,0,12,50"
    VerticalAlignment=
    "Bottom">
    <Image HorizontalAlignment="Left"
    Source="Images/
    CompanyLogo.png"
    Stretch="None" />
    <TextBlock Margin="0,12,0,0"
    FontSize=
    "{StaticResource PhoneFontSizeNormal}"
    Text="Copyright (c)
    2013 Nokia Developer Wiki." />
    <TextBlock FontSize=
    "{StaticResource PhoneFontSizeNormal}"
    Text="Nokia" />
</StackPanel>
</Grid>
</phone:PhoneApplicationPage>
```

Here is the code behind:

```
public partial class About :
    INotifyPropertyChanged
```

```
{
    /// <summary>
    /// The application manifest
    /// </summary>
    private readonly
    ApplicationManifest _applicationManifest;
    /// <summary>
    /// Initializes a new instance of the
    /// <see cref="About"/> class.
    /// </summary>
    public About()
    {
        InitializeComponent();

        // class that helps to get the
        // application name and version (that
        // is defined in manifest)
        var applicationManifestService =
            new ApplicationManifestService();
        _applicationManifest =
            applicationManifestService.
                GetApplicationManifest();
    }
    /// <summary>
    /// The property changed.
    /// </summary>
    public event PropertyChangedEventHandler
        PropertyChanged;

    /// <summary>
    /// Gets the application name.
    /// </summary>
    public string ApplicationName
    {
        get
        {
            if (_applicationManifest != null)
            {
                return
                    _applicationManifest.App.Title;
            }
            return "N/D";
        }
    }
    /// <summary>
    /// Gets the version.
    /// </summary>
    public string Version
    {
        get
        {
            if (_applicationManifest != null)
            {
                return
                    _applicationManifest.App.Version;
            }
            return "N/D";
        }
    }
}
/// <summary>
```

```
/// The on property changed.
/// </summary>
/// <param name="propertyName">
/// The property name.
/// </param>
protected virtual void OnPropertyChanged
    ([CallerMemberName] string propertyName =
        null)
    {
        var handler = PropertyChanged;
        if (handler != null)
        {
            handler(this, new
                PropertyChangedEventArgs(propertyName));
        }
    }
}
```

Nota: O exemplo não implementa o padrão MVVM, mas poderia.

Usa a classe ApplicationManifest do (Cimbalino toolkit - cujo código fonte é este).

Usa a classe SendAnEmailCommand que é a implementação do ICommand, e tem como finalidade enviar emails. Sendo utilizado na opção de suporte / feedback.

Usa a classe RateThisAppCommand que é a implementação do ICommand e serve para avaliar a aplicação.

Truque: Se pretende um exemplo mais avançado consulte o artigo Página avançada Sobre para aplicações de Windows Phone.

Referências

[Online Help](#) ou, [download CHM](#) (ZIP, 14MB) (*)

[Telerik Examples - Windows Phone Application](#) (*)

[Telerik RadControls for Windows Phone](#) (*)

[Join Nokia Premium Developer Program and get RadControls for free.](#) (*)

(*) *Artigos disponíveis apenas em Inglês.*

Em conclusão, podemos concluir que existe várias formas de implementar, numa aplicação da Windows Phone, uma página “Sobre” ou “Acerca”, sendo um processo que se pode minimizar e enriquece a aplicação.

Este artigo foi originalmente escrito para a comunidade [Nokia Developer](#), mais especificamente para a [Wiki: Página “Acerca de” para aplicações de Windows Phone](#)

AUTOR



Escrito Por Sara Silva

é licenciada em Matemática – Especialidade em Computação, pela Universidade de Coimbra e é Microsoft Certified Professional 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](https://twitter.com/saramgsilva)

No Code

Game Salad

No Code

Game Salad

“Numa altura em que as aplicações estão a dominar em força, torna-se imperativo saber programar...” E o leitor, também tem esta opinião?

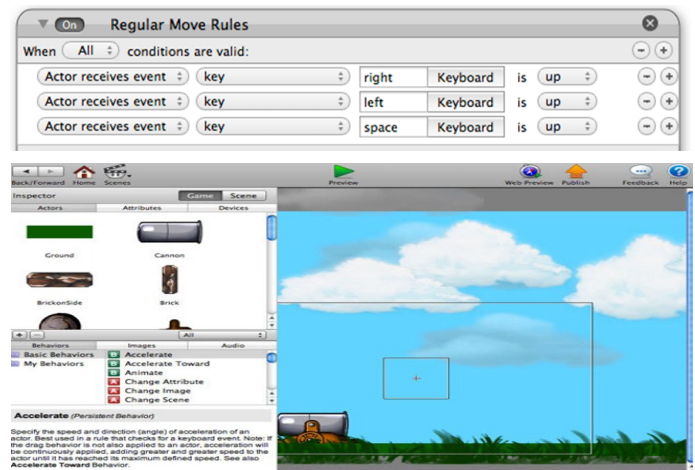
Ora neste artigo, esta é uma ideia que vamos deixar por terra. Se o leitor é alguém que gosta e se interessa pelo mundo dos pequenos jogos e por novas tecnologias mas que não se sente à vontade no mundo da programação, pode (e deve) experimentar o GameSalad. Mas por outro lado, mesmo sendo um programador experiente, o uso desta ferramenta, pode também ajudar bastante.

A GameSalad Inc., é uma empresa norte-americana que desenvolve ferramentas web para a criação de pequenos jogos. Fundada em 2007 com o nome de Gendai Games, em 2009 lançou o GameSalad Creator e em 2010 mudou oficialmente o seu nome passando chamar-se GameSalad.

O importante para os criadores desta tecnologia é a conceção de uma ideia. O Game Salad Creator é principalmente direccionado aos utilizadores que não têm bases em programação, permitindo que qualquer pessoa possa fazer o seu próprio jogo 2D para plataformas móveis IOS, Android ou jogos HTML5 para browsers. Com uma interface limpa e relativamente simples, este software utiliza o sistema “drag-and-drop” para desenvolver os seus jogos, ou seja, desde que tenhamos uma ideia em mente, podemos facilmente pô-la em prática arrastando os componentes da mesma para o “palco de jogo”.

Assim, facilmente o leitor pode definir as imagens de fundo das várias cenas que compõem o jogo e todos os intervenientes dessas cenas são chamados “Atores”, em que o comportamento destes “Atores” é definido por regras de jogo (eventos), que o próprio utilizador define. É tudo uma questão de lógica, e é o utilizador que define essa lógica. Por exemplo, podemos definir eventos específicos como colisões e definir o que o “ator” deve fazer quando ocorre uma colisão, isto é, se deve mudar de rumo e perder pontos ou se deve ser destruído, perdendo uma vida, por exemplo. Tudo se baseia na nossa escolha.

Podemos também adicionar a banda sonora a nosso gosto e, para verificar o comportamento real da nossa aplicação, basta-nos apenas recorrer ao simulador que o *GameSalad*



Creator nos disponibiliza. Não precisamos de ter o equipamento real para o qual estamos a projetar o nosso jogo.

O site principal deste projeto www.gamesalad.com, em pouco tempo atingiu milhares de utilizadores um pouco por todo o mundo. Inicialmente projetado para desenvolver aplicações apenas para dispositivos Apple, hoje em dia é também possível fazer o download da versão *Windows*. A versão base é gratuita e permite a qualquer interessado publicar os seus jogos para a versão *web* e para a versão *MAC*. Caso prefira a versão paga, o *GameSalad Creator Pro* permite tudo isto e também publicar os seus projetos para a plataforma *Android* e para o *Windows 8*, além de outros extras. Caso opte por ter uma conta *premium* pode ainda publicar no mercado de aplicações, escolhendo se quer disponibilizar gratuitamente o seu jogo ou não. Caso opte disponibilizar uma versão paga do jogo que desenvolveu, o utilizador tem direito a 70% do ganho obtido pelo seu jogo.

Este projeto tem também através do próprio site, uma grande comunidade de utilizadores que partilham ideias e experiências, ajudando-se mutuamente em vários projetos.

Tem uma ideia para um jogo? Então está na altura de a por em prática! Saber programar deixou de ser um requisito com a ajuda do GameSalad Creator, aproveite e dê largas à imaginação!

AUTOR



Escrito por Rita Peres

Natural de Castelo Branco, licenciou-se em Engenharia Informática pela Universidade da Beira Interior. Membro do P@P desde Janeiro de 2010.

Elege o melhor artigo desta edição

Revista PROGRAMAR

http://tiny.cc/ProgramarED41_V

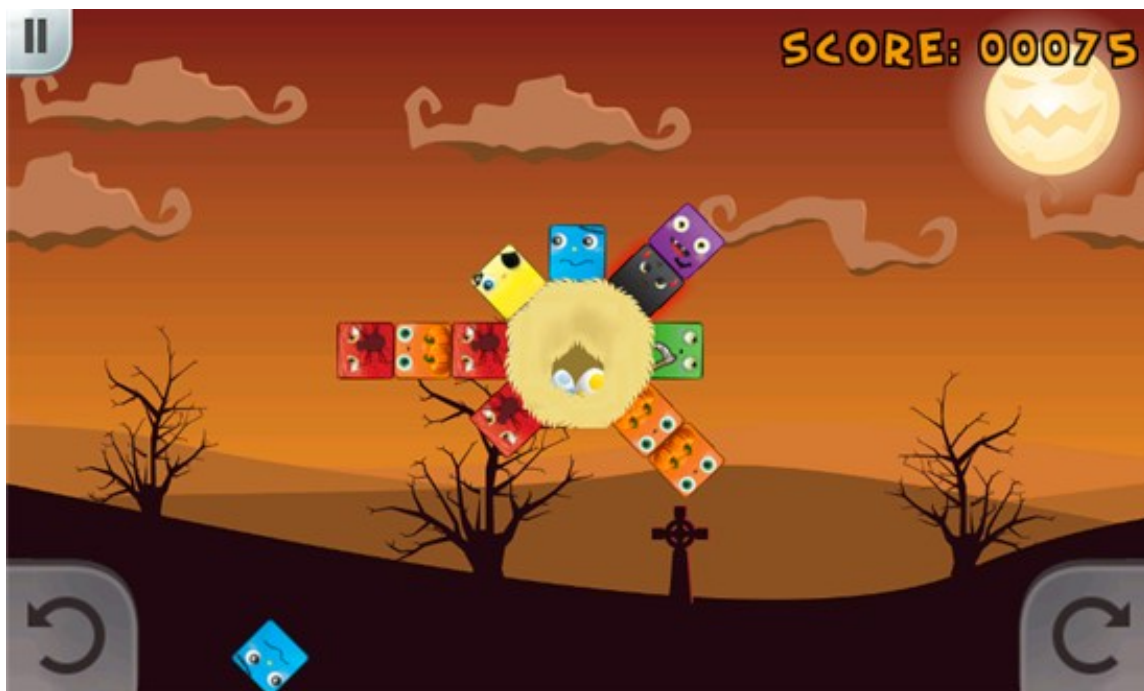
PROJECTO EM DESTAQUE NA COMUNIDADE P@P: LITTLE BITS

Esta foi a aplicação vencedora do concurso de aplicações que se realizou no 1º Evento Presencial da Comunidade Portugal-a-Programar. Trata-se de um jogo para Windows Phone 7.x e 8, bastante interessante e até “viciante” para quem gosta de jogos do tipo puzzle.

Os pequenos Bits estão em perigo e só você pode ajudá-los! Bits são uma espécie muito especial de aves, encontradas em uma ilha remota ... E eles precisam chegar ao seu ninho, a fim de criar seus filhos! Mas, os bits das Trevas estão com

ciúmes e vão tentar tudo ao seu alcance para impedir os Bits de chegar ao ninho.

Bits é um jogo de quebra-cabeças fantástico onde você tem de combinar habilidade com sorte para ajudar os Bits a sobreviver! Combine três Bits da mesma cor para ganhar pontos, enquanto os Bits prestos “evil” tentam detê-los! Os “Anjos Bits” (brancos) podem salvar os Bits de cor, destruindo os Bits Pretos próximos.



PASSATEMPO: DESENVOLVIMENTO WINDOWS 8



Windows® 8

A comunidade Portugal a Programar com o apoio da Microsoft está a criar um passatempo de desenvolvimento de aplicações para Windows 8 exclusivamente para membros registados na comunidade.

Desenvolve uma ou mais aplicações para o Windows 8 e ganha fantásticos prémios!

- **1 MOCHILA APP ME UP PARA TODOS** os que publiquem uma App na Windows 8 Store
- **2 TELEFONES NOKIA LUMIA 820 (Dev Edition)** para os primeiros 2 a publicarem 3 Apps na Windows 8 Store
- **1 TABLET ASUS ME400C** para a melhor App publicada e **1 NOKIA LUMIA 820** para a segunda melhor aplicação.

São válidas apenas aplicações publicadas entre 1 e 30 de Junho de 2013.

Todos os estudantes têm acesso gratuito à Windows Store através do programa [DreamSpark](#) e por isso não há motivos para não concorrer. Mais informações e candidaturas através do email: passatemp@portugal-a-programar.pt

Alguns recursos interessantes para iniciar:

- [Windows 8 Store Apps - Do sonho à realidade \(revista PROGRAMAR\)](#)
- [Microsoft DreamSpark Windows 8 Apps development](#)
- [Windows App \(MSDN\)](#)

O staff da comunidade será o júri que irá escolher a melhor aplicação a concurso e o vencedor será divulgado na primeira quinzena de Julho. Para atribuição do Tablet Asus ME400C é necessário existirem no mínimo 3 aplicações a concurso. Qualquer caso omissivo será resolvido pelo staff da comunidade não sendo passível de recurso e o júri irá rever e validar todas as aplicações de modo a que tenham um mínimo de qualidade e originalidade.

Veja também as edições anteriores da Revista PROGRAMAR

39ª Edição - Fevereiro 2013



38ª Edição - Dezembro 2012



37ª Edição - Outubro 2012



36ª Edição - Agosto 2012



35ª Edição - Junho 2012



34ª Edição - Abril 2012



e muito mais em ...
www.revista-programar.info

DUVIDAS?

IDEIAS?

AJUDAS?

PROJECTOS?



portugal-a-programar
•org

