PROGRAMAR

A REVISTA PORTUGUESA DE PROGRAMAÇÃO

Revista nº7 - Março de 2007 www.portugal-a-programar.org

Introdução à
Programação em WML

Desenvolva aplicações web para dispositivos móveis.

Iniciação ao C

Aprenda a programar nesta linguagem.

WAP MESSAGE X

ELECTRÓNICA

Redes CAN

Programação e conceitos das topologias das redes de dados

Linguagens POA:
ASPECTJ

Domine esta linguagem de programação orientada a aspectos.

índice

equipa PROGRAMAR

contacto
revistaprogramar
@portugal-a-programar.org

website
www.revista-programar.info

Criar, Agir e Apresentar

Nos vários eventos por onde tenho passado, um tema tem sido recorrente: a inovação em Portugal ou, como alguns indicaram, a falta dela. O exemplo mais apresentado era no desenvolvimento de aplicações web, uma área que actualmente está em franco desenvolvimento. Todos os dias surgem novas e originais aplicações, mas ainda é raro encontrar uma equipa portuguesa envolvida. Foram apresentadas várias explicações para esta greve de criação: a localização (cada vez menos importante), a mentalidade, a falta de investimento e, principalmente, a falta de coragem. Cada vez mais se pensa duas vezes antes de se atirar de cabeça nalgum projecto arriscado. No entanto, o melhor caminho apresentado para o sucesso foi mesmo "tentar" (muitas vezes), dedicar-se a um projecto e fazer tudo para que vingue. Fica aqui o incentivo a todos os portugueses que estiverem a ler, se mesmo com toda a dedicação não obtiverem resultados (o que poderá acontecer muitas vezes), aprendam com a experiência e avancem para novas ideias. Mas tentem...

A partir desta edição iremos tentar acompanhar mais eventos que aconteçam por Portugal, no sentido de divulgar o que se vai fazendo pelo nosso país e promover a discussão nos mesmos. Por isso, pedimos que nos notifiquem de futuros eventos, para podermos divulgar na revista, dentro do período possível. Também poderão enviar-nos resumos e fotos de eventos já decorridos para surgirem na nova secção Eventos, inaugurada com o evento Tecnonov 2007, no qual estive presente.

Sérgio Santos

UE ameaça Microsoft

A empresa de Bill Gates tem de cobrar menos pela cedência de códigos-fontes a outras companhias de software. O aviso foi feito no dia 1 de Março pela Comissão Europeia. A CE deu quatro semanas à Microsoft para reduzir os custos de acesso a código-fonte por produtoras de software que querem assegurar a compatibilidade e interacção com aplicações da Microsoft.

De acordo com o executivo comunitário, os preços praticados pela Microsoft apenas estão a alcance das maiores companhias. A Microsoft respondeu à advertência da CE, alegando que os preços de acesso aos códigos são 30% mais baixos que a média praticada em casos similares. A Comissária para a Concorrência Europeia, acusa a Microsoft de violar o artigo 82 do Tratado Europeu da Concorrência, com práticas de abuso de posição dominante. Se não satisfizer as exigências da CE, a Microsoft arrisca-se a gastar numa multa o correspondente às receitas de cinco meses no mundo inteiro…

Adobe lança versão do Photoshop online

Segundo informações de Bruce Chizen, presidente da empresa, nos próximos seis meses será lançada uma versão online do Photoshop, o mais conhecido e utilizado programa de edição de imagem do mundo.

O Photoshop juntar-se-á portanto, ao Adobe Remix, o já existente software online para edição de vídeo. Com estas medidas, a Adobe estará a competir directamente com empresas como o Google, que já oferecem serviços semelhantes.

Seminários IST-Tagus 2007

Os Seminários IST-Tagus, iniciados no ano de 2004, são um evento realizado anualmente visando promover a discussão de experiências e troca de ideias relacionadas com a área tecnológica, bem como a sua influência no mundo empresarial. Sendo organizado por alunos do Instituto Superior Técnico (pólo do Taguspark), este evento tem tido uma crescente participação de profissionais, sendo já um conceituado palco para a apresentação de projectos inovadores e de forte impacto no mercado.

A 4ª edição dos Seminários IST-Tagus realizar-se-á nos dias 19 e 20 de Março, no Centro de Congressos do Taguspark, sendo subordinada ao tema "Real Time Business". Outro dos grandes atractivos da edição deste ano será a exibição de Trabalhos de Final de Curso por parte de alunos do IST.

O evento irá contar com um leque significativo de oradores de prestígio, entre os quais se destacam José Magalhães (Secretário de Estado da Administração Interna), Carlos Batista (T-Systems), Jorge Lopes (BRISA), António Marcelo (TECMIC Easytran Vodafone), entre outros.

Para mais informações, consulte o site oficial: http://seminarios.tagus.ist.utl.pt

Introdução ao WML (WAP)

A tecnologia WAP foi criada pelo Fórum WAP, fundado em 1997 por marcas como a Nokia ou a Motorola, com o objectivo de criar um protocolo de troca de informação em dispositivos móveis (wireless) como os telemóveis. O WAP consiste numa linguagem de troca de informação própria denominada WML. O WML é compatível com os standards de TCP/IP, HTML e XML e inclui também uma linguagem de scripting semelhante ao Javascript, mas mais leve de forma a ser de rápida execução por parte dos dispositivos móveis, já que estes não possuem grande poder de processamento, denominada WMLScript.

A sigla WAP significa Wireless Application Protocol, ou seja, Protocolo para Aplicações Wireless e é definida como uma aplicação XML 1.0.

Os browsers que conseguem interpretar esta linguagem são específicos, ou seja, não pode usar o seu browser habitual (a não ser que possua algum plugin para o efeito) para ver páginas WAP. Os tipos de browser compatíveis com o WAP são denominados Micro Browsers, pelas suas características óbvias – não podem ter grandes necessidades ao nível do hardware, mais especificamente ao nível da memória e CPU, e por isso são ideais para ser usados nos dispositivos móveis.

O WML, como foi explicado anteriormente, é uma linguagem que a bastantes programadores parecerá familiar por ser um misto de HTML e XML, mas mais rígida em termos de interpretação.

As páginas em WML não são de muito diferente programação das páginas em HTML, por isso o programador com experiência em HTML pode facilmente iniciar-se nesta linguagem. Estas páginas têm a extensão .wml e são sensíveis a maiúsculas por terem por base o XML, ou seja, se escrever <abc> não vai ter o mesmo resultado do que se escrevesse <ABC>. Por standard do XML, as tags devem sempre ser apresentadas em minúsculas e devem ser sempre fechadas.

No WAP existem termos específicos para as páginas e o seu conjunto, e costuma fazer-se a analogia entre o WAP e um jogo de cartas. Imagine que tem uma torre de cartas, denominado por Deck, e cada elemento dessa torre é, obviamente, uma carta – denominada por Card no código. Ora assim sendo, uma página WML é considerado um deck e dentro da página WML podem existir múltiplas cards. Quando um utilizador acede à página WML são descarregadas todas as cards do deck, sendo que a navegação entre elas estará a cargo do dispositivo que as requereu, que nessa altura não estará em contacto com o servidor.

O leitor neste momento pode estar um pouco confuso em relação alguns tópicos menos explorados e por isso é preferível que observe um exemplo de um deck, para melhor consolidar os conhecimentos até agora adquiridos:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>

  <card id="carta1" title="Título da Carta 1">
    Isto é uma card. <br />
    Várias tags HTML funcionam aqui e serão abordadas mais à frente nesta
introdução ao WML.
  </card>

  <card id="carta2" title="Segunda Carta">
    <p>
      Segunda carta do deck
    </p>
  </card>

</wml>
```

Resultado:

Explicação do Código:

<wml> - Tag que delimita o deck.
<card> Tag que delimita cada card. Podem ter vários atributos, os apresentados são:
id – Identificação da carta, deve ser um nome único no deck.
title – Título da card, nos terminais WAP costuma ser apresentado no topo do ecrã.
<p> - Parágrafo, idêntico ao HTML.

 - Quebra de linha, semelhante ao HTML.

Várias tags HTML podem ser utilizadas da mesma forma em WML. Aqui fica uma pequena lista das mesmas:

<p> - Parágrafo

 - Quebra de linha
<big> - Texto Grande
<small> - Texto Pequeno
<b> - Bold
<i> - Itálico
<u> - Sublinhado (Underline)

Um elemento muito utilizado em HTML e que também pode ser utilizado no WML é a tabela. As tabelas funcionam praticamente do mesmo modo do que no HTML:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>
  <card title="Tabela">

    <p>
      <table columns="2" border="1">
        <tr>
          <td>Coluna 1</td>
          <td>Coluna 2</td>
        </tr>
      </table>
    </p>

  </card>
</wml>
```

Resultado:

Outras tags como o <a>, que tanto são utilizadas para anchors como para links em HTML podem ser utilizados em WML, mas com uma pequena alteração na forma como os anchors são utilizados:

```
<anchor>Clique Aqui
  <go href="teste.wml"/>
</anchor>
```

As imagens em WML já são uma história diferente. Por serem dispositivos com poucas capacidades em termos de hardware têm de ser utilizadas pequenas, normalmente monocromáticas. Estas imagens têm uma extensão específica, o .wbmp.
Não será abordada aqui a construção de imagens .wbmp, mas uma breve pesquisa na Internet deverá esclarece-lo relativamente a este assunto.

A imagem deverá ser inserida do seguinte modo:

```
<img src="imagem.wbmp" alt="imagem_alt" />
```

Os formulários são elementos muito úteis para adicionar interactividade ao seu site ou portal, apesar dos utilizadores não ficarem muito tempo a navegar na WAP ao contrário da Internet.
Nesta secção vamos deixar o leitor explorar por si próprio código de formulários já concebidos, bem como o seu resultado, de modo a poder aprender melhor a trabalhar com formulários em WML. *(exemplos retirados do site w3schools.com e traduzidos para português)*

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD
WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>
<card title="Input">

<p>
Nome: <input name="Nome" size="15"/><br/>
Idade:  <input name="Idade" size="15"
format="*N"/><br/>
Sexo:  <input name="Sexo" size="15"/>
</p>

</card>
</wml>
```

(imagem encurtada devido ao tamanho do ecrã do simulador)


```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD
WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>
<card title="Lista de seleccao">

<p>
<select>
<option value="htm">HTML Tutorial</option>
<option value="xml">XML Tutorial</option>
<option value="wap">WAP Tutorial</option>
</select>
</p>

</card>
</wml>
```

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD
WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>
<card title="Lista de seleccao 2">

<p>
<select multiple="true">
<option value="htm">HTML
Tutorial</option>
<option value="xml">XML Tutorial</option>
<option value="wap">WAP Tutorial</option>
</select>
</p>

</card>
</wml>
```

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD
WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>
<card title="Fieldset">

<p>
<fieldset title="CD Info">
Titulo: <input name="titulo"
type="text"/><br/>
Premio: <input name="premio" type="text"/>
</fieldset>
</p>

</card>
</wml>
```

Nota: Este simulador não reproduz o fieldset da melhor forma.

Em WML, para a navegação entre as diversas páginas e afins, temos os seguintes elementos:

Go:

```
<anchor>
  Ir para Carta
  <go href="carta.wml"/>
</anchor>
```

Previous:

```
<anchor>
  Anterior
  <prev/>
</anchor>
```

Refresh:

```
<anchor>
 Actualizar
 <go href="pagina.wml"/>
 <refresh>
    <setvar name="a" value="10"/>
  </refresh>
</anchor>
```

Comandos WML

Go
Dirige para a página indicada no href.

```
<anchor>
  Ir para abc
  <go href="abc.wml"/>
</anchor>
```

Previous
Dirige para a página visitada anteriormente.

```
<anchor>
  Página Anterior
  <prev/>
</anchor>
```

Refresh
Actualiza a página ao clicar, e também actualiza as variáveis dadas.

```
<anchor>
  Actualizar a página
  <go href="paginaactual.wml"/>
  <refresh>
    <setvar name="a" value="20"/>
  </refresh>
</anchor>
```

No Operation (noop)
Previne uma acção de ser realizada.

```
<anchor>
  Ir para abc
  <go href="abc.wml"/>
  <noop/>
</anchor>
```

Timers

Utilizam-se normalmente para redireccionar para uma página específica, depois de x segundos.

```
<card ontimer="test.wml">
<timer value="30"/>
<p>Vai ser redireccionado em 3
segundos.</p>
```

Variáveis em WML

Definir uma variável:

```
<setvar name="a" value="500"/>
```

Com isto é criada uma variável a com o valor 500.

Para definir uma variável a partir de um input fazemos:

```
<card id="carta1">
<select name="escolha">
<option value="pap">Portugal-a-
Programar</option>
<option value="revista">Revista
PROGRAMAR</option>
</select>
</card>
```

E seguidamente usamos o seguinte código para usar a variável obtida:

```
<card id="carta1">
  Seleccionaste: $(escolha)
</card>
```

Se seguiu atentamente este artigo com certeza irá gostar de aprofundar mais os seus conhecimentos e aplica-los nos seus próprios sites, por isso deixo aqui uma excelente forma de aprender mais sobre este tema.

Consulte o tutorial da W3Schools sobre este assunto, mais especificamente a página de exemplos (*http://w3schools.com/wap/wml_examples.asp*).

Daqui é tudo, bons sites!

Daniel Correia

Programação Orientada aos Aspectos com AspectJ

Introdução

Durante os últimos anos assistimos a uma grande divulgação da programação orientada aos objectos (POO). Esta veio responder a requisitos essenciais no desenvolvimento de software, nomeadamente no que diz respeito às facilidades de manutenção e reutilização do código. Mas apesar de todas estas evoluções, nem tudo aquilo que a POO se propunha conseguir foi alcançado. É neste contexto que surge a programação orientada aos aspectos (POA). Este novo paradigma de programação surgiu em 1996 por Greger Kiczalos e pela sua equipa no Xerox PARC, que também foram responsáveis pelo desenvolvimento do AspectJ, a linguagem POA mais usada.

Quando vamos desenvolver uma aplicação, é conveniente dividir o problema que nos foi proposto em partes. A separação do problema de acordo com os dados e as funcionalidades a eles associadas facilita o estudo dos requisitos da aplicação. Isto é o que se designa por separação de interesses. O paradigma OO estabelece uma separação de interesses tendo em conta os tipos de dados, representados pelos objectos, e as funções que utilizam cada tipo de dados, ou seja, os métodos a que um objecto responde.

Mas, como já foi referido anteriormente, apesar da orientação aos objectos ter trazido melhorias na forma de programar, não conseguiu resolver alguns problemas. Exemplos clássicos são o registos das operações em ficheiros de log que encontramos em muitos softwares, ou programação concorrente/distribuída. O código associado a este tipo de funcionalidades encontra-se normalmente espalhado por todos os módulos, o que dificulta a manutenção e a evolução do código. Em POA dizemos que isto são interesses entrecortantes, pois cortam transversalmente todos os módulos. A POA vem resolver este problema introduzindo um novo nível de separação de interesses, os aspectos.

Basicamente estes interesses transversais à aplicação são encapsulados nestas novas unidades modulares e posteriormente fundidas com as classes num único sistema. Isto não só aumenta a facilidade de manutenção destas funcionalidades da aplicação, visto que tudo se encontra num único local, como aumenta as possibilidades de reutilização das classes noutros contextos, pois estas encontram-se limpas destas funcionalidades que normalmente variam muito de caso para caso.

O AspectJ

O AspectJ (http://www.eclipse.org/aspectj) foi a primeira linguagem OA desenvolvida e é, provavelmente, a mais usada. Esta linguagem permite acrescentar aspectos a programas feitos em Java. Para compreender o AspectJ é necessário conhecer três conceitos fundamentais presentes na POA:

• advices - são fragmentos de código com as acções referentes aos interesses entrecortantes;

• join points - são locais de um programa onde os advices serão executados, que podem ser a chamada/execução de um método, o acesso a membros de uma classe, a criação de um objecto, o lançamento de excepções, etc.;

• pointcuts - são um conjunto de join points, definidos segundo um determinado critério, que identificam os locais onde um determinado advice será executado.

Pointcuts

Genericamente, a definição de um pointcut obedece à seguinte sintaxe:

```
pointcut <nome>() : <tipo>(<padrão>);
```

Onde:
• <nome> é o identificador deste pointcut
• <tipo> indica a que tipo de join point nos referimos (chamada de uma função, acesso a um atributo, etc.);
• <padrão> é uma expressão que restringe os join points a serem considerados àqueles que fazem matching com esta expressão.

De seguida indicam-se os tipos de join points mais importantes:

• call - chamada de uma função;
• execution - execução de uma função;
• get - consulta dos atributos de um objecto;
• set - alteração dos atributos de um objecto;
• within - join points que ocorrem dentro de determinadas classes;
• target - envio de mensagens a um objecto de um determinado tipo;
• args - envio de mensagens em que os argumentos são de um determinado tipo.

A expressão pode ser uma palavra (nome de uma classe, nome de um método, etc.), mas também temos caracteres especiais:

• * - representa uma sequência de caracteres qualquer, desde que não contenha pontos;
• .. - representa uma sequência de caracteres qualquer, mesmo contendo pontos;
• + - representa as subclasses de uma dada classe.

Vejamos agora alguns exemplos:

• *pointcut tostr() : execution(String toString());*

Representa todas as execuções do método toString. Podemos restringir apenas à execução do método a objectos da classe Xpto substituindo String toString() por String Xpto.toString(), ou a objectos da classe Xpto e todas as suas subclasses fazendo String Xpto+.toString();

• *pointcut sets() : call(* Xpto*.set*(..));*

Representa a chamada de métodos começados pela palavra set de classes começadas pela palavra Xpto, com um número qualquer de argumentos. Podemos limitar apenas a métodos que recebam argumentos de um determinada tipo, por exemplo, fazendo * Xpto*.set*(int,String) limitaríamos apenas aos métodos cujo o primeiro argumento fosse um inteiro e o segundo uma string (para além de verificar todas as outras condições já indicadas).

• *pointcut gets() : get(private String abc*);*

Representa todos os acessos (para consulta) a atributos private do tipo String, começados por abc.

Para definirmos pointcuts também temos à nossa disposição operadores lógicos (!, || e &&, cujo significado é o habitual).

```
pointcut gets() : get(* *) && within(Xpto);
```

Este pointcut representa um acesso a qualquer atributo, desde que seja da classe Xpto.
Agora vamos ver como utilizar o target e o args para restringir o tipo dos objectos a que o método é enviado, assim como o tipo dos argumentos com os quais o método foi invocado.

```
pointcut p1(Xpto x,int y,float z) : execution(* *(..)) && target(x) && args(i,j);
```

Desta forma restringimos os join points aos métodos enviados a um objecto do tipo Xpto e que recebem 2 argumentos (o primeiro do tipo int e o segundo do tipo float). É claro que isto poderia ter sido conseguido com a expressão * Xpto.*(int,float), mas, como veremos de seguida, esta opção permitirá usar o objecto ao qual foi enviado o método, assim como os argumentos recebidos na especificação de advices.

Advices

Se os pointcuts indicavam o conjunto de pontos de uma aplicação onde queríamos realizar outras operações, os advices permitem-nos definir quais são essas operações.
Temos 3 (ou 5) tipos de advices. Usamos o:

- before, quando queremos executar acções antes de um join point;
- after, quando queremos executar acções depois de um join point, existindo três alternativas:
  - o after(), que é sempre executado;
  - o after() returning, que apenas é executado quando o método associado ao join point retorna;
  - o after() throwing, que apenas é executado quando o método associado ao join point lança uma excepção.
- around, quando queremos executar acções em vez de um método (sendo que dentro do advice podemos executar o método original).

Vamos agora analisar alguns exemplos concretos. Para tal vamos considerar as seguintes classes Java:

```
public class Classe1 {
  private int x;
  private boolean a;

  public Classe1(int x,boolean a) {
    this.x = x;
    this.a = a;
  }

  public void setX(int x) {this.x=x;}

  public void setA(boolean a) {this.a=a;}

  public String toString() {
    return ("x=" + this.x + " / a=" + this.a);
  }
}
```

```
public class Classe2 {
  private int y;

  public Classe2(int y) {
    this.y=y;
  }

  public void setY(int y)
{this.y=y;}

  public String toString() {
    return ("y=" + this.y);
  }
}
```

O primeiro problema que vamos resolver consiste em alterar os métodos toString(). Vamos alterá-los de forma a que passem a criar uma string que contenha também o nome da classe. Tal pode ser conseguido do seguinte modo:

```
pointcut tostr(Object obj) : execution(String toString())
                          && target(obj);

String around(Object o) : tostr(o) {
  String s = proceed(o);
  return ("<" + o.getClass().getName() + " @ " + s + ">");
}
```

Primeiro definimos o pointcut que, para além de restringir os métodos à execução do toString(), também capta o objecto ao qual o método é enviado através do target. Desta forma foi possível utilizar o objecto no advice, como se pode ver nas linhas 5 e 6. Foi também usado proceed, que executa o método original. Depois é só acrescentar o nome da classe e devolver o resultado.

No próximo exemplo vamos ver como usar o args, que é semelhante ao target, mas este capta os argumentos do método (ou construtor, como será o caso). Este aspecto irá imprimir uma mensagem sempre que um objecto de uma classe começada por Classe for inicializado através de um construtor que receba um argumento do tipo inteiro.

```
pointcut init(int x) : initialization(new(..)) && within(Classe*) && args(x);

after(int x) : init(x) {
  System.out.println("Classe: " +
    thisJoinPointStaticPart.getSignature().getDeclaringType().getName());
  System.out.println("Argumentos: " + x);
}
```

Neste exemplo foi também usada a variável thisJoinPointStaticPart, que contém uma série de informações sobre o join point em que o aspecto vai ser executado. Existe também a variável thisJoinPoint, que poderia ser usada neste caso para saber quais os argumentos recebidos. Tal será feito no próximo exemplo.

Suponhamos agora que queríamos registar todas as alterações realizadas em qualquer uma das classes. Para alterar o valor dos atributos é necessário recorrer aos métodos set*. Logo vamos criar um pointcut que agrupe todas as execuções destes métodos. Também vamos querer registar qual o objecto afectado, quais os argumentos usados no método (sendo que o seu tipo nem sempre é o mesmo) e qual o nome do método.

```
pointcut sets(Object obj) : call(* Classe*.set*(..)) && target(obj);

before(Object obj) : sets(obj) {
  try {
    BufferedWriter bw = new BufferedWriter(new FileWriter("history.txt",true));
```

```
    Date d=(Calendar.getInstance()).getTime();
    bw.write("Data: " + d + "\nObjecto: " + obj + "\nMetodo: "
            + thisJoinPointStaticPart.getSignature().getName()
            + "\nArgumentos:\n");
    Object[] args = thisJoinPoint.getArgs();
    String[] ids = ((CodeSignature)thisJoinPoint.getSignature())
                    .getParameterNames();
    for (int i = 0; i < args.length; i++) {
      bw.write("  " + ids[i] + "=" + args[i] + "\n");
    }
    bw.close();
  } catch(Exception e){System.out.println(e);}
}
```

Aqui já recorremos à variável thisJoinPoint para obter os argumentos usados no método, assim como os identificadores dos argumentos. O resultado obtido será algo como:

```
Data: Sat Dec 09 17:56:30 WET 2006
Objecto: <Classe1 @ x=12 / a=true>
Metodo: setX
Argumentos:
x=10
```

Conclusão

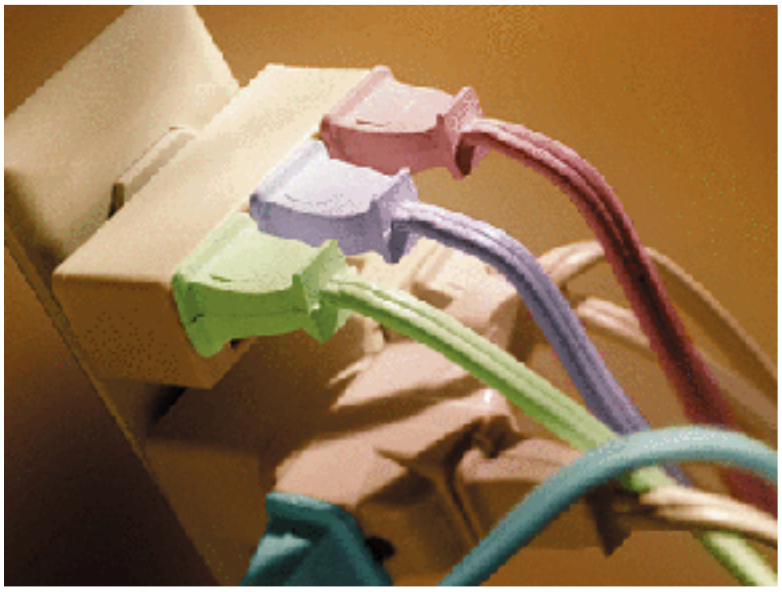
Ao longo deste artigo ficaram visíveis alguns dos benefícios que podemos obter com a POA. Contudo há alguns factos que é necessário realçar. Em primeiro, que uma abordagem dos problemas segundo esta metodologia, é bastante mais complicada do que a abordagem OO. É difícil definir a estrutura do software, identificando correctamente todas as partes que o compõem. Mesmo depois de definida a estrutura, cometemos facilmente erros na definição dos aspectos (acontece várias vezes definir aspectos que originam ciclos infinitos, pois ao executá-los criávamos situações onde outros aspectos poderiam ser executados), que por vezes não são muito visíveis. Sublinha-se ainda que não é objectivo da POA substituir a POO, tratam-se sim de dois paradigmas complementares.

Por último referia-se que este texto é apenas uma introdução à POA/AspectJ e, como tal, muitas das capacidades deste novo paradigma não foram aqui demonstradas. Sugere-se a quem quiser aprofundar os seus conhecimentos nesta área, a consulta da página web do AspectJ, onde poderá encontrar uma vasta documentação sobre este tema.

Referências

http://www.eclipse.org/aspectj
http://en.wikipedia.org/wiki/Aspect-oriented_programming
http://www.research.ibm.com/hyperspace
http://www.devx.com/Java/Article/28422

Rui Gonçalves



Sockets de Berkely em linguagem C

Na anterior edição da Revista PROGRAMAR foi dado a conhecer como funcionam os sockets em Java. Também foram descritas as principais diferenças entre UDP e TCP. Neste artigo vamos fazer uma breve introdução aos Sockets de Berkeley em linguagem C para ambientes UNIX e Linux.

Basicamente os Sockets de Berkeley são um API, isto é, um conjunto de bibliotecas de funções para a programação sobre protocolos de comunicação.

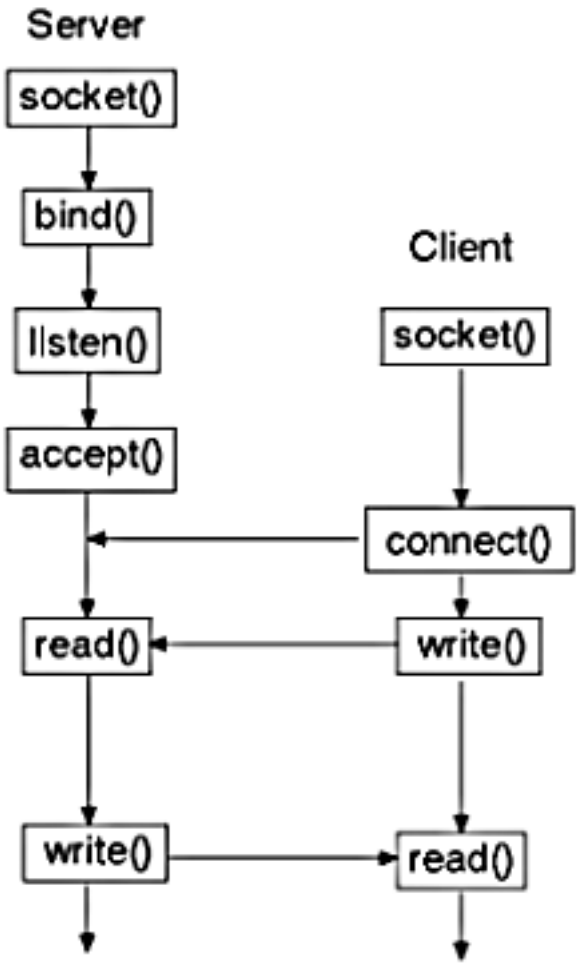
Modelo Cliente – Servidor

O modelo Cliente – Servidor é composto por um conjunto de dois programas em execução que comunicam entre si. O servidor está sempre à espera de pedidos efectuados pelos clientes mas desconhece a sua localização. O cliente tem de conhecer obrigatoriamente a localização do servidor (IP) para poder ligar-se e comunicar com ele.

O código do cliente e do servidor não têm de ser necessariamente diferentes, facilmente se desenvolve uma aplicação que tanto pode funcionar como cliente como para servidor. A grande diferença está no seu comportamento, ou seja, o servidor inicia a execução e aguarda uma ligação, o cliente inicia a execução e a ligação ao servidor.

Para dar um exemplo prático de como funcionam os sockets, vamos fazer um simples par de aplicações em que o cliente envia uma palavra com letras minúsculas para o servidor. O servidor trata de converter as letras em maiúsculas devolvendo a palavra ao cliente. Vamos utilizar o modo TCP do protocolo TCP/IP e o conjunto de aplicações são fechadas quando o cliente digita a palavra “exit”.

Antes de mais nada convém ver como funciona o estabelecimento de uma ligação TCP. A seguinte figura é bastante explícita quanto a isso:



Agora vamos ver passo a passo as funções utilizadas:

Cliente

1 - Conjunto de bibliotecas a adicionar às tradicionais:

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/ioctl.h>
#include <unistd.h>
```

2 - Definir a estrutura de destino (servidor) e as variáveis:

```
// definição da estrutura do servidor
struct sockaddr_in target;
// criar o socket, indicando a família
// a que pertence (AF_INET), o tipo de
// protocolo (neste caso TCP -
// SOCK_STREAM) e o parâmetro do
// protocolo (0).
int sock = socket(AF_INET,SOCK_STREAM,0);
// variáveis
char palavra[50], palavra2[50];
// tamanho da estrutura do servidor
int ad1 = sizeof(target);
```

3 - Inicializar a estrutura:

```
// inicializa a estrutura do servidor
bzero((char *)&target, ad1);
```

4 - Definir as propriedades do servidor ao qual nos vamos ligar:

```
// indica a família do protocolo
target.sin_family = AF_INET;
// especifica o endereço (IP) do
// servidor
target.sin_addr.s_addr =
inet_addr("127.0.0.1");
// porta que o programa vai usar
target.sin_port = htons(8450);
```

5 - Estabelecer a ligação ao servidor:

```
// efectua a ligação ao servidor.
// Se falhar (por exemplo o servidor
// estar em baixo) o programa termina
if(connect(sock, (struct sockaddr
*)&target, ad1) == -1)
{
  close(sock);
  puts("Conexao falhou!");
  exit(0);
}
```

6 - Lê uma palavra do teclado enquanto esta for diferente de “exit”. Em seguida envia-a para o servidor e recebe-a com letras maiúsculas:

```
do {
  scanf("%s", palavra);
  // envia para o servidor os dados
  // contidos na variável “palavra”
  write(sock, palavra, 50);
  if(strcmp(palavra, "exit") != 0){
    // recebe do servidor os dados
    // e guarda-os na variável
    // “palavra2”
    read(sock, palavra2, 50);
    printf("%s\n", palavra2);
  }
}while(strcmp(palavra, "exit") != 0);
```

7 - Fechar o socket:

```
close(sock);
```

Servidor

1 - Conjunto de bibliotecas a adicionar às tradicionais:

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <ctype.h>
```

2 - Definir a estrutura do cliente, do servidor e as variáveis:

```
// definição das estruturas do
// cliente e do servidor
struct sockaddr_in me, from;
// criação dos sockets.o “sock” é
// declarado identicamente ao cliente
int newSock, sock =
socket(AF_INET,SOCK_STREAM,0);
// declaração das variáveis
int tam = 0, i = 0;
int ad1 = sizeof(me);
char palavra[50], palavra2[50];
```

3 - Inicializar a estrutura:

```
// inicializa a estrutura do servidor
bzero((char *)&me, ad1);
```

4 - Definir as propriedades do servidor:

```
// indica a família do protocolo
me.sin_family = AF_INET;
// fica associado a todos os
// endereços IP do host local
me.sin_addr.s_addr = htonl(INADDR_ANY);
// porta em que o servidor vai
// estar à escuta
me.sin_port = htons(8450);
```

5 - Alocar a porta. Se estiver ocupada é fechado o servidor.

```
// se a porta estiver ocupada o
servidor não pode correr e é
terminado
if(bind(sock, (struct sockaddr
*)&me, ad1) == -1)
{
      close(sock);
      puts("Porta Ocupada!");
      exit(0);
}
```

6 - Esperar por pedidos. É criado um novo socket para tratar apenas deste pedido enquanto que o outro socket continua à espera de pedidos:

```
// coloca o socket à escuta.
// Podem ser mantidos em espera 5
// pedidos de ligação
listen(sock, 5);
// gera um novo socket específico
// para essa ligação
newSock = accept(sock, (struct
sockaddr *)&from, (void *)&ad1);
// fechar o socket antigo
close(sock);
```

7 - Ciclo infinito que trata dos pedidos do cliente:

```
for(;;){
  // recebe os dados do cliente e
guarda-os na variável “palavra”
  read(newSock, palavra, 50);
  if(strcmp(palavra, "exit") != 0){
    tam = strlen(palavra);
    for(i=0;i<tam;i++){
      if((palavra[i]>='a') &&
(palavra[i]<='z')){
        palavra2[i] =
toupper(palavra[i]);
      }
    }
    palavra2[i] = '\0';
    // envia os dados que estão na
variável “palavra2” para o cliente
    write(newSock, palavra2, 50);
  }
  else{
    close(newSock);
    exit(0);
  }
}
```

Servidor

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <ctype.h>

int main() {
      struct sockaddr_in me, from;
      int newSock, sock = socket(AF_INET, SOCK_STREAM, 0);
      int tam = 0, i = 0;
      int ad1 = sizeof(me);
      char palavra[50], palavra2[50];

      bzero((char *)&me, ad1);

      me.sin_family = AF_INET;
      me.sin_addr.s_addr = htonl(INADDR_ANY);
      me.sin_port = htons(8450);

      if(bind(sock, (struct sockaddr *)&me, ad1) == -1)
      {
            close(sock);
            puts("Porta Ocupada!");
            exit(0);
      }

      listen(sock, 5);
      newSock = accept(sock, (struct sockaddr *)&from, (void *)&ad1);
      close(sock);

      for(;;){
            read(newSock, palavra, 50);
            if(strcmp(palavra, "exit") != 0){
                  tam = strlen(palavra);
                  for(i=0;i<tam;i++){
                        if((palavra[i]>='a') && (palavra[i]<='z')){
                              palavra2[i] = toupper(palavra[i]);
                        }
                  }
                  palavra2[i] = '\0';
                  write(newSock, palavra2, 50);
            }
            else{
                  close(newSock);
                  exit(0);
            }
      }
      return 0;
}
```

Cliente

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/ioctl.h>
#include <unistd.h>

int main() {
      struct sockaddr_in target;
      int sock = socket(AF_INET, SOCK_STREAM, 0);
      char palavra[50], palavra2[50];
      int ad1 = sizeof(target);

      bzero((char *)&target, ad1);

      target.sin_family = AF_INET;
      target.sin_addr.s_addr = inet_addr("127.0.0.1");
      target.sin_port = htons(8450);

      if(connect(sock, (struct sockaddr *)&target, ad1) == -1)
      {
            close(sock);
            puts("Conexao falhada!");
            exit(0);
      }

      do {
            scanf("%s", palavra);
            write(sock, palavra, 50);
            if(strcmp(palavra, "exit") != 0){
                  read(sock, palavra2, 50);
                  printf("%s\n", palavra2);
            }
      }while(strcmp(palavra, "exit") != 0);
      close(sock);
      return 0;
}
```

Para conhecer melhor como são compostas algumas estruturas o leitor pode visitar os links:
http://jan.netcomp.monash.edu.au/ClientServer/socket/socket.html e
http://www.dei.isep.ipp.pt/~andre/documentos/sockets-berkeley.html

Senão pode também pesquisar na Internet já que existe inúmera informação no que diz respeito à API sockets de Berkeley.

Estatísticas em PHP 1ª Parte

Quem já criou um website certamente teve a curiosidade de saber quantas pessoas por lá passaram e de onde seriam essas pessoas. A curiosidade humana é natural e leva a estas questões. Esta é a primeira parte dum tutorial que será concluído na próxima edição.

Para resolver de uma forma simples e inteiramente ao nosso gosto irá ser desenvolvido aqui um tutorial "passo-a-passo" de forma a que no final deste artigo tenha um sistema de estatísticas simples e funcional com as informações mais utilizadas. Todas estas informações serão mostradas com auxílio visual de gráficos, para que os resultados sejam mais apelativos aos visitantes.

Antes de começar a programar, convém fazer o download de algumas classes em PHP que irá utilizar de forma a minimizar o seu trabalho. São elas:

• GEOIP
É uma classe com versão opensource e versão licenciada. Nós iremos utilizar a versão livre para nos retornar o código e nome do País de origem do visitantes. Poderão realizar o download em:
http://www.maxmind.com/app/geoip_country

• Maani PHP/SWF Charts
É uma ferramenta livre que nos permite criar gráficos em Shockwave Flash em tempo real com dados estáticos ou dinâmicos. Encontram o download e informação mais completa em:
http://www.maani.us/charts/index.php

Devido à extensão do script, neste artigo apenas irei falar das funções mais relevantes. O script na sua totalidade, implementa 4 gráficos (2 de barras, e 2 circulares), e 4 tabelas resumo. No entanto apenas iremos falar aqui da construção de um gráfico e de uma tabela resumo. Na conclusão do artigo (na próxima edição) será referido um link para o script completo.

Instalação

Após ter feito o download de ambas as ferramentas (GeoIP e Maani Charts), deverá descompactá-las para a raiz do seu website.
Os ficheiros descritos nas próximas páginas deverão também ser guardados na raiz do website.

SQL

Como o projecto irá salvaguardar os dados de todos os visitantes que passam no website, deverá guardar os respectivos dados numa base de dados em MySQL.

```
CREATE DATABASE IF NOT EXISTS `stats`;

USE `stats`;

DROP TABLE IF EXISTS `estatisticas`;

CREATE TABLE `estatisticas` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `ip` varchar(15) NOT NULL,
  `referer` varchar(150) DEFAULT NULL,
  `browser` varchar(150) DEFAULT NULL,
  `os` varchar(150) DEFAULT NULL,
  `pagina_visitada` varchar(150) NOT NULL,
  `cod_pais` varchar(4) DEFAULT NULL,
  `nome_pais` varchar(75) DEFAULT NULL,
  `dominio_visitante` varchar(150) DEFAULT NULL,
  `data` date NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM;
```

Recolha de Informações

Agora que já tem a base de dados pronta e preparada para receber dados, passa à fase do PHP criando o ficheiro 'statistics.php'.
Inicia assim o ficheiro logo com a função mais importante de todo o sistema. A função que permite recolher os dados do visitante, e irá inserir na respectiva tabela.

```
<?php

function saveStatistics ( ) {

  include( "connection.php");
  include( "geoip.inc" );

  $geoIP_Pais = geoip_open( "GeoIP.dat" , GEOIP_STANDARD );
```

Utilizando a função 'geoip_open' que faz parte da Classe GeoIP faz com que se carregue com a 'base de dados' que irá servir para identificar os Países.

```
$ip = $_SERVER[ 'REMOTE_ADDR' ];
if ( ! $_SERVER[ 'HTTP_REFERER' ] ) {
  $referer = "URL Directo";
}
else {
  $temp_referer=parse_url(htmlspecialchars(strip_tags($_SERVER['HTTP_REFERER'])));
  if ( $temp_referer[ 'host' ] == $_SERVER[ 'HTTP_HOST' ] ) {
    $referer = "URL Directo";
  }
  else {
    $referer = $temp_referer[ 'host' ];
  }
}
```

O próprio PHP tem quase todas as instruções necessárias para recolher a informação que necessita. A variável '$_SERVER' permite saber o IP do visitante através de 'REMOTE_ADDR'. Com 'HTTP_REFERER' conseguir saber de que página é originário. No entanto o 'HTTP_REFERER' é uma instrução muito traiçoeira, pois por vezes dá informações incorrectas que tem que salvaguardar. Se não existir nenhum 'HTTP_REFERER' significa que o utilizador provavelmente inseriu directamente o URL do website e como tal, ficará catalogado como 'URL Directo'. Caso exista um 'HTTP_REFERER' então não irá precisar de todo o 'REFERER' para o tratamento, irá apenas necessitar de algo do género: www.portugal-a-programar.com. Para tal utiliza uma série de instruções do PHP que irá isolar a informação que pretende de toda a informação que vem a mais no URL.

O último problema a de resolver prende-se com o facto de que se fôr efectuado um 'REFRESH' na própria página o 'HTTP_REFERER' devolve apenas um "array()" vazio. Irá catalogar um 'REFRESH' como um 'hit' de 'URL Directo'.

```
$temp_page_visited = explode ( "/" , $_SERVER[ 'PHP_SELF' ] );
$page_visited = end ( $temp_page_visited );
```

Num sistema de estatísticas é importante que tenha conhecimento de quais as páginas mais visitadas no seu website. Utilizará $_SERVER['PHP_SELF'], contudo ao utilizar esta instrução recebe informação adicional no URL que não necessita. Terá que isolar a parte final do URL de forma a que tenha apenas a página visitada.

```
$country_code = geoip_country_code_by_addr ( $geoIP_Pais , $ip );
$country_name = geoip_country_name_by_addr ( $geoIP_Pais , $ip );
$visitor_hostname = @gethostbyaddr ( $ip );

if(ereg("^(([1]?[0-9]{1,2}|2([0-4][0-9]|5[0-5]))\.){3}([1]?[0-9]{1,2}|2([0-4][0-
9]|5[0-5]))$", $visitor_hostname)) {
  $visitor_hostname = "Indefinido";
```

Para que possa saber qual o código e o nome do País do visitante através do IP volta a utilizar a classe GeoIP com as funções 'geoip_country_code_by_addr()' e 'geoip_country_name_by_addr()' respectivamente.

Com o '@gethostyaddr()' pode saber o domínio do visitante. No entanto pode ter não conseguir saber o nome do domínio por estar a ser utilizado um Servidor Anónimo, ou ainda a ser utilizado um servidor que se esconde por várias interligações em forma de saltos (hops), e acaba sempre por ser retornado um novo IP. Salvaguardando esta hipótese, testa se o que é retornado é um IP válido com a função 'ereg()', caso seja uma condição verdadeira então define-se o domínio do visitante como 'Indefinido'.

```
$browser_info = $_SERVER[ 'HTTP_USER_AGENT' ];

if ( eregi ( "(opera) ([0-9]{1,2}.[0-9]{1,3}){0,1}" , $browser_info ) )
  $browser = "Opera";
elseif ( eregi ( "(opera/)([0-9]{1,2}.[0-9]{1,3}){0,1}" , $browser_info ) )
  $browser = "Opera";
elseif ( eregi ( "(konqueror)/([0-9]{1,2}.[0-9]{1,3})" , $browser_info ) )
  $browser = "Konqueror";
elseif ( eregi ( "(konqueror)/([0-9]{1,2})" , $browser_info ) )
  $browser = "Konqueror";
...
else
     $browser = "Desconhecido";
```

Devemos conhecer que Explorador de Internet (Browser) utiliza o visitante, ao criar um website deverá ser sempre compatível com todos os Exploradores existentes. É óbvio que compatível com todos a 100% é praticamente impossível, portanto é necessário saber sempre quais são os mais importantes e mais utilizados pelo visitante, de forma a desenvolver no seu website a compatibilidade com os outros Exploradores dos visitantes. Com o código acima descrito pode-se então verificar que a instrução $_SERVER['HTTP_USER_AGENT'] devolve uma string, na qual permite saber o Explorador de Internet que está a ser utilizado na visualização do seu site.

Aqui, apenas é colocado alguns Exploradores a título de exemplo, no script completo existem muitos mais.

```
if ( eregi ( "linux" , $browser_info ) )
  $os = "Linux";
elseif ( eregi ( "Win 9x 4.90" , $browser_info ) )
  $os = "Windows ME";
elseif ( eregi ( "win32" , $browser_info ) )
  $os = "Windows";
elseif ( eregi ( "windows 2000" , $browser_info ) )
  $os = "Windows 2000";
...
else
  $os = "Desconhecido";
```

Da mesma forma que soube qual o Explorador de Internet através da 'string' retornada pelo 'HTTP_USER_AGENT', também se consegue saber o Sistema Operativo do utilizador. É um dado bastante importante, principalmente se estiver a desenvolver um website com conteúdo para PC's, Smartphone's ou PDA's. Também aqui o código foi resumido, podendo encontrar mais Sistemas Operativos no Script completo.

```
mysql_select_db ( $database , $connection );

$sqlQuery_Statistics = sprintf("INSERT INTO estatisticas (ip, referer, browser,
os, pagina_visitada, cod_pais, nome_pais, dominio_visitante, data) VALUES ('%s',
'%s', '%s', '%s', '%s', '%s', '%s', '%s', '%s')", $ip, $referer, $browser, $os,
$page_visited, $country_code, $country_name, $visitor_hostname, date("Y-m-d") );
$query_result = mysql_query($sqlQuery_Statistics, $connection) or die
(mysql_error());

}
```

Com todos os dados essenciais do visitante já recolhidos fica a faltar a sua inserção na Base de Dados, terminando assim a função que recolhe dos elementos essenciais ao funcionamento do script.

Classe

Continuando o ficheiro 'statistics.php' chegou a altura de criar então a 'Classe' responsável pela geração dos gráficos e relatórios através dos dados existentes na Base de Dados.

A próxima função da 'Classe Statistics' é de mera personalização do gráfico para o seu aspecto visual, como tal não irá se comentada. Mais informação sobre a personalização dos gráficos deverá ser obtida em: http://www.maani.us/charts/index.php?menu=Gallery .

```
class Statistics {

   function setGraphicColumnStyle ( $chart ) {

      // Formatar o grafico com as propriedades visuais para o eixo do 'X'
      $chart[ 'axis_category' ] = array ( 'font'=>"arial", 'bold'=>true,
                           'size'=>9, 'color'=>"000000",
                           'alpha'=>50, 'skip'=>2 );
      // Formatar o grafico com as propriedades visuais para o eixo do 'Y'
      $chart[ 'axis_value' ] = array ( 'font'=>"arial", 'bold'=>true,
                           'size'=>10, 'color'=>"000000",
                           'alpha'=>50, 'steps'=>4,
                           'prefix'=>"", 'suffix'=>"",
                           'decimals'=>0, 'separator'=>"",
                           'show_min'=>true );
      // Formatar o grafico com as propriedades visuais da legenda
      $chart[ 'legend_label' ] = array( 'layout'=>"horizontal",
                           'font'=>"arial",
                           'bold'=>true,
                           'size'=>12, 'color'=>"333355",
                           'alpha'=>90 );
      // Formatar o grafico com as propriedades visuais do fundo da legenda
      $chart[ 'legend_rect' ] = array ( 'height'=>20, 'margin'=>5,
                           'fill_color'=>"000066",
                           'fill_alpha'=>8,
                           'line_color'=>"000000",
                           'line_alpha'=>0,
                           'line_thickness'=>0 );
      // Formatar o grafico com a cor para as colunas de dados
      $chart[ 'series_color' ] = array ("666666");

      // Formatar o grafico com as propriedades visuais para o fundo do grafico
      $chart[ 'chart_rect' ] = array ( 'positive_color'=>"000066",
                           'negative_color'=>"000000",
                           'positive_alpha'=>10,
                           'negative_alpha'=>30 );
      // Formatar com as propriedades visuais para a grelha do eixo do 'X'

      $chart[ 'chart_grid_h' ] = array ( 'alpha'=>20, 'color'=>"000000",
                           'thickness'=>1, 'type'=>"dashed" );
      // Formatar o grafico com as propriedades visuais dos valores das colunas

      $chart[ 'chart_value' ] = array ( 'color'=>"ffffff", 'alpha'=>85,
                           'font'=>"arial", 'bold'=>true,
                           'size'=>10, 'position'=>"middle",
                           'prefix'=>"", 'suffix'=>"",
                           'decimals'=>0, 'separator'=>"",
                           'as_percentage'=>false );
       return $chart;
   } // FIM DE "setGraphicColumnStyle"
```

Com esta função prepara-se o aspecto visual para o gráfico de colunas, mas existe também uma função para personalização do gráfico circular no Script completo.

Com esta função prepara-se o aspecto visual para o gráfico de colunas, mas existe também uma função para personalização do gráfico circular no Script completo.

```
function getPageViewsGraph ( $beginDate , $endDate ) {
  include ( "connection.php");
  include ( "date_functions.php" );
  mysql_select_db ( $database , $connection );
  $sql_pageViews = "SELECT count(ip) AS hits, data FROM estatisticas WHERE data >= '" .
   changeDate($beginDate) . "' AND data <='" . changeDate($endDate) . "' GROUP BY data";
  $pageViews=mysql_query($sql_pageViews,$connection) or die(mysql_error());
```

Esta função irá gerar o Gráfico para totais de visitas por dia, extraindo os dados da base de dados com contagem de todos os IP guardados e com os totais agrupados por dia.

```
for ($i=0;$i<(($endDate-$beginDate)+1);$i++) {
  $days[$i]=date("d-m",strtotime("-".(($endDate-$beginDate)-$i)."
                            day",strtotime(changeDate($endDate))));
}
```

Como as datas introduzidas para visualização dos gráficos e relatórios é variável, cria-se um 'array' dos dias existentes entre as duas datas com os rótulos no formato de 'dia-mês' (ex: 17-01).

```
while($row_pageViews=mysql_fetch_array($pageViews,MYSQL_ASSOC)) {
  for ( $i=0 ; $i < sizeof ( $days ) ; $i++ ) {
    if($days[$i]==date("d-m",strtotime($row_pageViews['data']))){
      $dailyVisits[$i]=$row_pageViews['hits'];
    } else {
      if ( $dailyVisits[ $i ] > 0) {
        $dailyVisits[ $i ] = $dailyVisits[ $i ];
      } else {
         $dailyVisits[ $i ] = 0;
      }
    }
  }
}
```

Com um 'array' de intervalo dos dias escolhidos chegou a altura de preencher um outro 'array' com o mesmo número de posições que contenha os totais de visita naquele respectivo dia.

```
$chart = $this->setGraphicColumnStyle ( $chart );
```

Para se atribuir um aspecto visual personalizado ao gráfico utiliza-se a função criada anteriormente para os gráficos de colunas.

```
  $chart [ 'chart_data' ][ 0 ][ 0 ] = "";
  $chart [ 'chart_data' ][ 1 ][ 0 ] = "Total de Visitas por dia";
  for ( $i=0 ; $i < sizeof ( $days ) ; $i++ ) {
    $chart['chart_data'][0][($i+1)] = $days[$i];
    $chart['chart_data'][1][($i+1)] = $dailyVisits[$i];
  }
  return $chart;
} // FIM DE "getPageViewsGraph"
```

A forma de preencher a informação para gerar o gráfico em Flash é como num programa de Folha de Cálculo. Cria-se o rótulo do 'X' e do 'Y' e no cruzamento dos rótulos preenche-se com os dados respectivos. Após o preenchimento da variável '$chart' com todos os dados necessários à geração, retorna-se essa varável que irá ser utilizada mais tarde pela Classe Charts. O restante artigo será apresentado na próxima edição.

Bruno Vaz

Pharming

Introdução

Nunca o computador foi utilizado para armazenar tanta informação e tantos dados pessoais como agora. E não é necessário ser-se muito conhecedor para conseguir retirar essas informações de um computador se este não estiver bem protegido. Hoje em dia, com a crescente utilização da Internet para quase tudo, é necessário ter o conhecimento sobre o que lhe pode acontecer se for alvo de um ataque por parte de alguém mal intencionado. Existem inúmeros esquemas on-line nos quais os menos prevenidos podem ser apanhados. Neste artigo vamos falar um pouco de Pharming, uma técnica que deriva do Phishing e que é complementar desta segunda…

O que é?

O Pharming é uma variante mais sofisticada de Phishing que explora vulnerabilidades dos browsers, dos sistemas operativos e dos servidores de DNS (Domain Name System) para conseguir conduzir os utilizadores a sites fictícios com o objectivo de obter os códigos de acesso.

Todos os sites da Internet podem ser acedidos através de um identificador único, conhecido por “endereço IP”, que os permite localizar.

O endereço IP é constituído por quatro números, de 0 a 255, separados por pontos, como por exemplo “127.0.0.1”. Como os endereços IP são mais difíceis de memorizar para o cérebro humano que tem mais “capacidades” quando se trata de memorizar nomes, existe o “nome de domínio”, que é mais fácil de memorizar (por exemplo, o www.portugal-a-programar.org é muito mais fácil de decorar que “195.22.25.172”). Sempre que introduzimos o nome de domínio num browser, é traduzido para um IP permitindo deste modo aceder ao site pretendido. Se existir algum erro na tradução do “nome de domínio” para “endereço de IP” e esse erro tiver origem em alguém com intenções maliciosas podemos considerar que existe um ataque de Pharming baseado nas alterações de DNS's.

O ataque

Ataque nos servidores DNS

Alguns softwares usados em servidores DNS possuem falhas de segurança, programação ou má configuração, que permitem "envenenar" a memória temporária (cache) do sistema atacado. Assim o intruso consegue aceder ao servidor e alterar certas configurações atribuindo “nomes de domínio” a IP’s que não lhes deveriam corresponder e que são controlados pelo intruso. Assim, num ataque específico, o endereço IP associado ao domínio portugal-a-programar.org, por exemplo, poderia ser mudado de 195.22.25.172 para 209.85.129.99 num servidor DNS atacado e consequentemente quando se introduzisse no browser www.portugal-a-programar.org este seria redireccionado para a página portuguesa do Google.

Neste ataque, um servidor de nomes (servidor DNS) é comprometido, de tal forma que as requisições de acesso a um site feitas pelos utilizadores deste servidor sejam redireccionadas para outro endereço, sob controlo dos intrusos.

Este tipo de ataques também pode ser feito remotamente ou por meio de programas maliciosos como cavalos-de-tróia, alterando um ficheiro presente nos computadores de utilizadores finais, chamado "hosts". Este ficheiro, encontrado na maioria das versões do Windows e outros sistemas operativos, inclui uma lista de nomes de sites associados a determinados endereços electrónicos. Se estes endereços forem alterados, o computador do utilizador poderá direccioná-lo para um falso site sempre que o nome de um site legítimo presente na lista for digitado no browser.

Ataque aos utilizadores

Estas invasões dão-se no computador da vítima fazendo modificações nos ficheiros hosts. Ao contrário dos ataques nos servidores DNS estes apenas afectam a máquina que infectam e é necessário o utilizador clicar em alguma ligação ou instalar algum programa. Pode haver desde a instalação de spyware's para propaganda e publicidade, à imitação perfeita de um site bancário que tem por objectivo roubar os dados da vítima (aqui entra a junção do Pharming com o Phishing).

Medidas de protecção

Instale um antivírus, mantenha-o activo e actualizado.

Mantenha as aplicações instaladas no seu computador actualizadas, nomeadamente o sistema operativo e o seu browser.

Utilize uma firewall para controlar e verificar a comunicação do computador com a Internet.

Em caso de dúvida da veracidade de uma página ou e-mail, contacte a entidade mas não utilize os contactos existentes no e-mail ou página.

Conclusão

Depois de lido este artigo estará certamente mais informado e mais ciente do que pode acontecer, de como ser burlado on-line, mas isto é apenas uma gota no oceano da segurança porque o mundo das burlas on-line é imensamente vasta e não é possível abordá-las todas em conjunto. No entanto, o Pharming é uma das técnicas de burla on-line mais usada em todo o Mundo por isso é bom que fique de alerta de forma a não cair em erros desnecessários.

João Pereira

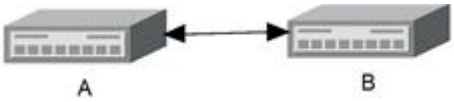
Rede CAN

Introdução

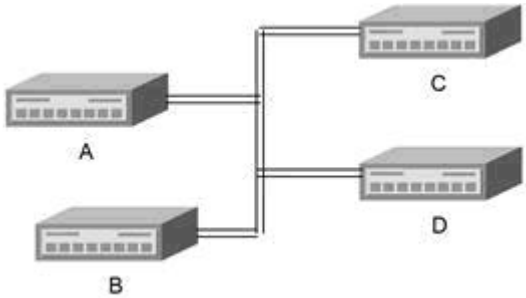
No seguimento do artigo Microcontroladores da edição 4 desta nossa revista vamos abordar neste artigo, alguns conceitos simples introdutórios das topologias das redes de dados. Para exemplificar a aplicabilidade destas redes vamos usar microcontroladores da Microchip (Pics). No final serão apresentados alguns exemplos práticos relacionados com o conteúdo do artigo.

Conceitos de ligações de dispositivos

Nalguns sistemas electrónicos há a necessidade de manipular e tomar decisões com base em informação que não está localmente acessível, bem como há a necessidade de transferir informação para outros locais. Para se poder lidar com estas situações existem "redes" que interligam dois ou mais pontos de interesse. De um modo simplista, pois existem várias variantes de interligações, estas podem ser ponto-a-ponto ou multi-ponto. Tal como os nomes indicam, a primeira liga um ponto a outro ponto.



E a segunda liga vários pontos, onde é possível que exista troca de informação entre todos, podendo existir um master ou não…



Em ligações ponto a ponto ponto a ponto: SPI, RS232… Ligações multiponto : I2C, USB, RS422, RS485, CAN (Controller Area Network).

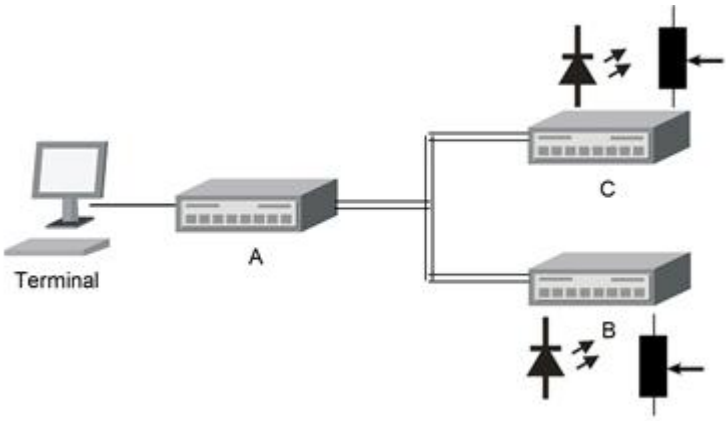
Rede CAN e microcontroladores

Define apenas a ligação física e o controlo do acesso ao meio de transmissão (ligação de dados).Foi desenvolvido para aplicações embedded (ligação dos controlos nos motores de automóveis) - Bosch 1991.

Uma rede Can é composta por um grupo de nós. Cada nó pode comunicar com qualquer outro nó da rede. A comunicação é suportada por pacotes robustos denominados de Mensagens. O protocolo Can faz uso do CSMA/CD-CR (Carrier Sense Multiple Access / Collision Detection and Collision Resolution), ou seja, cada nó antes de enviar informação para a rede, "escuta" o que se passa na rede, se eventualmente esta estiver livre, então coloca a sua mensagem na rede, se dois ou mais nós tentarem colocar informação na rede ao "mesmo tempo", então dá-se uma colisão. Ambos detectam essa colisão e esperam um tempo aleatório até voltar a transmitir.

Ao contrário do artigo da edição 4, onde foi usado um PIC 16f877 neste artigo irão ser usados PIC18f2580. O objectivo do trabalho será pôr 4 Pics a comunicar entre si. Para fácil implementação vamos usar um Master e 2 Slaves. De notar que o CAN não implica que exista um Master, mas para que seja mais simples a aprendizagem, adopta-se esta estratégia.

Em cada nó vamos ter um Led e um potenciómetro. O utilizador, através do terminal irá escolher um nó e pede a esse nó o valor do potenciómetro, ou então pede a um determinado nó para acender o led x vezes. Mais uma vez a comunicação entre os Pic master e o PC será feito com uso da porta série (ou usb com adaptador), tal como apresentado no artigo da 4ª edição.



Para não tornar este artigo muito extenso nem muito específico, assume-se que as funções da USART, ADCs já estão desenvolvidas. Visto o protocolo CAN ter muitos pormenores, o que tornaria o tempo de realização de determinados projectos bastante grande, a Microchip disponibiliza as suas principais funções para o CAN.

Antes de começar a programar, há alguns pormenores relativamente ao modo de funcionamento dos Pics que devem ser tidos em conta. Neste artigo, não irão ser explicados, somente apresentados, momeadamente o modo de operação dos Pics, registos de Baud Rate, filtros e máscaras dos buffers CAN.

Programação:

Os nós B,C vão ser idênticos, o que vai diferir é nome pelo qual ele irão "responder" (Identifier). O nó A ao servir de Master, vai ser o responsável pela recepção e apresentação da informação. Assim começando pelo nó Master:

Em primeiro lugar há que começar por fazer a inicialização dos vários módulos.

```
void main(void)
{

  TRISA=0xff;//input port
  ADCON0=0b00000001; //channel AN0
  ADCON1 = 0x00;   //Vref=vss&VDD  All PORTA analogic I/O
  ADCON2=0b00001010;//rigth justified;2tad;fosc/32

  //Inicialização da USART

  //SYNC = 0, BRGH = 1, BRG16 = 0  fosc=40Mhz,
  //baudrate=115200  -->spbrgh=21

  OpenUSART(USART_TX_INT_OFF & USART_RX_INT_OFF & USART_ASYNCH_MODE
           &USART_EIGHT_BIT & USART_CONT_RX & USART_BRGH_HIGH, 21);

  //Inicialização do móodulo ECAN

  ECANInitialize();
  INTCONbits.GIE = 1; //Enable interrupts
```

O Master só irá actuar consoante a informação recebida do utilizador. Usando uma interface com base no Hyper Terminal, assim o Master vai ser responsável pelo envio de strings que irão solicitar a informação ao utilizador. Outra estratégia poderia ser tomada, caso a interface fosse baseada em Visual Basic, Java..., pois nessas circunstâancias do ponto de vista do controlado, só interessava a "ordem" propriamente dita.

Ver no final a estrutura da função print_menu().

Esta irá ter como valor de retorno o char 'r' caso seja para pedir informação a um nó ou 's' caso seja para enviar:

```
switch(print_menu())
{
  case 'r':
    Putsr("\r\r\r\r\r\r\r\r");
    Putsr("\r######################");
    Putsr("\r#_Select NODE");
    Putsr("\r-->");
    can_rx=Re();
```

Recebido o nome do nó tem de se verificar se ele existe, caso não exista aborta-se a função e volta-se a pedir uma nova função:

```
  if(! find_node(can_rx))
  {
    Putsr("\rNODE DOESNT EXIST\r");
    Putsr("\r press any key to
                        continue");
    Re();
    can_rx=0;
    break;
  }

break;
```

Caso a função pedida pelo utilizador seja a de envio então é recebido um 's':

```
case 's':
  Putsr("\r\r\r\r\r\r\r\r");
  Putsr("\r########################");
  Putsr("\r#_Select NODE");
  Putsr("\r-->");
  can_tx=Getn();

 //teste se o nó existe
 if(!find_node(can_tx))
 {
   Putsr("\rNODE DOESNT EXIST\r");
   Putsr("\r press any key to
                        continue");
   Re();
   can_tx=0;
   break;
 }
break;
```

Caso a letra recebida não seja nem o 'r' nem o 's':

```
default:
    Putsr("\r parametro incorrecto\r");
    break;
```

Caso o nó exista em can_rx teremos o nome do nó do qual pretendemos receber o valor do potenciómetro. ADC_REQ é uma macro definida como 0xa, mais àá frente vamos usar a macro ADC_VAL que está definida como 0xb. Estas macros só servem para facilitar a programação e torná-la mais legível.

```
if(can_rx!=0)
{
  data[0]=ADC_REQ;
```

A função ECANsendMessage é um das muitas funçõoes disponibilizadas pela Microchip, para que se possa trabalhar com o protocolo CAN sem que para isso seja necessário uma grande perda de tempo na manipulação dos registos dos Pics

Assim, esta função tem como parâmetros de entrada:

- Identifier - nome do node destinatário
- Data - array onde estão as mensagens
- Size - número de elementos do array data
- ECAN_TX_STD_FRAME - flags relativas ao funcionamento do módulo ECAN.

De notar que só se avança no código quando a mensagem for enviada. É aqui neste ponto que entra o protocolo CSMA/CD-CR. Do ponto de vista do programador o que interessa é que a mensagem seja enviada, no entanto é possível saber porque é que ela não foi enviada àá primeira, mas este assunto não vai ser abordado neste artigo.

```
while(
ECANSendMessage(can_rx,data,1,
ECAN_TX_STD_FRAME) );
  Putsr("\rRequest send...");
  Putsr("\rwaiting...\r");
```

A função *ECANReceiveMessage* é formalmente identica à anterior.

```
while( !ECANReceiveMessage(&id,
data, &dataLen, ECAN_TX_STD_FRAME) );
```

O master tem um identifier igual ao byte "0xa".

```
if((id==0x')&&(data[0]=ADC_VAL))
{
  Putsr("\r\rADC VALUE\r --> ");
```

Tal como no artigo da edição 4 a ADC irá ler um valor entre 0 a 5V o que vai corresponder a um valor compreendido entre 0 e 255 (ADC de 8 bits), assim multiplica-se por 1.9 para que a sua apresentação seja legível pelo utilizador.

```
aux_char=data[1];
aux_int=aux_char*1.9;
Putn(aux_int,3,2);
can_rx=0;
}
```

Se a operação exigida pelo utilizador for a de acender o led e se o nó destinatário estiver ligado, ou seja, se existir, can_tx irá ser diferente de zero.

```
if(can_tx!=0)
 {
  Putsr("\r#########################");
  Putsr("\r#_Number of repetitions ");
  Putsr("\r-->");
  data[1]=Getn();
```

O valor 0xc (código abaixo) à primeira vista não tem nenhum significado, no entanto ele indica que a função a ser desempenhada pelo node é a de acender o led, tal como as macros ADC_VAL e ADC_REQ

```
 data[0]=0xc;
  while( !ECANSendMessage(can_tx,
           data,4,ECAN_TX_STD_FRAME));
   can_tx=0;
}
```

Assim está concluída a operação do master.

Tal como foi dito anteriormente, os outros dois nós vão ter o código muito parecido. As diferenças vão estar nos identificadores das mensagens CAN, pois um vai ter que actuar quando for recebido o caracter B e o outro tem que actuar quando for recebido o caracter C.

Assim sendo, só irá ser apresentado o código de um dos nós.

NÓ B

```
void main(void)
{

     adcInit();
     ECANInitialize();
```

Tipo de saída do porto B (in=1 e out =0), visto o led ir estar ligado na saída RB7.

```
     TRISBbits.TRISB7=0 ;
     LATBbits.LATB7=0;

     //Enable interrupts
     INTCONbits.GIE = 1;
     INTCONbits.PEIE = 1;
     can_tx=0;
     can_rx=0;

     while(1)
     {
```

Quando o nó recebe uma mensagem, vai analisar o identifier. Se o identifier for o seu, ou seja B ou C então realiza a operação

```
  while(!ECANReceiveMessage(&id,data,
      &dataLen, ECAN_TX_STD_FRAME));
    if(id=='0xb')
    {
```

Na primeira posição do array de dados do CAN está definida a função a realizar

```
  switch (data[0])
  {
    case ADC_VAL:
    data[0]=ADC_VAL;
```

Na segunda posição do array data é colocado o valor da adc. Esse valor será uma média de várias leituras…

```
    data[1]=adc_read(0);
    id=0; //send to master
    while(ECANSendMessage(id,data,4,
      ECAN_TX_STD_FRAME) );
    break;
  case LED_VAL:
    for(i=0;i<data[1];i++)
    {
      LATBbits.LATB7=1;
```

Para que seja visível ao olho humano a luz do led ao piscar tem de se criar alguns atrasos, para que a transição não seja instantânea

```
      Delay10KTCYx(255);//255ms
      Delay10KTCYx(255);//255ms
      LATBbits.LATB7=0;
      Delay10KTCYx(255);//255ms
      Delay10KTCYx(255);//255ms
    }
    break;
   }
  }
 }
}
```

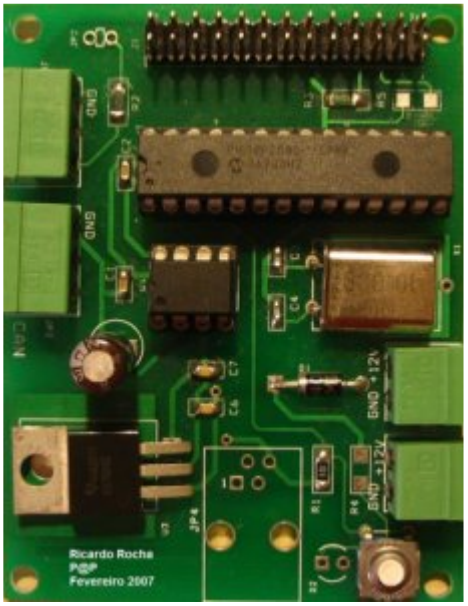
Conclusão

Embora todos estes programas sejam simples, eles são a essência de todas as redes CAN. Aqui foram usados 3 Pics. Dependendo do objectivo do projecto normalmente esta solução é dispendiosa. Na realidade o que se usa são controladores CAN cuja sua operação é idêntica ao princípio apresentado (por exemplo o MCP2515 também da Microchip).

Na prática este tipo de redes CAN podem ser usadas nas mais variadas aplicações.

O CAN foi criado pela BOSCH para o ramo automóvel. Este era usado para controlar os mais variados elementos de um carro, nomeadamente equipamentos de segurança, monitorização, controlo… O CAN ao longo do tempo já teve várias actualizações, com vista a melhorar a sua performance. Outro tipo de aplicação pode ser por exemplo a de um robô. Um microcontrolador pode receber informação dos mais variados sensores através de um barramento CAN, tal como enviar informação para os actuadores (motores…). Tomando como exemplo a figura 2, ao longo da estrutura do robô só são necessários dois fios (CAN high e CAN low) para que todos os periféricos possam comunicar entre si.

Na domótica, também se pode usar redes CAN. Supondo que se tem vários microcontroladores, lâmpadas e sensores de presença, estes só necessitam, tal como no exemplo do robô de estar ligados entre si através de um barramento CAN, o que torna simples a sua implementação.No entanto, é sempre necessário fazer o acondicionamento do sinal vindo dos sensores e controlar os actuadores que irão controlor a intensidade das lâmpadas (On-Off, PWM,...), processo este idêntico ao da edição 4.



Ricardo Rocha

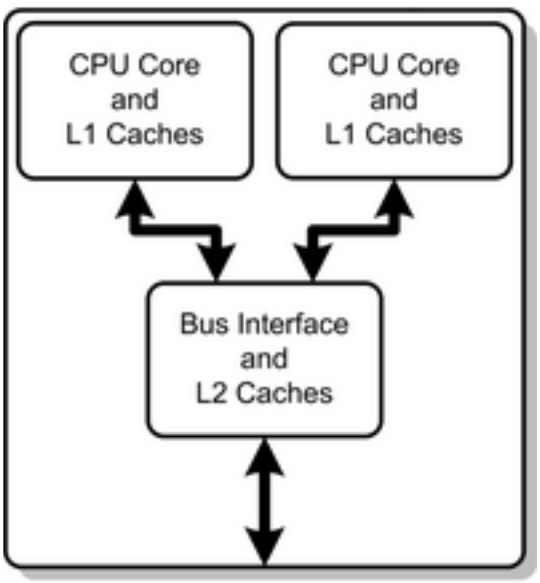
Multi-Core

Introdução

A indústria de processadores tem vindo a evoluir a um passo constante nos últimos anos. A proliferação dos computadores e das tarefas que lhes incumbimos continuam a pressionar a necessidade de processadores mais poderosos. Assim a transição para processadores multi-core torna-se num ponto crítico neste novo desenvolvimento.

O Multi-Core consiste em colocar dois ou mais núcleos (cores) no interior de um único chip. O objectivo deste design é para possibilitar ao sistema executar várias operações em simultâneo e assim alcançar melhor overall performance para responder às necessidades dos dias de hoje.

No entanto existem várias abordagens a este problema, no decorrer deste artigo vamos falar sobre tecnologias que estendem a eficiência do processador – para além dos GHz e da Lei de Moore (uma noção errada do desempenho dos processadores é associar este à velocidade do seu relógio).



Symmetric Multi-Processing

SMP é a solução mais comum no que toca a criar um sistema multi-processador em que dois ou mais processadores estão conectados a uma main memory partilhada.

Uma arquitectura SMP pode facilmente mover cargas de trabalho entre processadores com o devido suporte por parte do SO (Sistema Operativo). O lado negativo desta arquitectura é que como em sistemas com apenas um processador os acessos à memória são mais lentos que o processador a acedê-la. No caso do SMP, já que apenas um processador pode aceder à memória de cada vez, é possível que vários processadores não possam executar trabalho (starvation). Ambos os novos processadores dual-core da AMD e Intel podem ser considerados capazes de SMP.

Existem limitações ao uso do SMP, vários SO não suportam esta tecnologia (como o Windows XP Home) e não farão uso do segundo processador. Também a maior parte dos programas existentes são single-threaded, o que significa que apenas um processador pode executar os comandos desse programa não tomando partido da capacidade total do sistema.

Multithreading, Hyper-Threading, or Multi-Core?

Todos os programas são feitos de threads, sequências de instruções que o processador vai executar. Programas sequenciais são feitos a partir de uma única thread. No passado este tipo de programas dominavam o campo do software.

Os SO desses dias eram apenas capazes de executar uma thread de cada vez o que resultava num bloquear do computador sempre que mais que um programa era executado ao mesmo tempo.

Com o tempo foram surgindo novas tecnologias e nos dias de hoje os SO são capazes de múltiplos programas em simultâneo (multitasking). Tal foi alcançado por parar por um momento um programa para outro correr. Trocando rapidamente os programas que estão a ser executados o sistema faz parecer que todos os programas estão a ser executados ao mesmo tempo mas, no fundo, o processador pode apenas executar uma thread de cada vez.

Um processador normal permite que várias threads e processos sejam executados dentro da sua unidade de tempo (time slice) – isto é chamado multithreading.

Um avanço desta tecnologia é o Super-threading onde o processador pode executar várias instruções de diferentes threads em cada clock cycle (ciclo de relógio), assim ciclos que não sejam usados por uma thread podem ser executados por outra. Mas é provável que uma thread não utilize todos os recursos providenciados pelo processador (como as execution units) dentro do seu time slice, assim SMT (Simultaneous Multithreading) permite que múltiplas threads possam executar diferentes instruções no mesmo ciclo.

HTT (Hyper-Threading Technology) é uma trademark da Intel de SMT embora mais avançada que super threading que chega a permitir que duas threads diferentes sejam executadas ao mesmo tempo onde uma thread usa os recursos parados da outra. Um exemplo seria enquanto uma thread executa operações de vírgula flutuante, uma outra poderia executar operações inteiras em simultâneo.

A ideia por trás da HTT é duplicar a actividade no processador para reduzir o problema de um cache miss que reduzem o tempo de resposta deste.
Embora esta tecnologia aparenta para uma parte dos SO como sendo dois processadores diferentes mas esta não é comparável com os Dual core visto que ambas as threads partilham a mesma pipeline e cache. HTT melhora a performance do sistema mas nunca tanto como em sistemas dual/multi core já que não consegue realizar duas operações iguais. Esta diferença torna-se óbvia quando juntamos estas tecnologias aos dual-core fazendo, em teoria, quase quadruplicar o desempenho final.

Implicações no Software criadas pelo Multi-Core

Quanto mais opções para resolver um problema existirem, mais fácil se torna este de resolver. Uma estrada com apenas uma via é mais susceptível a ficar congestionada por transito que uma com duas vias. Numa estrada de apenas uma via a velocidade do trânsito fica condicionada ao elemento mais lento.

No caso dos processadores a regra é a mesma. Até agora temos vindo a assistir a métodos que melhoram o desempenho como as threads que permitem ao computados executar diferentes operações quase em simultâneo. Mas quase não basta, nunca poderá igualar outro igual a si e é esse o conceito de multi-core.

As tecnologias a serem desenvolvidas como o multi-core dependem do paralelismo, isto é, múltiplas actividades a serem executadas ao mesmo tempo. Servidores típicos possuem múltiplas portas por caixa. O paralelismo começa a tomar importância e se os produtores de software quiserem ser relevantes, terão de aprender a lidar com isso.

Um bom começo para analisar o problema seria observar o que tem vindo a ser feito no caso dos super-computadores. Estes computadores atingem o seu nível de computação através da conexão existente entre vários mais pequenos computadores. Existem duas arquitecturas distintas: memória distribuída múltiplas instruções múltipla data (MIMD) ou memória partilhada MIMD. É chamada de 'memória partilhada' o sistema onde existe um endereço onde todos os elementos a serem processados partilham. Se a memória é distinta e os elementos a serem processados apenas podem interagir através da rede entre eles, chamamos o sistema de 'memória distribuída'.

No futuro vamos assistir a um crescimento explosivo no número de sistemas com base em memória partilhada. E com o crescimento do multi-core sistemas de memória partilhada passarão a ser considerados uma norma. Estes eventos irão presentear o programador com plataformas de hardware paralelas.

Existem duas opções para lidar com estes problemas dependendo do trabalho e das necessidades do consumidor. A primeira é completar uma colecção de trabalhos no menor tempo possível. Neste caso cada trabalho individual pode correr num único processador. Tudo o que o programador se precisa de preocupar é que o software que controla os tempos de execução dos trabalhos consiga lidar com estes de uma maneira eficiente.

A segunda opção ocorre quando os trabalhos individuais precisam de ser completados no menor tempo possível. Um exemplo seria o mundo dos negócios onde são necessárias respostas rápidas para suportar decisões em tempo real. Neste caso apenas um trabalho precisa de ser executado em menos tempo. Chamamos a isto tirar partido do paralelismo dentro de um único trabalho para permitir a este que corra em menos tempo para um tamanho do problema dado, "programação paralela."

Programação paralela tem sido durante anos o domínio dos programadores de software de computação de alta performance (HPC, high performance cumputing). Hoje, contudo, todos os programadores de software precisam de perceber programação paralela.

Inicialmente na sua maior parte, senão todos, os programadores são treinados para escrever programas sequenciais. Mais tarde aprendem que existem outros modelos de programação sendo um deles a programação paralela que temos vindo a descrever. Isto significa que em vez de executar o programa numa dada sequência, uma instrução de cada vez, este vai ser executado por diferentes entidades em simultâneo. Um exemplo onde programação paralela se torna absolutamente necessária é em servidores de modo a estes poderem responder a todos os pedidos feitos por possíveis clientes em simultâneo.

A fundação da programação paralela é a concorrência: a condição de um sistema em qual duas ou mais tarefas estão activas em simultâneo.

A programação paralela permite aos programadores dividir um programa para este executar operações de acesso ao disco enquanto o processador se encarrega de efectuar cálculos, isto em simultâneo, algo que nunca seria possível num programa sequencial.

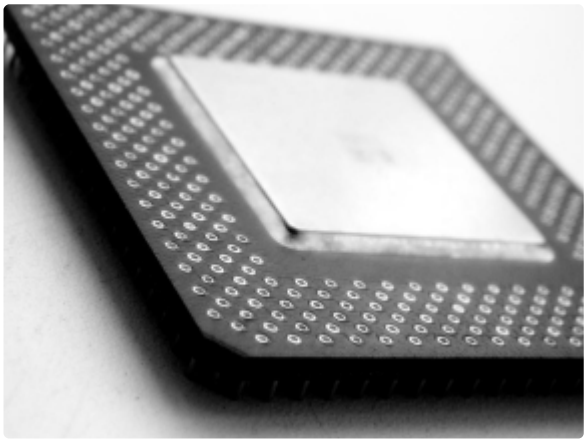
Conclusão

Pretendeu-se com este artigo proporcionar um conhecimento geral sobre as tecnologias a serem desenvolvidas, os seus propósitos e os seus suportes. Assim o leitor de certo conseguirá lidar com certas decisões da próxima vez que se encontrar face-a-face com estas novas tecnologias.

Infelizmente, o mundo por traz do multi-core das suas vantagens e aplicações no seu todo não pode ser descrita num trabalho tão pequeno nem sequer em papel.

Há 30 anos atrás um engenheiro da Intel aproximou-se de Gordon Moore com uma ideia para um computador que poderia ser usado em casa. Moore perguntou as vantagens que tal poderia trazer e tudo o que o engenheiro conseguiu imaginar foi que esse instrumento poderia ser usado para as donas de casa armazenarem as suas receitas. Moore não conseguiu imaginar a sua mulher com receitas num computador na cozinha e declarou que a ideia não fazia sentido nenhum.

Hoje existem mais de meio bilião de usuários de PCs em todo o mundo com as mais diversas aplicações. O futuro ainda reserva muitas surpresas e quem sabe se o multi-core não será apenas mais que um instrumento de transição.



Guilherme Rodrigues & Nuno Correia

Revista Linux

A Verdadeira Revista Portuguesa de Linux

Edição Bimestral em formato PDF

Download Gratuito

www.revista-linux.com

Programação em C (1ª Parte)

Este artigo vem na sequência de um anterior, "Programação em Scheme", que saiu na edição número 6. E como tal, visa estabelecer uma evolução lógica do programador principiante. Embora o Scheme seja uma linguagem excelente para se aprender a programar, não é ficarmos agarrados a uma linguagem para sempre que seremos bons programadores. O saber programar difere do saber muitas linguagens de programação na medida em que depois de se saber programar, basta estudar alguma sintaxe, que todas as linguagens são programáveis.

Para um programador, o primeiro passo no início da criação de um novo programa, deve ser a busca da melhor ferramenta e linguagem para o fazer. Portanto, sendo o Scheme uma linguagem para dar bases de conhecimento de programação, o C é uma linguagem, de baixo nível e, por isso, com um potencial extremo, onde é possível operar tudo sem excepção.

A Linguagem C foi desenvolvida por Dennis Ritchie em 1972 para ser usada nos sistemas UNIX. É uma linguagem que se baseia no paradigma procedimental, utiliza passagem de referência através de ponteiros e endereços de memória, o que a torna uma linguagem minimalista e de baixo nível em termos de acesso à memória, sendo por isso, muito versátil para criar software [parecida com o Assembly] e adapta-se bem ao hardware. Como na maior parte das linguagens, é possível criar-se bibliotecas para expandi-la. O C serviu de evolução para muitas linguagens, onde a principal e mais influenciada linguagem foi o C++, desenvolvida por Bjarne Stroustrup em 1983.

Não vamos entrar em grandes pormenores mas para trabalharmos em C precisamos de um compilador e de um editor de texto do estilo Notepad2 ou Gedit, para Windows e Linux respectivamente.

Para Windows, existe o Devcpp que tem um compilador de C e de C++, tem um editor de texto incluído e é freeware. Posto isto, podemos começar...

Hello World

Não sejamos originais e por isso aqui vai o primeiro programa universal de qualquer linguagem de programação que se preze: "Hello World!".

```
#include <stdio.h>
int main()
{
        printf("Hellow World\n");
        return 0;
}
```

A única coisa que esta programa faz é imprimir no ecrã a frase "Hellow World" e acrescentar-lhe uma nova linha. Embora só faça isto, tem muito que se lhe diga, por isso aqui vai a explicação pedaço a pedaço:

#include <stdio.h> diz ao compilador para incluir a biblioteca stdio.h onde, entre elas, está definida a função printf(). Ao fazermos #include "ficheiro.h" estamos a dizer que já criámos um ficheiro, anteriormente, onde estão definidas funções que queremos usar agora neste nosso novo programa. Por agora fica assim, mais para a frente veremos melhor como realmente funciona.

int main() é a função que nunca nos esquecemos pois sem ela não há programa. Todos os programas feitos em C têm de ter a função main() definida. Para além disso, a primeira coisa que um programa em C faz é ir à função main() e correr o seu conteúdo. Neste caso, o int que precede a definição main(), podia nem estar lá.

Não vamos aprofundar agora o estudo de funções mas as funções retornam "coisas" neste caso, dissemos à função main() para ela retornar um int, ou seja, um valor inteiro. Mais para a frente será explicado o significado do retorno de um valor inteiro na função main, através da penúltima linha deste pequeno código, return 0;.

{ } isto é um bloco. As chavetas servem para delimitar um bloco de código que tenha mais do que uma instrução (embora um bloco de código possa ter apenas uma instrução). Neste caso, a função main() possui duas instruções, o printf(), lá chegaremos, e o return 0.

Já dentro do bloco da função main() temos as anteriormente mencionadas instruções printf() e return 0. Uma de cada vez. A função printf() recebe uma string e mostra-a no ecrã (neste caso na consola). Ou seja, printf("Hello World\n"); imprime na consola a frase "Hello World" seguida de uma nova linha. A nova linha deve-se ao facto de a frase "Hello World" estar seguida de um caracter especial, o '\n' (newline). Existem vários caracteres especiais em C que veremos mais à frente.

E finalmente temos o return 0. Como já foi referido anteriormente, uma função se foi definida, terá de retornar. Neste caso foi definida para retornar um valor inteiro, int main(). E por essa razão, ela retorna o valor 0 [zero]. Como já foi referido, será explicado mais à frente o retorno de funções.

Variáveis (1ª parte)

Visto já sabermos fazer um programa em C, que mostra "frases" no ecrã", podemos avançar para algo mais interessante: variáveis.

Uma variável em é algo que não tem valor fixo e que, portanto, pode ser alterado e tomar qualquer valor. Isto é verdade, mas em C não é assim tão linear.

Uma variável em C pode tomar qualquer valor do seu tipo, isto é, existem vários tipos de variáveis: inteiros [int e suas variantes unsigned (sem sinal) e signed (com sinal)], de vírgula flutuante [float] doubles [caso int/float não cheguem], e caracteres [char]. Por agora estes chegam e já nos vão dar muito jeito.

Com isto acho que podemos começar com um simples programa que mostra no ecrã cada um dos tipos de dados que acabámos de ver.

```
#include <stdio.h>
int main()
{
 int inteiro = 3; //definição de um int
 float pi = 3.14; //definição de um float
 //definição de um double
 double inteiro_gigante = 1234567890;
 //definição de um char
 char caracter = 'A';
 /* Imprimir as variáveis no ecrã*/
 printf("inteiro:%d\n
   pi: %f\n
   inteiro_gigante: %d\n
   caracter: %c\n", inteiro, pi,
   inteiro_gigante, caracter);
 return 0;
}
```

Há aqui muita coisa nova e que pode parecer complicada mas que no fundo não é. Vamos por partes. O programa começa e, como já referimos, vai direito à função main().

Ao acedermos ao seu bloco, encontramos nas primeira 4 linhas, definições de variáveis, int inteiro = 3; float pi = 3.14; double inteiro_gigante = 1234567890; e char caracter = 'A'; Ao criarmos, por exemplo o int inteiro = 3, estamos a dizer ao compilador para nos reservar um pedaço [uma zona, com um endereço] de memória do nosso computador [RAM] para podermos lá guardar algo do tipo int [inteiro] e que, neste caso, contenha o valor inteiro 3.

A mesma coisa para as outras 3 definições de tipos de variáveis. Há aqui dois passos que foram evitados escrever no código para poderem ser falados agora. Na declaração de variáveis, não é obrigatório atribuir-se automaticamente um valor a ela.

Podia ter-se escrito o mesmo da seguinte forma:

```
int inteiro;
inteiro = 3;
```

que o resultado seria exactamente o mesmo.

Ou seja para além de se declarar a variável inteiro como sendo um int, atribui-se-lhe o valor 3 através do símbolo de atribuição '='. A atribuição funciona da direita para a esquerda, ou seja, o 3 é atribuído à variável inteiro e não ao contrário, e podem pensar: que estupidez, é óbvio que não é ao contrário. Mas um exemplo mostra bem o problema que muitas vezes acontece.

```
int inteiro = 3;
int outro_inteiro;
int e_outro_inteiro = 5;
outro_inteiro = inteiro;
inteiro = ainda_outro_inteiro;
```

O que acontece nesta situação é: 3 é atribuído a inteiro; outro_inteiro é criado com um valor qualquer indefinido; 5 é atribuído a e_outro_inteiro. Depois, inteiro, que vale 3, é atribuído a outro_inteiro, passando este a valer 3, em vez do valor anterior, aleatório; e_outro_inteiro, que vale 5, é atribuído a inteiro, destruindo o seu valor anterior, 3. Como vêem, o símbolo de atribuição '=' tem o "poder" de destruição e de atribuição, sendo esta, sempre da direita para a esquerda. Posto isto, e não esquecendo o resto do código deste segundo programa, voltemos a ele.

O printf() é uma função que nos permite imprimir frases. Mas e se quisermos imprimir o valor que está dentro de uma variável? No nosso caso queremos imprimir o seguinte texto:

```
"inteiro: 3
 pi: 3.14
 inteiro_gigante: 1234567890
 caracter: A"
```

Se repararem, as aspas acabam na linha de baixo pois foi acrescentado o caracter especial '\n', para além das outras linhas, no final da frase. Ora para imprimir as variáveis dentro da frase usamos a seguinte sintaxe:

printf("Caracteres normais: %d [para inteiros ou doubles] , %f[para floats] e %c [para caracteres]...

Depois de fechar aspas, põe-se o nome de cada variável separada por vírgulas, pela ordem que apareceram dentro da frase, neste caso será um int, um float e depois um caracter e no nosso caso: ", inteiro, pi, caracter);

A ideia é pôr a %dfc, etc, e no final das "" "" pôr o nome das variáveis pela ordem que foram inseridas dentro da frase. Se bem repararam pôs-se o printf() em várias linhas. Isto pode ser muito útil para se poder visualizar, mais ou menos, como vai ficar o aspecto final ou mesmo em termos de organização do código para que outra pessoa possa vir a ler e compreender bem.

Por falar em organização de código, podemos ver que também foram acrescentados comentários. Desde já, é bom hábito acrescentar comentários ao código para que, mais uma vez, qualquer pessoa que o vá ler, o possa entender ou mesmo para não perdermos um raciocínio que se estava a ter num dia e quando voltamos ao ficheiro não fazemos a menor ideia do que se passa ali. Existem duas formas de comentar código em C: ou numa única linha, usando // ou em mais que uma linha, usando /* para começar e */ para acabar. E tudo o que dentro desse espaço estiver, será ignorado pelo compilador. Com estas bases estamos prontos para criar variáveis, atribuir-lhes valores, sejam elas de que tipo forem, e ainda de imprimi-las no ecrã.

Variáveis 2 e Símbolos Operadores

Embora já tenhamos aprendido a usar inteiros, floats e caracteres, seria mais interessante se pudéssemos interagir com eles: somando-os, subtraindo-os, multiplicando-os, dividindo-os, etc.

Tudo ficará mais claro com um simples programa.

```
#include <stdio.h>
int main()
{
 //declaração das variáveis a=3,b=4 e c
 int a = 3;
 int b = 4; //podia ter sido
 //declarado da seguinte maneira:
 int soma; //int a = 3, b = 4, soma;
 //que teria o mesmo efeito
 int sub,mult,div;

 soma = a + b; // soma a com b
 //e atribui o resultado a soma
 sub = a - b; // subtrai b a a e
 //atribui o resultado a sub
 mult = a * b; // multiplica a por b
 //e atribui o resultado a mult
 div = a / b; // divide a por b e
 //atribui o resultado a div
 printf("soma=%d, sub=%d, mult=%d,
    div=%d\n", soma, sub, mult, div);
 return 0;
}
```

Tão simples quanto isto. Criámos as variáveis a e b e atribuímos-lhes os valores 3 e 4 respectivamente. Depois criámos as 4 variáveis para guardarmos os valores respectivos às operações efectuadas a a e b, soma, sub, mult e div. Em seguida atribuímos a cada uma destas operações, o valor da respectiva operação entre a e b. No final, imprimimos, como já vimos, as variáveis dentro da frase. Outra maneira de ter feito isto seria a seguinte:

```
int main()
{
 //declaração das variáveis a=3, b=4
 //e soma
 int a = 3;
 int b = 4;
 int soma;
 //podia ter sido declarado da
 //seguinte maneira:
 //int a = 3, b = 4, soma;
 //que teria o mesmo efeito
 printf("soma=%d, sub=%d, mult=%d,
      div=%d\n", a+b, a-b, a*b, a/b);
 return 0;
}
```

Mas é uma maneira muito preguiçosa e na qual não poderíamos voltar a usar o valor dos resultados, como acontece na versão anterior.

Input/Output

Como já referimos no ínicio, usamos o #include <stdio.h> para podermos utilizar as funcionalidades de input, teclado, rato e outros, e output, monitor, ficheiro ou periférico. Até agora só usámos o output, com a função printf(). Vamos agora poder alargar, e muito, a maneira de usar o C, podendo interagir com o programa.

```
#include <stdio.h>
int main()
{
 //declaração de variáveis
 int a,b,soma;
 printf("Insira um valor para a: ");
 // obter valor de a
 scanf("%d", &a);
 printf("Insira um valor para b: ");
 // obter valor de b
 scanf("%d", &b);
 soma = a + b; // soma a com b e
atribui o resultado a soma
 // imprimir o resultado no ecrã
 printf("soma=%d\n", soma);
 return 0;
}
```

O scanf() pode parecer difícil à partida mas no fundo é muito simples.

No entanto, uma vez mais, vejamos do início o código deste programa. No início declarámos 3 variáveis, a, b e soma, com o objectivo de atribuirmos um valor a a e a b e depois de adicionados atribuído a soma. Passemos, então, ao scanf(). Nesta função incluímos o tipo de dados que vamos inserir e no fim, fora das aspas, o endereço dela e no caso de ser mais do que uma, pomo-las por ordem a seguir às aspas, assim como fazemos no printf().

Ao escrevermos &a e &b estamos a dizer ao compilador que ao receber dois inteiros ["%d"], queremos que sejam alocados no endereço de memória &a e &b respectivamente. Pensemos nisto como sendo casas. Temos a casa a com o endereço &a e a casa b com o endereço &b. Ou seja, o programa espera o valor para o inteiro a seguido de ENTER e depois, espera o valor de b seguido de mais um ENTER. Se percebermos bem como o scanf funciona, fica aqui um programa que recebe a data de nascimento e depois a confirma, mostrando no ecrã as opções escolhidas pelo utilizador:

```
#include <stdio.h>
int main()
{
//declaração de variáveis
 int ano;
 int dia;
 int mes;

 printf("Insira a sua data de
 Nascimento [mes em numero]\n\nAno:");
 // obter valor de ano
 scanf("%d", &ano);
 printf("Dia: ");
 // obter valor de dia
 scanf("%d", &dia);
 printf("Mes: ");
 // obter valor de mes
 scanf("%d", &mes);
 // imprimir os dados do utilizador
 // no ecrã
 printf("Data:\n%d\\\%d\\\%d\n",
                      ano, dia, mes);
 return 0;
}
```

Expressões Condicionais e Operadores Lógicos

Nesta secção vamos aprender lógica. A lógica é muito importante para decidir situações. E para as decidir existem as expressões condicionais. Exemplo disso são o if () e o switch (). Por agora vejamos estas duas.

O if () funciona da seguinte maneira:

```
if (condição)
{
 expressão 1;
}
else
{
 expressão 2;
}
```

Como já referimos anteriormente, um bloco {}, só precisa de existir, se existir mais do que uma expressão. Em C, algo é verdadeiro se for diferente de 0 [zero], ou seja, 1 é verdadeiro, 642 é verdadeiro, -2,3 é verdadeiro e 0 [zero] é sempre falso. Posto isto, vamos a um simples programa para pôr em prática o nosso if ():

```
#include <stdio.h>
int main()
{
 int x = 10;
 int user;
 printf("Insira um numero maior que
                              %d: ", x);
 scanf("%d", &user);
 if (user > x) // se condição for
 // verdadeira, executa o seu corpo
 {
  printf("Inseriu o numero %d que é
maior que %d\n", user, x);
 }
 else // caso contrário
      // executa este corpo
 {
  printf("O numero %d que inseriu
     nao é maior que %d\n", user, x);
 }
 return 0;
}
```

Como podemos ver, é muito fácil perceber a sintaxe do if(). Se o número inserido for maior do que x, que vale 10, o corpo do if é executado, caso contrário, o corpo do else é executado. Não é obrigatório que o if tenha else. Esta lógica é muito simples mas preciosa. Por outro lado, existem operadores lógicos que nos ajudam a precisar uma condição. São eles o AND o OR e o NOT, que em C se representam por, &&, || e ! respectivamente.

E ser-nos-ão muito úteis no seguinte programa [uma versão 2 do anterior].

```
#include <stdio.h>
int main()
{
 int MIN = 5, MAX = 10;
 int user;
 printf("Insira um numero maior que
     %d e menor que %d: ", MIN, MAX);
 scanf("%d", &user);
 if ((user > MIN) && (user < MAX))
 // executa o seu corpo do if
 {
  printf("Inseriu o numero %d que
  é maior que %d e menor que %d\n",
  user, MIN, MAX);
 }
 else
 // caso contrário executa este corpo
 {
  printf("O numero %d que inseriu
  nao é maior que %d e menor que %d
  \n", user, MIN, MAX);
 }
 return 0;
}
```

Como podemos constatar, neste caso, o corpo do if só é executado se o valor inserido pelo utilizador satisfizer, obrigatoriamente, as duas condições, ser maior do que MIN, que vale 5, e ao mesmo tempo, ser menor do que MAX, que vale 10. É fácil perceber estes conceitos lógicos nas condições, o importante é praticar para ganhar "traquejo" na matéria. Posto isto passemos ao switch (). Esta função é muito importante quando queremos condicionar uma variável a uma escolha por opções. A sua sintaxe é a seguinte:

```
switch (variável)
{
 case 'caso_1': { expressão_1 }
break;
 case 'caso_2': { expressão_2 }
break;
 default
 {
  expressão
 }
}
```

Uma vez mais, pode parecer muito complicado mas é verdadeiramente simples e vai-nos dar imenso jeito. Passando à sua explicação: o switch recebe a variável a analisar. Depois, já dentro do corpo, temos todos os casos que queremos e por ordem, cada um seguido de break, de modo a que uma vez executado o primeiro "case" que der verdadeiro, o switch pare por aí e não execute todas os outros "case", e é finalizado por uma condição default, caso nenhuma das anteriores seja verdadeira. Um programa para explicar o switch. Porque não um menu de um leitor de música?

```
#include <stdio.h>
int main()
{
 int menuOption;
 printf("Escolha uma opcao do
 menu:\n\n");
 printf("1-Play\n");
 printf("2-Pause\n");
 printf("3-Stop\n");
 printf("4-Next\n");
 printf("5-Previous\n\n");
 printf("Opcao: ");
 scanf("%d", &menuOption);
 switch (menuOption)
 {
  case 1: printf("Escolheu a opcao 1-
          Play\n"); break;
  case 2: printf("Escolheu a opcao 2-
          Pause\n"); break;
  case 3: printf("Escolheu a opcao 3-
          Stop\n"); break;
  case 4: printf("Escolheu a opcao 4-
          Next\n"); break;
  case 5: printf("Escolheu a opcao 5-
          Previous\n"); break;
  default:
   printf("Opcao invalida\n");
 }
 return 0;
}
```

Ou seja, depois de mostrarmos o menu, utilizamos o scanf() à variável menuOption para determinar a opção escolhida e conforme esta, cada "case" dirá a respectiva opção escolhida ou "Opção invalida" caso nenhum dos "case"s seja verdadeiro. Na próxima edição prosseguiremos com esta introdução ao C.

Miguel Wahnon

Ambientes gráficos GNU/Linux em sistemas de fraco desempenho

A quem faz por não perder pitada de cada novo passo no reino das Novas Tecnologias, a quem é capaz de estar meses e meses a juntar o pé de meia para comprar aquela placa gráfica ou processador topo de gama, pode parecer relativamente estranho que ainda haja quem trabalhe na primeira versão do Pentium. Mas, mais (supostamente) raro existirem será, dentro dessa mesma minoria, haver quem não queira perder o prazer de utilizar GNU/Linux, e que como tal procure saber as melhores opções tendo em conta a sua situação.

GNU/Linux é, afinal de contas, uma das melhores soluções no "mercado" para esta gama de utilizadores, como tal, neste artigo decidimos dar umas luzes sobre o principal factor de gasto de desempenho numa distribuição: o ambiente gráfico.

É-nos muitas vezes, em vários fóruns em que participamos, perguntado qual a melhor distribuição para low pc's. A resposta é simples: não depende da distribuição, depende do ambiente gráfico que se utiliza (geralmente!), escolhendo-se muitas vezes a distribuição em função disso. É nessa altura que vem à baila a questão: *Sendo assim, qual o melhor ambiente gráfico?*

Não é a esta pergunta que pretendemos responder, mas queremos expor-vos as principais vantagens e características das duas mais adoptadas escolhas para low pc's: XFCE e Fluxbox.

Fluxbox

A palavra de ordem neste pequeno ambiente de trabalho minimalista é mesmo a simplicidade em termos estéticos e funcionais. Desde logo, há a realçar que o Fluxbox não é mais que um Windows Manager, como tal, não é de esperar um leque variado de aplicações com uma raiz comum a suportar a utilização de produtividade (exemplo: aplicações K do KDE, que é um Desktop Environment). O Fluxbox só se encarrega de modificar o sistema gráfico Xorg de maneira a incluir, entre outras coisas, uma pequena barra de tarefas, a possibilidade de decorar o fundo do ambiente com um wallpapper e um menu de aplicações accionável pelo clique direito do rato, além da possibilidade de personalização por via de temas. Logo, todas as aplicações a utilizar terão de ser instaladas pelo próprio utilizador, mas não se preocupe, FluxBox é compatível com as aplicações de GNOME e KDE.

Este pequeno ambiente minimalista não é propriamente algo novo, é baseado noutro WM: o BlackBox. Sendo muito semelhante a este, principalmente visualmente, tem-se progressivamente destacado e ganho a popularidade que o torna numa escolha a considerar para pc's de baixo desempenho. Em relação ao BlackBox apresenta mais algumas funcionalidades importantes, como uma barra de tarefas configurável, a possibilidade de utilizar atalhos de teclado, e poder agrupar todas as janelas em apenas uma com separadores. Versões mais recentes apresentam inovações no design como janelas com cantos arredondados e transparências.

Simplicidade?

Não será preciso muito tempo para que fique a conhecer os cantos à casa deste WM, com utilização intensiva poderá até estranhar e não se habituar a mais nenhum sistema que não dê ao clique direito do rato um papel fulcral na acção. No entanto, tudo isto tem um custo, ou nenhum, dependendo do utilizador que seja. Além da sua simplicidade, FluxBox é também conhecido por conseguir apresentar um sistema com um design actual e com uma boa dose de eyecandy, mas não é referido como chegar lá. A verdade é que este pequeno sistema é totalmente configurável mas, para o fazer, terá de se habituar, tanto à utilização da consola, como a investir algum tempo em aprender a adicionar novos temas, editar o menu e definir aplicações de começo automático no startup do sistema, tudo configurável em ficheiros de texto simples. Além disso, e apesar dos muitos temas existentes na Internet, nem sempre ficamos contentes com o obtido, e estaremos a um passo de andar a vasculhar nos ficheiros dos temas, e a editá-los a nosso belo prazer.

Se se refere o abominável que isto possa ser para um utilizador que apenas quer algo leve e pronto a usar, há que referir também o espantoso desta personalização extrema a que este pequeno sistema chega, o que é um chamariz para utilizadores com mais interesse em aprender, ou mais conhecimentos.

É rápido, ao menos?

Neste campo não há a menor dúvida, FluxBox é o WM mais simples do reino GNU/Linux, que incorpora igualmente um maior número de funcionalidades. A sua source não tem mais de 1MB, demorando apenas alguns minutos a compilar. Mal se entra poder-se-á desde logo constatar a rapidez com que se consegue utilizar este pequeno sistema, geralmente ocupando, sem mais nenhuma aplicação exterior aberta, tendo apenas o sistema base por trás e sem swap utilizada, aproximadamente 39MB de memória. Isto permitirá corrê-lo até nos pc's mais antigos sendo necessária, no entanto, a consciência de que qualquer aplicação, por mais simples que seja, elevará logo a memória para valores mais altos e quiçá abusivos, como aliás é lógico.

Se se quiser reduzir ainda mais a memória ocupada, poder-se-á optar por uma sistema base mais leve do que o que nos é proporcionado por uma normal distribuição com a opção de instalar Fluxbox (como Debian, Gentoo, SuSe, etc), caindo a escolha no bastante famoso Damn Small Linux, uma distribuição de 50MB de tamanho que inclui FluxBox.

Conclusão

Se gosta de aprender, e quer o melhor que existe não comprometendo a performance, FluxBox é a escolha a tomar. O seu design simples e funcionalidade já conquistaram muitos, de qualquer forma, não há como experimentar e tirar a prova dos nove. Se não quiser instalar propositadamente este ambiente gráfico, poderá simplesmente fazer o download do LiveCD da distribuição Damn Small Linux e comprovar o que leva muitos, mesmo possuindo pc's de elevada performance, a tê-lo como escolha nº 1.

Xfce

Podemos caracterizar Xfce numa única frase: Desktop Environment leve sem desprezo da funcionalidade. Tem na rapidez, em relação aos dois DE's dominantes, o seu principal trunfo. Segue uma filosofia modular, ou seja, vem com um sistema base sem muitas dependências a que se pode facilmente acrescentar funções à medida das necessidades, através da instalação de pacotes. Assim, consegue ocupar menos recursos pois não instala por default funções muitas vezes desnecessárias ao utilizador.

O nome Xfce provinha originalmente de "XForms Common Environment", mas como foi reescrito duas vezes já não usa XForms. O nome continuou o mesmo, já não sendo "XFce" mas "Xfce", não significando a sigla absolutamente nada, tal como o logo, que é apenas uma piada.

O desenvolvimento começou por volta de 1997 sendo um clone para Linux do Common Desktop Environment (CDE) um DE proprietário para UNIX. A segunda versão foi lançada já com o Xfce's window manager (Xfwm) ainda hoje presente. Foi na terceira versão que deixou Xforms e passou a basear-se em GTK, principalmente por causa de o primeiro ser proprietário, o que limitava o progresso do projecto. Com a versão 4.2.0 foi feito um grande salto com o update para as bibliotecas GTK + 2 e na 4.4 (actual) passou a usar o Thunar como file manager em vez do Xffm.

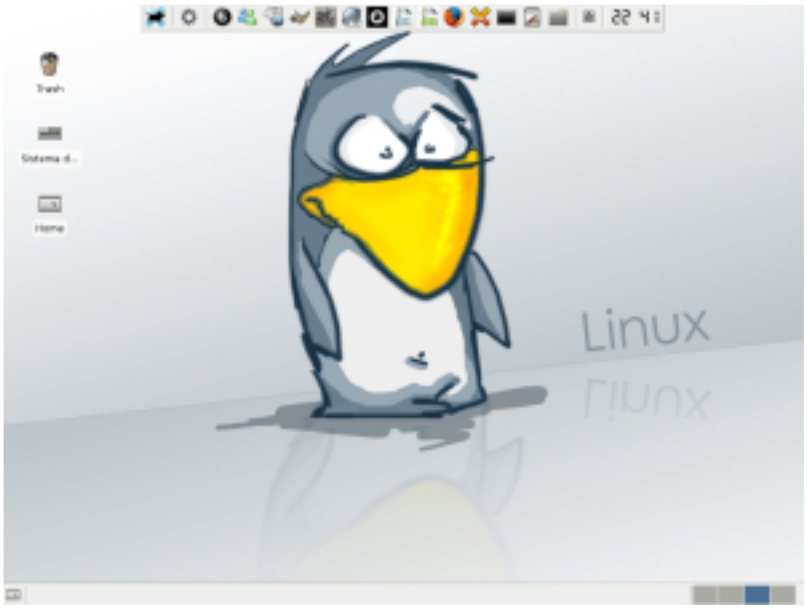
Simples e Funcional

Usa a biblioteca GTK (tal como GNOME) e vem por default com o Thunar como file manager. Este último criado foi para ser "leve" quando comparado com, por exemplo, o nautilus (file manager default do GNOME), apesar de, em relação a este, sacrificar algumas funcionalidades (tais como aceder por ssh).

A última versão consegue ser um bom substituto de GNOME ou KDE. É-nos apresentado um Desktop simples, com duas barras horizontais como em GNOME, que o torna bastante prático. Possui um painel simples mas bastante útil permitindo o fácil acesso a várias configurações do sistema. Ocupa cerca de 55Mb de memória com a SWAP totalmente livre, tendo por trás o sistema base a correr. Vem com transparências e efeitos sombra nativos (necessita de aceleração 3d embutida no desktop), apresentando uma excelente fluidez.

Nota final

Se quiser apenas um DE leve ou mesmo um substituto para GNOME ou KDE, tem no Xfce uma boa aposta. Além de ser fácil de utilizar, principalmente para quem vem do GNOME, tem um aspecto limpo e de personalização fácil. De notar que para computadores quase "pré-históricos" provavelmente fluxbox é mais indicado...



Celso Ramos & Miguel Pais

Encontro Nacional de Tecnologia e Inovação

TECNONOV 2007

27 de Janeiro de 2007

No passado dia 27 de Janeiro, decorreu o TECNONOV 2007, um encontro nacional dedicado à Tecnologia e Inovação.

Este evento veio dar continuidade a outros já decorridos sobre a mesma área de acção, como BarCampPortugal e a SHiFT, sendo este último mais virado para um público internacional. O TECNONOV, pelo contrário, tinha a intenção de mostrar um pouco do que se faz no nosso país, aos portugueses, e principalmente fomentar a partilha de ideias e motivar a inovação.

O encontro decorreu em Coimbra, no café/auditório da FNAC, localizada no Fórum Coimbra, espaço que, por vezes, não foi suficiente para o público que ia passando e resolvia espreitar. Lá dentro iam decorrendo apresentações, maioritariamente dirigidas a um público não-profissional , com excepção de alguns temas, como a terceira versão da licença GPL, que era direccionada para a área de desenvolvimento de software. Ao todo, foram sete apresentações, realizadas por oradores na sua grande maioria profissionais da área, mas com historiais distintos. Durante todo o evento estiveram também disponíveis gratuitamente cds da distribuição GNU/Linux Ubuntu, para diversas plataformas.

A primeira apresentação, de título "Linux: Um caminho para a produtividade", ficou a cargo de Octávio Filipe Gonçalves, fundador da empresa MagicBrain. Foi uma apresentação dedicada à divulgação do "mundo" do Software Livre e de uma das suas principais "bandeiras": as distribuições de sistemas operativos GNU/Linux.

Foram desmistificados mitos que se perpetuam sobre este tipo de software e enunciadas algumas das principais vantagens.

Seguidamente, decorreu a apresentação da terceira versão da licença GNU GPL, por Rui Miguel Seabra, Vice-Presidente da ANSOL (Associação Nacional para o Software Livre). Foram clarificadas as principais diferenças desta nova versão em relação à anterior e apresentadas algumas vantagens da sua adopção. Para terminar a primeira parte do evento, Marcos Marado, analista da SonaeCom IT, realizou uma das apresentações que gerou maior interesse e discussão de todo este evento, à volta do tema "A Indústria Discográfica e o DRM (Digital Rights/Restrictions Managment)". Após a descrição desta tecnologia, foram apresentadas diversas conclusões negativas a tirar da sua utilização, justificando assim as acções que o orador incentivava, de forma a combater a aplicação de tais restrições na indústria discográfica.

Após um curto coffee break, Mário Lopes, estudante na FEUP, veio apresentar o IMAGE (Industrial Management Game), projecto composto por uma equipa de estudantes do curso de Engenharia Informática. Este consiste num jogo web-based de simulação de gestão e desenvolvimento industrial. Tendo por base aquilo que realmente se utiliza no sector, este jogo torna-se assim numa boa plataforma para testes de tácticas e planos empresariais.

Aproveitou-se para abordar principalmente as decisões feitas pela equipa para o desenvolvimento deste projecto, sendo dado um maior foco à plataforma web escolhida, Ruby on Rails.

O fundador da WeBreakStuff, empresa portuguesa dedicada ao desenvolvimento, design e estratégia para produtos web, Frederico Oliveira, veio falar sobre o tema "Inovação em Portugal". Foram feitas diversas críticas à falta de empreendedorismo da maioria dos portugueses e à falta de oportunidades que são dadas a novas ideias, comparando com outros pontos do globo.

Seguiu-se uma Introdução à linguagem de programação Perl, embora mais virada para o aspecto social, ou seja, para as comunidades e suporte gerado à sua volta: CPAN, mailing lists em língua portuguesa, Perl Mongers, entre outros... Apresentação esta que foi realizada por Nelson Ferraz, profissional da Segula Technologies.

Para terminar, a última apresentação ficou a cargo de Pedro Sousa, programador web para a Accenture, que já havia apresentado o seu projecto na 5ª edição da Revista PROGRAMAR, o weSpendMoney. Esta foi uma continuação do tema "Inovação", desta vez apresentando alguns bons exemplos, dando especial enfase à Wii, a nova consola da Nintendo.

Em suma, foi, sem dúvida nenhuma, uma óptima iniciativa, pois para além de surgir mais um espaço de divulgação do que se faz em Portugal nesta área, surgiu principalmente mais um espaço de sensibilização, motivação e incubação de ideias, prontas a ser partilhadas e desenvolvidas. Este é um dos caminhos para atingir a recorrentemente falada "Inovação". Felizmente é já possível observar um resultado prático deste evento: um local de divulgação e sensibilização sobre as práticas de algumas empresas, ao inserirem DRMs nos seus produtos (http://www.drm-pt.info).

Para mais informações sobre este evento, visitem o site oficial http://tecnonov.net

* **TECNONOV 2007**
  Encontro Nacional de Tecnologia e Inovação

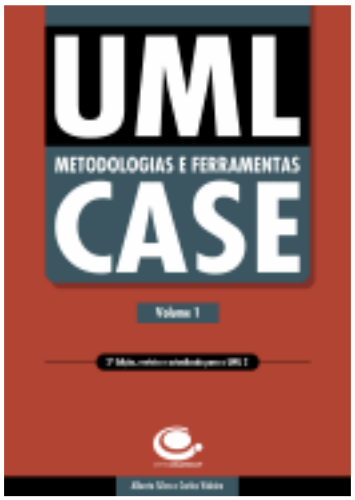
TECNONOV 2007 COIMBRA

* 27.01. SÁB 15H00

* O TecnoNov pretende ser um Encontro Português sobre Tecnologia e Inovação, a par de outros que têm decorrido, como o BarCamp Portugal, num contexto mais ad-hoc e para um público nacional, e a SHiFT, também ele um encontro social mas dirigido a um público internacional. Esta iniciativa assume-se como um evento de referência, abordando temas emergentes na área das novas tecnologias e dirigindo-se a um público não profissional. O grupo de oradores é proveniente de várias empresas portuguesas.

Sérgio Santos

UML, Metodologias e Ferramentas CASE



Alberto Silva
Carlos Videira

Editora: CentroAtlantico.pt
Colecção: Tecnologias
Páginas: 357
2ª Edição, 1ª Edição: Março de 2005
ISBN: 989-615-009-5

UML, Metodologias e Ferramentas CASE, volume 1, apresenta-se como um livro de referência para alunos de licenciaturas e de cursos de pós-graduação, na área de concepção de sistemas de software, com especial foco em métodos, modelação e ferramentas CASE (Computer Aided Software Engineering).

Introduzido o livro, o primeiro impacto ao correr das páginas não impressiona. O aspecto gráfico parece indicar que o livro foi desenvolvido num corriqueiro editor de texto, ficando muito aquém da qualidade gráfica que as editoras de renome mundial habituaram os profissionais e os alunos do ensino superior. Este primeiro impacto consolida-se ao longo do livro, onde se juntam os imensos gráficos e diagramas cuja qualidade gráfica fica abaixo das expectativas, ao ponto de tornar difícil a leitura em alguns casos.

Centrando a atenção no conteúdo e não na forma, o livro divide-se em duas partes. A primeira efectua o enquadramento introduzindo uma visão geral sobre os sistemas de informação, arquitecturas, planeamento, processos e métodos de desenvolvimento de software. Não sendo este o objectivo do livro em questão, esta primeira parte permite que um leitor fora da área de tecnologias de informação se familiarize com conceitos e conhecimentos básicos da área.

A segunda parte centra-se na linguagem de modelação UML (Unified Modeling Language), sendo a segunda versão do UML a grande razão de ser desta reedição. Na segunda parte efectua-se uma enumeração das várias modelações suportadas pelo UML, efectuando uma descrição, introduzindo os diagramas correspondentes e fazendo referência, de forma superficial, às limitações das versões anteriores do UML, mas não abordando possíveis problemas e soluções na actual versão. Um ponto interessante é a inclusão de exercícios no final de cada secção de forma a testar a solidez dos conhecimentos adquiridos. Era expectável que após a introdução dos conceitos UML se ilustrasse a sua utilização consolidada e a sua ligação a ferramentas CASE, dado que o livro tem a pretensão da aplicação do UML no processo de desenvolvimento, no entanto tal não acontece.

Ao contrário das aspirações assumidas no início do livro, o mesmo parece indicado para leitores que tomem contacto pela primeira vez com UML, mas fica aquém das necessidades de alunos do ensino superior que pretendam construir uma carreira na área das tecnologias de informação.

Fernando Martins

Directório de aplicações web 2.0

GO2WEB20.net

THE COMPLETE WEB 2.0 DIRECTORY

Um directório extenso e pesquisável de ferramentas web 2.0, mantido pela TechCrunch. Aqui é possível encontrar as mais variadas aplicações, de uma forma bastante intuitiva.

http://www.go2web20.net

Digg BigSpy

Dos laboratórios do popular portal de notícias Digg foi criada uma nova e muito apelativa forma de observar a actividade de todos os utilizadores. Assim, é possível visualizar em tempo real, todas as notícias votadas pelos membros, enquanto as notícias com mais importância (ou simplesmente mais votadas) são realçadas.

http://labs.digg.com/bigspy/

Video-aulas de PHP

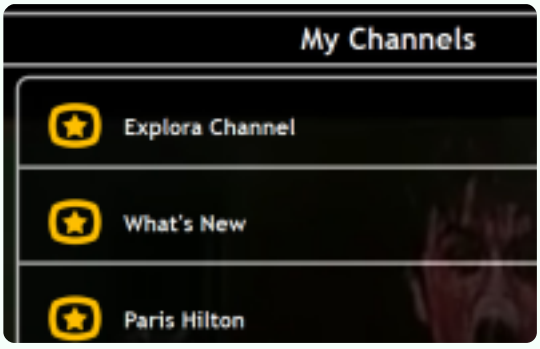
- PHP Day 1: Getting Started – Five to make a PHP file with Notepad
- PHP Day 2: Strings and Variables
- PHP Day 3: Numbers and Operat

Blog mantido por uma programadora, com um tutorial de PHP. Cada aula corresponde a um vídeo flash, sobre um assunto particular desta popular linguagem de programação.

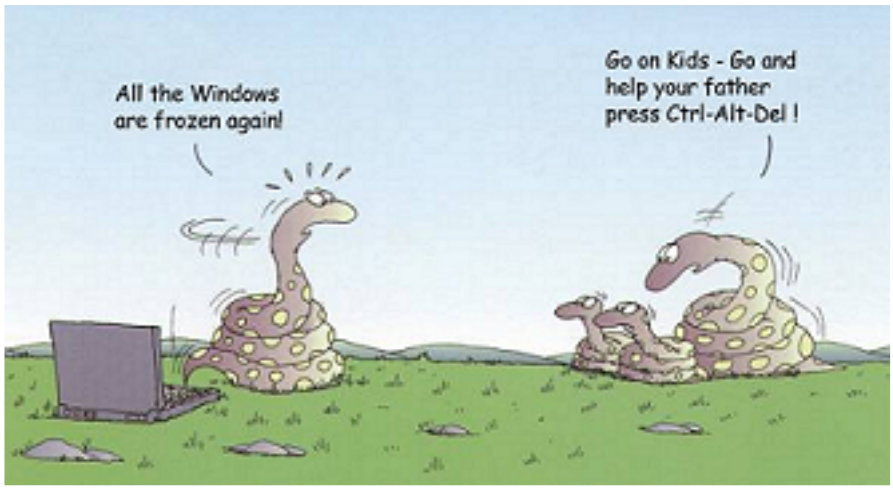
http://programming.learningnerd.com/php-blog

Joost

Nova aplicação *peer-to-peer* para o streaming de canais de TV. Uma nova forma de aproveitar o melhor dos dois mundos: TV e internet, através de uma aplicação fácil de usar e onde podemos verdadeiramente escolher o que queremos ver.



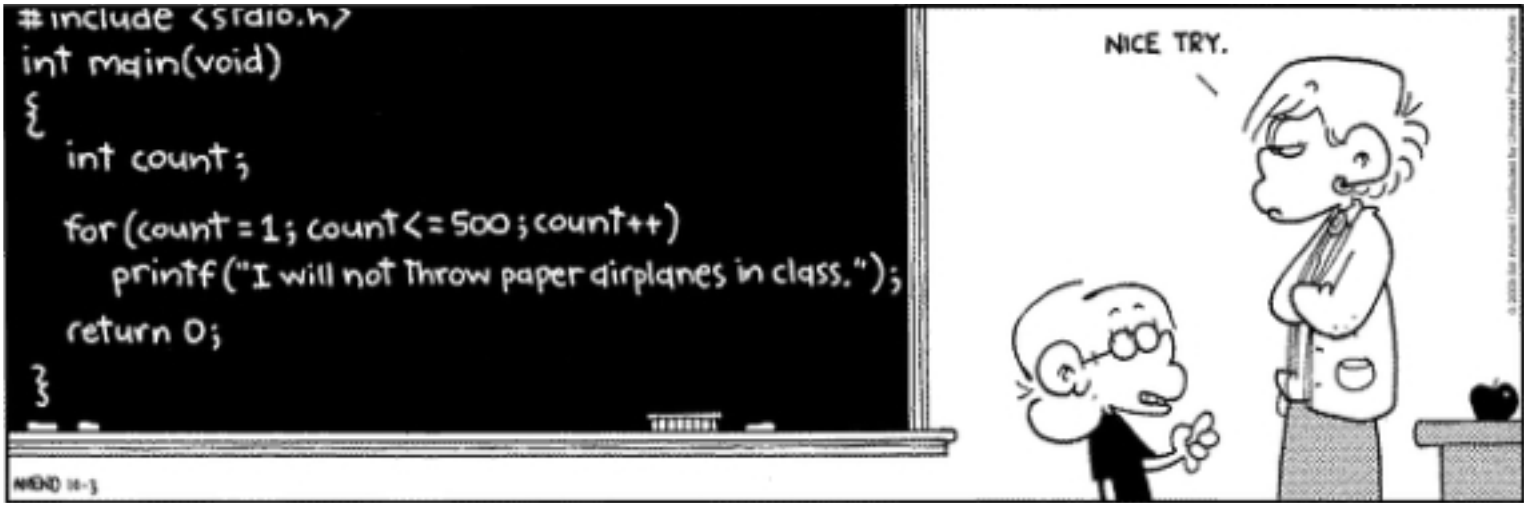
http://www.joost.com



Problemas de teclado

Leve o seu Windows/Mac consigo

Solução desesperada



As crianças estão cada vez mais inteligentes...

Estamos no último mês do primeiro trimestre do ano e sinto-me optimista em relação ao rumo que as coisas estão a tomar na comunidade. Vejo o número de utilizadores activos a aumentar no fórum, assim como o número de utilizadores no seu total, vejo várias equipas de desenvolvimento a serem formadas a partir do fórum, vários projectos a surgir, várias dezenas de pessoas a querer contribuír activamente para o crescimento da comunidade, enfim, vejo vontade de trabalhar. Por vezes torna-se difícil para o staff apoiar todas as iniciativas, todos os projectos e dar iniciação a todas as ideias que surgem. No entanto, com algum esforço e vontade as coisas acabam por se fazer com relativa rapidez e sucesso.

O wiki (http://wiki.portugal-a-programar.org) é mais uma das plataformas de apoio à formação desses projectos, de apoio à escrita de artigos na nossa língua-mãe e de apoio a quem quer aprender algo sobre programação. O próximo passo será dado também no sentido de continuar a promover a construção de conteúdos e aprendizagem de quem se disponibilizar para tal.

Enfim, concluo o "A Comunidade" desta edição de sorriso estampado na cara pelo facto do Portugal-a-Programar continuar a fazer sucesso por este país fora, muito devido ao trabalho e colaboração de todos os moderadores do fórum, administradores dos pólos e aos utilizadores que têm demonstrado vontade e capacidade de trabalho. Espero que o segundo trimestre do ano traga mais prosperidade e que os resultados do trabalho da comunidade agradem a todos os programadores nacionais.

Queres participar na revista PROGRAMAR? Queres integrar este projecto, escrever artigos e ajudar a tornar esta revista num marco da programação nacional?

Vai a

www.revista-programar.info

para mais informação como participar
ou então contacta-nos por

revistaprogramar
@portugal-a-programar.org

Precisamos do apoio de todos para tornar este projecto ainda maior...

contamos com a tua ajuda

Rui Maia