# PROGRAMAR

A REVISTA PORTUGUESA DE PROGRAMAÇÃO

Revista nº4 - Setembro de 2006

www.portugal-a-programar.org

Electrónica:

## Microcontroladores

Saiba como programar microcontroladores

## JavaScript

Inicie-se numa das linguagens mais usadas na web

## Orion's Belt

Conheça este jogo *web-based* português

## Opera Mini

Análise a este novo browser móvel


NOTÍCIAS: SHIFT: Social and Human Ideas For Technology - Intel Core 2 Duo Apresenta 67 Bugs

## índice



## equipa PROGRAMAR



**contacto**
revistaprogramar
@portugal-a-programar.org

**website**
www.revista-programar.info

# Evolução

*Este projecto continua em evolução e nesta edição isso será mais visível que nunca. Uma mudança de visual foi algo que nos era pedido nos comentários sobre as anteriores edições. Para auxiliar esta mudança, usámos um programa propriamente destinado a publicações, o Scribus. Também é open-source e, embora possua ainda alguns defeitos, como na edição de texto, possibilitou-nos a revolução gráfica desta edição, para melhor ou pior depois saberemos, mas gostámos dos resultados.*

*Entretanto, o nosso projecto continua a crescer, tanto em número de participantes como em contribuições. Numa revista deste tipo, que vive das contribuições dos membros e da variedade destas, é sempre importante manter um grupo activo de redactores e outro grupo de membros que participam ocasionalmente. Este objectivo tem sido cumprido até agora, resultando numa publicação sempre composta por conteúdos variados, requesito necessário numa área tão vasta como é a da programação de computadores.*

*Esperamos que as alterações efectuadas sejam do vosso agrado e prometemos continuar a prosseguir os nossos objectivos principais: chegar a cada vez mais programadores e atrair cada vez mais gente para o nosso mundo.*

Sérgio Santos

# SHIFT

Vai realizar-se na Universidade Nova de Lisboa, entre os dias 27 e 29 de Setembro de 2006, o Congresso "Shift – Social and Human Ideas for Technology" centrado nas tecnologias emergentes e no seu impacto na vida das pessoas e respectivas comunidades.

Shift é uma conferência sobre o papel e as implicações da tecnologia na sociedade actual, serviços centrados no utilizador, movimentos sociais que criam novas relações sociais e tecnologia que melhora a nossa vida diária. Tem como público-alvo, europeus entusiastas da tecnologia, com principal incidência para o público português e espanhol.


Temas propostos para este ano:

- A relação entre Pessoas e Tecnologia
- Gestão do Conhecimento
- Os Blogs e as novas formas de participação cívica
- Liberdades e Privacidade nos meios digitais

Mais informações:
www.shift.pt

## Bugs encontrados nos novos processadores da Intel

Num documento disponibilizado há poucos dias, a Intel informou os utilizadores que qualquer um dos processadores Intel Core 2 Duo recém chegados, possuem mais de 60 bugs identificados.

Apenas 20 dos 60 bugs terão correcção a curto-prazo, o que garante que nenhum dos outros, segundo a Intel, prejudicará o utilizador em termos de utilização do sistema, à excepção do AI39, que pode corrompê-lo.

## Google aproxima-se de novo mercado

Com o recente lançamento do novo serviço Google, que possibilita utilizar o domínio do utilizador como domínio de e-mail nas contas Gmail, a empresa norte-americana aproxima as suas soluções de um pacote de produtividade Office. Recebendo ainda grande parte dos seus lucros de serviços publicitários, a Google já lançou diversos serviços, como folhas de cálculo, calendários, conversação, que possibilitam soluções simples para individuais e pequenas empresas.

Embora a empresa ainda não tenha declarado as suas intenções, cada vez mais surge a ideia que a Google pretende atingir uma nova área de negócio, desta vez com soluções totalmente baseadas em plataformas web. Resta-nos esperar pelo que poderá surgir futuramente e verificar se se tornará numa opção viável neste importante mercado.

# Microcontroladores

Este artigo tem como objectivos a apresentação de Microcontroladores e a dependência que hoje em dia a electrónica no geral (Robótica, Domótica, Electrónica Industrial/Residencial, Telecomunicações...) tem dos mesmos.

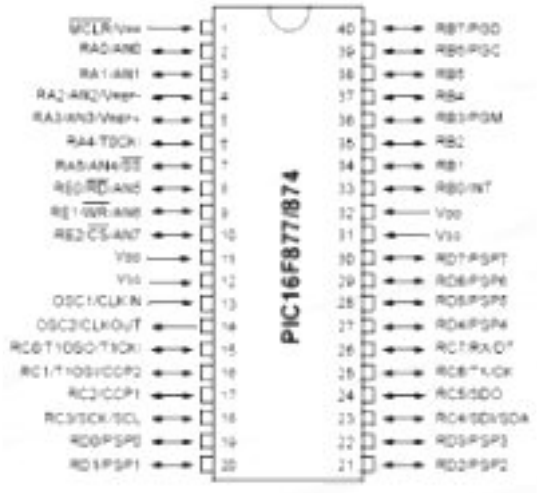
Os microcontroladores são unidades que podem ser programas em linguagens de alto nível, como por exemplo C. Existem linguagens de mais baixo nível (assembly), embora seja mais difícil de descrever um algoritmo, do ponto de vista do projectista do circuito. Esta programação confere aos circuitos digitais constituídos por, microcontroladores/microprocessadores, uma autonomia e inteligência, que de outra maneira seria muito difícil. Graças a esta "inteligência" começaram a ser usados em circuitos que requerem alguma autonomia e capacidade de decisão mediante factores externos. Estas enormes vantagens originaram o aparecimento de circuitos "inteligentes" e capazes de processar e relacionar a mais variada informação.

Num simples automóvel, a centralina (microcontroladores em paralelo) é capaz de ao mesmo tempo controlar: a combustão, a energia consumida, o ABS, o ESP e as mais variadas funções. O uso de microcontroladores num Robô confere-lhe a capacidade para se desviar de objectos, reconhecer a voz e outros sons, fazer movimentos que visem um objectivo. Um microcontrolador pode ser usado em aplicações médicas, para ler sinais eléctricos provenientes de sensores (ECG, TAC...) e mediante o processamento desse sinal (transformadas Z, transformadas Fourier, filtros...) gerar dados que transmitam informação útil aos Profissionais Médicos.

Um Microcontrolador é um circuito integrado, em que num único dispositivo contém alguns dos principais componentes necessários à realização de Sistemas Digitais Programáveis (memória e o mais variado tipo de periféricos). Por sua vez os Microprocessadores, são fundamentalmente constituídos por uma ALU (arithmetic and logic unit), registos de dados, TCU (time and control unit) e necessitam que a memória e os periféricos sejam adicionados externamente, comunicando com o CPU à custa de três barramentos (data, address e control).

A vantagem de poder ter um grande número de periféricos disponíveis dentro de um chip, torna os microcontroladores muito atractivos em projectos electrónicos.

Neste exemplo será usado o PIC 16F877 que é um Microcontrolador desenvolvido pela Microchip Techonology.




Este Microcontrolador tem disponíveis:
- Timers/Counter com Prescaler
- Módulos de comparação, Pwm, captura
- ADC de 10 bits
- Synchronous Serial Port (SSP) com SPI e I2C
- Usart
- Portas I/O
- ...

Quando se compra um PIC, o conteúdo da sua memória EPROMS vem vazia. Será necessário recorrer a um "Programador" (ex: PicSTART da Microchip...), afim de enviar para a EPROM o código do boot-loader. O boot-loader é o programa que corre imediatamente após o "reset" do PIC. Quando corre, vai testar a linha série e verificar se está algum programa "do outro lado da linha" a tentar contactar com ele. Caso exista comunicação pela linha série/usb com o Programador então esse código será escrito na memória EPROM. Este tipo de software (MPLAB entre outros) é disponibilizado pelos fabricantes a troco de alguns euros tal como os programadores, mas existem alternativas tanto para programadores como para o software, nomeadamente o PICP.

A memória do PIC irá estar dividida. Duas dessas divisões dizem respeito à memória onde está o boot-loader e a outra onde irá ser armazenada o código responsável pelo funcionamento do PIC tal como dados que sejam guardados durante o mesmo.

## DEVICE-DRIVERS

Neste artigo serão abordados os principais device drivers para um PIC. Estes device-drivers serão a base para futuros projectos, desde os mais simples até aos mais complexos. Por exemplo, no controlo de temperatura de um processo térmico, os device-drives terão que:
- Fazer leitura da temperatura indicada pelo sensor (termopar)
- Ligar e desligar a resistência de aquecimento
- Ler e escrever de interfaces com o utilizador (ex: teclado do PC, display...)

Assumindo que o PIC já tem devidamente o boot-loader configurado, passa-se ao desenvolvimento do Software que será desenvolvido em C e posteriormente compilado (MPLAB...) no formato Intel-Hex.

Para uma melhor compreensão do código desenvolvido é aconselhável consultar o Datasheet do PIC em causa.

Mais uma vez para simplificar o raciocínio vamos assumir que se estão a realizar device-drivers para o controlo de um processo térmico. Assim a temperatura será dada por um Termopar. Os Termopares necessitam de uma compensação da sua junção fria, aqui irá-se assumir que ela é feita por hardware e que à entrada do PIC chega um sinal analógico com valores entre 0 e 5 V.
Para controlar a resistência irá-se recorrer a um PWM (Pulse width modulation) com duty-cycle variável e sincronizado com a rede.
Para simplificar a leitura, não se irá fazer referência a todos os bits dos registos de configuração, para tal sugere-se a consulta do datasheet afim de verificar o significado dos bits de cada registo em causa.

```
#DEVICE DRIVERS - PAP

#include <pic.h>
#include <stdlib.h>
#include "sci.h"
//////////CONSTANTES/////////
#define TermCh 0 //canal termopar
#define MEDIA 6 //N de leituras da ADC
#define T1_10MS 0xC350 //t1 value at
10ms
```

Todos os registos e portos do PIC são tratados como variáveis, funcionalidade permitida pelo ficheiro pic.h, pois este indica ao compilador quais os registos do PIC e em que endereço e bancos de memória estão.

O ficheiro sci.h é disponibilizado pela Microchip e é onde estão descritos os protótipos das funções usadas nas comunicações pela porta série (RS-232).

A primeira tarefa a realizar será configurar os portos I/O (PORT_X e TRIS_X), especificando se são portos de saída de dados ou de entrada. Os portos usados como

entrada das ADC, interrupts devem ser configurados como entradas, ou seja, os bits dos registos devem ficar com o valor de 1.

A microchip disponibiliza um conjunto de funções para uso e inicialização da USART.

```
// Function Prototypes:

void Sci_Init();
void Sci_PutChar(unsigned char);
unsigned char Sci_GetChar(void);
unsigned char Sci_GetFERR(void);
unsigned char Sci_CheckOERR(void);
void Sci_PutStr(const char* );
void Sci_PutNum (unsigned int num,
unsigned char n, unsigned char comma);
```

Estas funções podem ser usadas em qualquer parte do código, à excepção da Sci_Init, função para responsável pela inicialização da USART. Esta função tem que ser invocada antes de qualquer uma das outras, visto todas as outras fazerem uso da USART.

```
void Sci_Init()   // 19.2kbps
{
BRGH = 1;  /*high baud rate*/
SPBRG = 64;  /*set the baud rate*/
SYNC = 0;  /*asynchronous*/
   SPEN = 1;  /*enable serial port
pins*/
CREN = 1;  /*enable reception*/
SREN = 0;  /*no effect*/
TXIE = 0;  /*disable tx interrupts*/
RCIE = 0;  /*disable rx interrupts*/
TX9  = 0;  /*8-bit transmission*/
RX9  = 0;  /*8-bit reception*/
TRISC7 = 1;  /*RX*/
TRISC6 = 0;  /*TX*/
TXEN = 1;  /*enable the transmitter*/
}
```

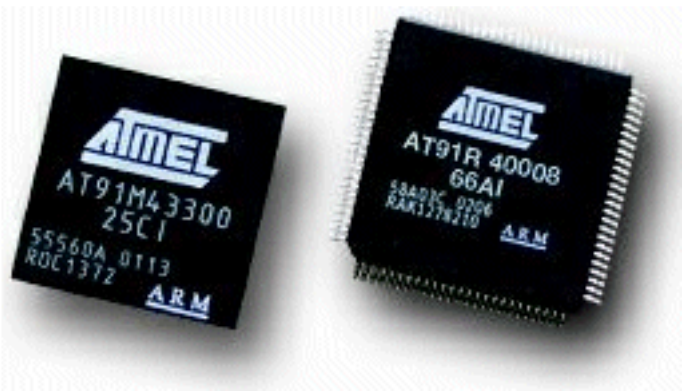
Uma das partes mais importantes é a obtenção de dados externos, provenientes de sensores (temperatura, pressão, humidade, alongamentos, posição, piezoeléctricos, químicos ...). A aplicação dos sensores é tão vasta que vai desde a medicina até à indústria automóvel. Os grandes avanços nestas áreas devem-se à constante evolução e diversificação dos sensores. Mas a maioria deles produz sinais de amplitudes muito baixas. Será necessário o seu correcto acondicionamento para que estes possam ser processados e usados por Microprocessadores. É nesta fase que entram as ADCs (Analogic to Digital Converter) pois convertem sinais analógicos em sinais digitais. Os PICs têm ADCs inbutidas, o que facilita em muito o projecto de hardware (custo, espaço, consumo potência ...)

```
void adc_read(unsigned char channel)
// Lê uma entrada analóogica
{
   ADCON0 = (channel << 3) + 0x81;
   //Enable ADC No canal Cannel
   //Acquisition time = 19.72uS

   delay_uS(22); /*Espera 22us
(definida em delay.h) para carregar
completamente Chold e depende da
impedância da fonte do sinal.*/

   ADGO = 1;
}
```

Para diminuir os erros inerentes à medição, normalmente faz-se uma média de consecutivas leituras da ADC. Supondo que o sinal proveniente do termopar já está devidamente acondicionado e dimensionado então uma possível função para aquisição de dados da ADC será:



```
void read_term(void)
{
   unsigned char i;

   adc_val=0;
   ADCON1=0x00; //left justified


   for(i=0;i<MEDIA;i++)
   {
      adc_read(TermCh);
      while(ADGO); //espera pela
conclusão da conversão
      adc_val +=ADRESH; // formata o
resultado
   }
   adc_val/=MEDIA; //return value
}
```

No caso do PIC 16f877 existem 8 canais para ADC (AN0..AN7). A ADC usada neste caso é de 8 bits = 2^8 =256. Supondo que se tem um sinal analógico, à entrada da ADC, variável entre 0 e 5 volts, garantidos pelo acondicionamento do sinal, então para 0V tem-se adc_val=0 e para 5V tem-se adc_val=255. Estes valores terão que ser tratados, se for necessário, para que tenham uma correspondência com a grandeza medida.

No caso de se estar a actuar sobre uma resistência, então o PWM terá que estar sincronizado com a tensão de rede. Este PWM terá um duty-cycle variável e proporcional ao tempo de actuação da resistência. A técnica do PWM é muito poderosa para controlar circuitos analógicos (motores...) através de circuitos digitais (PIC...).

Para que exista sincronização do PWM com a rede (a menos de uns nanossegundos de diferença) é necessário que "o PIC" saiba quando é que existe uma passagem por zero da tensão de rede.

Assim, é necessário um detector de passagens por zero, circuito esse implementado à custa de um simples transístor. Este evento pode ser idêntico a muitos outros, tais como, o enchimento de uma fila de espera de um buffer de um router, controlo de periféricos num PC ...

O controlo do duty cycle do PWM será variável (0.01% a 99,99%) e dependente da temperatura (neste caso). O valor do duty cycle estará dependente de um algoritmo de controlo (rampa, PI, PI-D...). Para que se possam usar estes algoritmos é necessário conhecer a dinâmica do sinal (overshoot, tempo subida, tempo estabelecimento...), pois os actuadores têm naturezas físicas distintas e reagem de maneiras diferentes. Assim dentro do programa, com os dados obtidos das conversões da ADC, calcula-se o valor correcto, para que o actuador obedeça aos requisitos.

Então uma possível rotina de atendimento às interrupções, fazendo uso do módulo de PWM existente no PIC, será:

```
…
time = T1_10MS-
dutycycle*(T1_10MS/100); // duty_cycle
em percentagem
…
```

```
void interrupt isr (void)
{
(1) if (INTF)
    {
(2)    TMR1L =0;
(3)    TMR1H =0;
(4)    CCP1CON=0b00000000;
(5)    CCP1CON=0b00001000;
(6)    TMR1ON=1;
(7)    CCPR1H= time>>8;
(8)    CCPR1L= time;
(9)    INTF=0;
    }
(10)else if (CCP1IF)
    {
(11)   TMR1ON=0;
(12)   CCP1IF=0;
    }
}
```

(1) Testa a origem da interrupção. Como a interrupção está a ocorrer no pino INT então a cada evento gerado pelo ZCD (zero cross detector) irá ser gerada uma interrupção.

(2)(3) Inicializa o timer1 a zero.

(4) Necessário para fazer o reset ao pino ccp, pois é a única maneira de colocar o pino a zero após o interrupt.

(5) Configura o timer 1 a funcionar em compare mode e a cada match o pino ccp é activado (5v).

(6) Activa o timer1 (início de contagem).

(7)(8) Valor a ser comparado com o timer 1 ao qual será gerado um match
(9) Limpa o bit que indica a origem do interrupt.

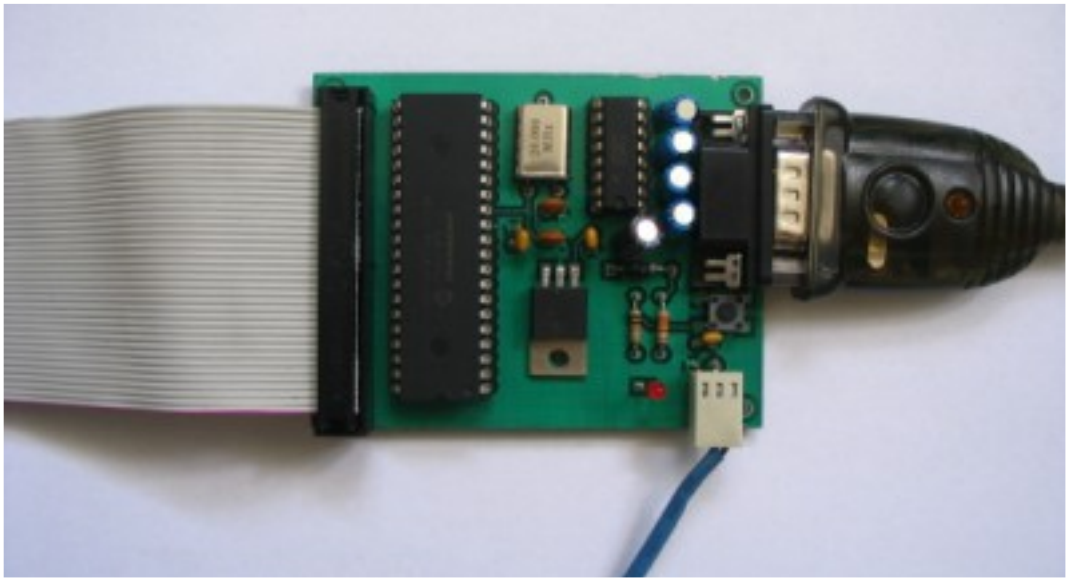
(10)(11)(12) No caso de ausência de um evento no pino INT, evita-se que o ccp esteja indefinidamente activo.

## Conclusão

Tirando os pormenores da interface com o utilizador, estes device-drivers, poderão ser usados em muitas aplicações:

- Para a medida de uma distância a que um robô está de um objecto, pode-se usar ultra-sons. Conhecida a velocidade de propagação (dependente das condições do meio), pode-se calcular a distância, sabendo o tempo que o sinal demorou a atingir o objecto e a voltar (reflexão). Para o cálculo desse tempo pode-se, após o lançamento do ultra-som, iniciar a contagem num timer e quando se receber a reflexão então o detector irá gerar uma interrupção. Conhecido este tempo o robô saberá a distância até ao objecto e tomará as devidas precauções.

- Nos automóveis o segredo da relação performance/economia, está no correcto controlo do combustível usado na combustão. O cálculo da quantidade usada, é dependente de muitos factores, como a concentração de combustível desperdiçado durante a combustão. Esse desperdício poderá ser mensurável à saída do motor. Assim, graças a um sensor químico que meça a quantidade de combustível, poder-se-á optimizar a combustão. Esta optimização será feita pela centralina, que embora seja uma unidade complexa, visto ter que controlar vários mecanismos (ABS, ESP, ...), pode ser vista como vários microprocessadores em paralelo...



Ricardo Rocha

# Classes

**Neste artigo vamos abordar as tão temidas classes de php, bem como os seus construtores, destrutores e herança. Vamos tentar demonstrar de uma forma simples e completa como criar e utilizar as classes em php (versão 5).**

Começamos por criar a nossa class, para isso vamos escrever o seguinte código.

```
1 <?php
2 class User{
3   public $nome = "guest";
4
5   function printNome(){
6     print($this->nome);
7   }
8 }
9 ?>
```

Vamos agora analisar o que foi feito, na linha 2 foi declarada uma class de nome "User", essa class contém uma variável "nome", com o valor "guest" como se pode ver na linha 3, na linha 5 é definida a função printNome() que ao ser chamada irá imprimir o valor da variável "nome". A referência $this é utilizada para dizer que a variável a usar será a variável definida na class, mais á frente vamos ver melhor esta referência.

Agora vejamos como usar a class que criamos, para isso adicionamos o seguinte código fora da class, atenção que o código adicionado não precisa de ser colocado dentro do ficheiro da class, pode e até deve ser colocado noutro ficheiro e importando a class através do include.

```
1  <?php
2  class User{
3    public $nome = "guest";
4    function printNome(){
5      print($this->nome);
6    }
7  }
8  $obj = new User();
9  $obj->printNome();
10 ?>
```

Como podemos ver na linha 8 vai ser criado através do comando new um objecto de nome "obj", este nome pode ser qualquer um tal como se tratasse de uma variável normal. Na linha 9 será pedido ao objecto "obj" que execute o método printNome() definido na class "User", este pedido é feito através da seguinte sintaxe Objecto -> Método.

Agora que já sabemos como criar uma class simples vamos agora ver como funcionam e para que servem os construtores e destrutores de classes.
Para isso vamos usar o seguinte código.

```
1  <?php
2  class User{
3     public $nome;
4     public $email;
5     public $pass;
6
7     function __construct($nome,
$email, $pass){
8           $this->nome = $nome;
9           $this->email = $email;
10          $this->pass = $pass;
11    }
12    function getNome(){
13          return $this->nome;
14    }
15    function printNome(){
16          print($this->nome);
17    }
18    function getEmail(){
19          return $this->email;
20    }
21    function getPass(){
22          return $this->pass;
23    }
24  }
25
26  $obj=new User("guest",
"guest@pap.pt", "123456");
```

```
27  $user = $obj->getNome();
28  $mail = $obj->getMail();
29  $pass = $obj->getPass();
30  }
31  ?>
```

Como podemos ver é definida uma class "User", com as variáveis "nome", "email" e "pass". Foram também definidos os métodos getNome(), getEmail() e getPass() que retorna cada um uma variável da class. Agora a novidade, a função __construct(), linha 7. Esta função deve ser nomeada da forma como se encontra aqui presente.

A função __construct() é executada automaticamente quando um objecto de class é criado, todo o código presente dentro desta função será executado quando for criado o objecto, neste caso o método __construct() foi definido para receber três argumentos "nome","email" e "pass" e atribuir esses valores às variáveis de class definidas anteriormente. Como podemos ver nas linhas 8, 9 e 10 é utilizada a referência $this e o que o $this vai fazer em Português é dizer que a variável nome para a qual ele aponta é a variável de class e não as variáveis que foram dadas como argumentos. Basicamente o $this deve ser utilizado sempre que nos referimos a uma variável, constante ou função definidas na class.

Nas linhas 26 a 29 podemos ver como utilizar esta class, primeiro criamos o objecto com os valores dados ou seja na linha 26 vai ser criado um objecto em que as variáveis "nome", "email" e "pass" vão ter os valores "guest", "guest@pap.pt" e "123456", linhas seguintes vemos como utilizar os métodos para retornar os valores das variáveis de class.

Para além da função __construct() temos também a função __destruct(), que ao contrário da função __construct(), esta é executada quando o objecto é destruído, ou seja quando é "apagado", não vamos entrar em maior detalhe visto ser um método de baixa importância.

Outra parte importante é a herança. Uma classe pode herdar métodos e variáveis de uma outra classe usando a palavra extends na declaração como podemos ver a baixo. Uma classe apenas pode herdar de uma outra classa. Não é possível ter uma classe a herdar de várias classes, mas é possível ter várias classes a herdar uma só classe. Os objectos herdados podem ser redefinidos ou utilizados na sua forma original, através da utilização da clausula parent:: para funções e self:: para variáveis. Um exemplo:

```
<?php
class ClassPrincipal{
   public $nome = "guest";

   function printNome(){
      print($this->nome);
   }
}
class SubClass extends ClassPrincipal{
   public $email = "guest@pap.pt";

   function printAll(){
      print($this->email);
      parent::printNome();
   }
   function printNome(){
      parent::printNome();
   }
}
$obj = new SubClass();
$obj->printAll();
$obj->printNome();
?>
```

Foram definidas duas classes a classe "ClassPrincipal" e a classe "SubClass". A classe "SubClass" tem acesso a todos os objectos da classe "ClassPrincipal". A função printall() utiliza a função printNome() definida em "ClassPrincipal" assim printAll(), para além de imprimir o valor da variável "mail", vai também imprimir o valor da variável "nome" como está definido na "ClassPrincipal".
Podemos também ver um exemplo de sobrecarregamento de funções em que se cria uma função com o mesmo nome do definido na "ClassPrincipal" e que vai executar a printNome() da "ClassPrincipal".

Fábio Correia

# Ruby + GTK 2

## Introdução à criação de GUIs

**Neste artigo iremos dar uma pequena introdução à criação de GUIs, usando Ruby e a biblioteca GTK 2.**

Ruby é uma linguagem de scripting open-source interpretada que, embora pouco usada, possui um grande potencial. Possui algumas semelhanças com Python e Java, sendo orientada a objectos. Ganhou grande notabilidade através do Ruby on Rails, uma framework destinada à criação de plataformas web, baseadas em base de dados. Usaremos GTK 2 para a criação do GUI, biblioteca base do ambiente gráfico Gnome, dos sistemas operativos GNU/Linux. Mas a biblioteca pode ser instalada noutros sistemas, como o Windows. Para instalar esta biblioteca convém seguir as instruções da página oficial: http://ruby-gnome2.sourceforge.jp. Existem vários pacotes para diferentes distribuições GNU/Linux e a possibilidade de compilar através do código-fonte. Escusado será dizer que é necessária a instalação prévia de Ruby, já incluída em algumas distribuições.

Primeiro iremos criar apenas uma janela, com todas as opções normais, como redimensionar minimizar, maximizar, em que apenas fechar a janela não é possível, sem finalizar o processo.

```
require 'gtk2'

Gtk.init
window = Gtk::Window.new(
           Gtk::Window::TOPLEVEL )
window.show
Gtk.main
```

Basta copiar o código acima, guardar o ficheiro como gtk.rb, por exemplo, e escrever na linha de comandos 'ruby gtk.rb'. Se tudo correr bem, poderão ver uma janela de 200x200 de tamanho. Agora a explicação detalhada.


```
require 'gtk2'
```

Esta linha irá carregar a biblioteca ruby-GTK2, para podermos usar os seus métodos.

```
Gtk.init
```

Este comando irá inicializar a API, de modo a prepará-la para ser usada. A partir de agora podemos começar a criar os objectos que irão compor o GUI.

```
window = Gtk::Window.new(
           Gtk::Window::TOPLEVEL )
window.show
```

Aqui é criada a janela principal, sob o nome de window. O argumento passado à função, Gtk::Window::TOPLEVEL, indica que a janela criada irá usar as características normais das janelas do ambiente gráfico, bordas, decorações, ... O comando seguinte irá mostrar a janela no ecrã.

```
Gtk.main
```

Esta chamada irá indicará que o programa irá esperar agora e recolher todos os eventos que ocorrerem, como o carregar de um botão, o arrastar da janela, ...

No exemplo seguinte, iremos já criar um botão, adicioná-lo à nossa janela e executar um comando cada vez que este for pressionado. O comando será escrever "Olá!" na linha de comandos. O programa passará também a encerrar sem problemas.

```
require 'gtk2'

def delete_event( widget )
      return false
end

def destroy( widget )
      Gtk.main_quit
end

Gtk.init

window = Gtk::Window.new(
            Gtk::Window::TOPLEVEL )
window.set_default_size( 200, 200 )

window.signal_connect(
            "delete_event" ) do
   delete_event( window )
end

window.signal_connect( "destroy" ) do
   destroy( window )
end

button = Gtk::Button.new( "Olá!" )
window.add( button )

button.signal_connect( "clicked" ) do
      puts "Olá!"
end

button.show
window.show

Gtk.main
```

Atribuam um nome ao programa e corram-no. O resultado deverá ser semelhante a este:


```
def delete_event( widget )
      return false
end

def destroy( widget )
      Gtk.main_quit
end
```

Estas duas funções irão permitir o encerramento e destruição da janela. Apenas necessitam de saber que a função delete_event deverá retornar false, de modo a proceder-se à destruição da janela. Esta função poderá ser usada para evitar que a janela feche logo, e perguntar ao utilizador se deseja mesmo sair.

```
window.set_default_size( 200, 200 )
```

Este comando irá definir o tamanho da janela para 200 por 200 pixels. Este comando é necessário desta vez porque, sem este, a janela iria tomar o tamanho mínimo do botão que iremos adicionar.

```
window.signal_connect(
            "delete_event" ) do
      delete_event( window )
end

window.signal_connect( "destroy" ) do
      destroy( window )
end
```

A função signal_connect permite capturar um evento que tenha ocorrido em algum objecto. Neste caso o objectivo é capturar os eventos de fecho e destruição da janela e chamar as funções anteriormente criadas.

```
button = Gtk::Button.new( "Olá!" )
window.add( button )
```

Criamos um objecto do tipo Button, com a etiqueta de Olá!. O comando a seguir adiciona o botão criado à nossa janela.

```
button.signal_connect( "clicked" ) do
      puts "Olá!"
end
```

Usamos novamente a função signal_connect, mas desta vez com o nosso botão, para verificarmos quando este é pressionado. Sempre que for pressionado, o comando puts irá escrever na linha de comandos Olá!.

```
button.show
window.show

Gtk.main
```

Finalmente, mostramos os dois objectos criados no ecrã e esperamos para que ocorram os eventos esperados.

Para terminar, neste último exemplo irá demonstrar o uso de caixas para conter e organizar diversos objectos e o uso de um novo tipo de objecto, o Label.

```
require 'gtk2'

def delete_event( widget )
      return false
end

def destroy( widget )
      Gtk.main_quit
end

Gtk.init

window = Gtk::Window.new(
            Gtk::Window::TOPLEVEL )
window.set_default_size( 200, 200 )

window.title = "Hello World"

window.border_width = 5

window.signal_connect( "delete_event"
) do
   delete_event( window )
end

window.signal_connect( "destroy" ) do
   destroy( window )
end

box = Gtk::VBox.new( false, 0 )

window.add( box )

button = Gtk::Button.new(
"Cumprimentar" )

text = Gtk::Label.new( "" )

button.signal_connect( "clicked" ) do
      text.label = "Olá!"
end

box.pack_start( button, true, true, 5 )
box.pack_start( text, true, true, 5 )

button.show
text.show
box.show
window.show

Gtk.main
```

Desta vez definimos mais algumas propriedades da janela, que são úteis em muitos casos. O resultado deverá ser semelhante a este:


```
window.title = "Hello World"

window.border_width = 5
```

O primeiro comando define o nome da janela, que irá aparecer na barra de título. O segundo comando define uma borda interna de 5 pixeis, ou seja, a distância a que os objectos se deverão encontrar da borda exterior da janela.

```
box = Gtk::VBox.new( false, 0 )

window.add( box )
```

Criamos um objecto do tipo VBox, ou seja, uma caixa vertical. Este tipo de caixas organiza os objectos que contêm verticalmente, enquanto as HBox o fazem horizontalmente. O primeiro argumento define se o espaço que cada objecto irá ocupar deverá ser homogéneo ou não, com os restantes objectos da caixa. O segundo argumento define o espaçamento que os objectos irão ter entre si, em pixels. O segundo comando adiciona a caixa à janela. Assim a janela poderá conter objectos dentro da caixa.

```
text = Gtk::Label.new( "" )

button.signal_connect( "clicked" ) do
      text.label = "Olá!"
end
```

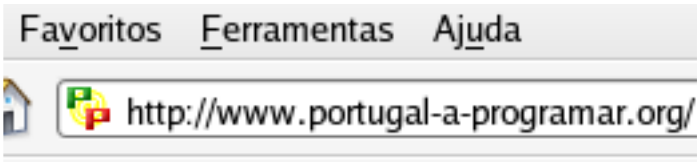
Criamos o objecto de tipo Label, sem nada escrito, como podem ver pelo argumento fornecido. Depois alteramos o comando que ocorre quando o botão é premido, para escrever algo na etiqueta que criámos: Olá!.

```
box.pack_start( button, true, true, 5 )
box.pack_start( text, true, true, 5 )
```

Agora adicionamos o botão e a etiqueta à caixa, usando a função pack_start. Esta função vai adicionando os objectos um a seguir ao outro, preenchendo a caixa, ao contrário de pack_end, que faz o oposto. O primeiro argumento indica o objecto a adicionar. O segundo argumento indica se os objectos deverão expandir-se, de modo a que a caixa ocupe o máximo de espaço possível. O terceiro argumento funciona se o segundo for verdadeiro, e define se o espaço ganho pela expansão da caixa deverá ser ocupado pelos objectos, ou apenas por espaçamento. O quarto e último argumento define o espaçamento que o objecto deverá ter, em pixels, à volta dele, dentro da caixa.

E aqui termina esta introdução à criação de GUIs usando o GTK 2. Se ficaram interessados pelos exemplos que viram, poderão ir ao site oficial (http://ruby-gnome2.sourceforge.jp) e procurar mais informações sobre diferentes objectos e os seus atributos. Esta é, sem dúvida, uma das melhores e mais simples biblioteca para a criação de GUIs.


Sérgio Santos




# JavaScript

## Introdução

JavaScript é uma linguagem muito simples e de fácil aprendizagem. Possui uma sintaxe (forma de escrita) semelhante ao Java, C, C++, PHP. Foi criada para ter uma utilização web based por Brendan Eich da Netscape em 1995, como uma extensão para o browser Navigator v2.0, com o principal propósito de permitir uma interactividade superior à que se conseguia com o HTML e para atender principalmente às seguintes necessidades:
- Validação de formulários em client side;
- Interação com a página. Assim, foi feita como uma linguagem de script.

Existem várias implementações de JavaScript, com ligeiras diferenças entre si: o standard definido pela ECMA, a implementação da Netscape e a implementação da Microsoft.

A principal característica do JavaScript é a forma como interage com o browser, podendo aceder a objectos. Assim, para uma boa programação em JavaScript é necessário conhecer o DOM (Document Object Model) de cada browser. Devido aos browsers terem DOM diferentes é necessário conhecê-los especificamente, daí vem as diversas implementações do JavaScript.

Á sua união com o CSS chama-se de DHTML, assim usando o JavaScript é possível modificar dinamicamente os estilos das páginas. A linguagem ActionScript da Macromedia baseia-se no standard do JavaScript criado pela ECMA.

Os recursos básicos para quem quer programar em JavaScript são um editor de texto simples para escrever o código e um browser para o testar.

O JavaScript é uma linguagem embebida no seio do HTML, é orientada a objectos e a eventos que acede do DOM do browser. É uma linguagem independente da plataforma onde corre. É uma linguagem que não é compilada e é interpretada pelo browser, baseando-se em objectos.

### Potencialidades do JavaScript a ter em conta:
- gerar código HTML;
- reagir conforme as acções do visitante;
- gerar conteúdos dinâmicos;
- validar formulários;
- criar efeitos gráficos.

### Vantagens da utilização do JavaScript:
- fácil aprendizagem;
- não exige recursos server side;
- é rápida por ser interpretada no browser;
- existem muitos recursos na Internet.

### Desvantagens da utilização do JavaScript:
- o código pode ser facilmente copiado pois fica exposto;
- não é uma boa ferramenta para interagir com base de dados;
- é menos versátil que outras linguagens genéricas.

## Estrutura dos programas em JavaScript

O JavaScript é normalmente inserido em ficheiros HTML e é colocado dentro das marcas <script> </script>.
O código tem de estar ou entre as tags <head> </head> ou entre as tags <body> </body>.

## Questões de Sintaxe

O JavaScript é sensível a maiúsculas e minúsculas, desta forma escrever 'Nome' e 'nome' não é a mesma coisa e vão ser lidas como independentes uma da outra.
Os espaços não preenchidos são ignorados, por isso deixar espaçamento é apenas uma questão de entendimento pois o JavaScript não vai notar diferenças.
Todas as instruções devem terminar com um ponto e vírgula (;).

É aconselhável desde o início a colocar comentários no código para o tornar de mais fácil leitura e edição. Para inserir comentários pode escrever entre os delimitadores '/*' e '*/' ou então '//' e escrever até ao fim da linha, tenha em atenção que os comentários vão ser ignorados pelo JavaScript, por isso não convém colocar código dentro de um comentário pois este não será lido.

Se pretende que o script que estiver a desenvolver em JavaScript só seja apresentado na página caso o browser em questão esteja bem configurado e suporte JavaScript coloque o seu código dentro de '<!–' e '–>', como mostra o exemplo:

```
<html>
<head>
<title>Inserir código seguro</title>
</head>
<body>
<script type="text/javascript">
<!--
   document.write("bla bla bla")
   //escreve bla bla bla na página
-->
</script>
</body>
</html>
```

## Ficheiros Externos

Quando é pretendido utilizar um código por várias páginas ou, por uma questão de organização, é possível e conveniente colocá-lo num ficheiro externo que deve seguir as seguintes regras:
1 - a extensão do ficheiro tem de ser .js
2 - deve ser invocado a partir de um ficheiro HTML com a seguinte forma <script src="oTalExterno.js"></script>
3 - dentro do ficheiro externo não se deve incluir a tag <script> nem a tag </script>
Desta forma um ficheiro externo deve ter esta formatação:

```
document.write("bla bla bla")
```

## Tipos de dados

Existem várias possibilidades de tipos de dados em JavaScript entre eles destacam-se os numéricos, os lógicos, as strings, o null e o undefined. Os numéricos como o nome indica podem armazenar valores inteiros e com ponto flutuante e podem fazer parte de operações de aritemética como a soma, a subtracção, a multiplicação e a divisão. Os valores lógicos apenas podem armazenar dois valores o true (verdadeiro) e o false (falso). Os valores strings são cadeias de caracteres, o maior número que uma string pode conter depende do browser em questãoe os valores strings são delimitados por apóstrofe(') ou por aspas("). Valores null são valores especiais, representa um objecto nulo, não deve ser confundido com uma variável não inicializada, pois o valor null existe e existe em memória. O valor undefined significa que a variável não foi criada, inicialmente todas as variáveis se encontram neste estado.

## Modelo de Objectos do Documento

O DOM (Document Object Model) de uma página é uma representação hierárquica, em formato de árvore. Desta forma, a raiz é a janela do browser, daí ramifica-se em parágrafos, tabelas, formulários, entre outros.
O JavaScript permite aceder de uma forma dinâmica aos vários ramos, lendo-os e, se necessário, alterando os seus valores.

Existem vários standards dos DOM, o da W3C, o da Netscape e o do Internet Explorer. O DOM do IE implementa as recomendações do W3C mas é mais vasto que este.

## Eventos

O JavaScript é uma linguagem orientada principalmente a eventos e objectos. Os eventos podem ser produzidos de duas formas: pelo sistema ou pelo utilizador. Pelo sistema quando resultam do facto de carregar ou descarregar uma página, pelo utilizador na medida em que pode interagir com as suas acções, através de sobrepor o rato ou carregar num botão, estes são os exemplos mais comuns apesar de existirem muitos mais.

## Eventos do Sistema

Um exemplo de resposta a eventos do sistema:

```
<html><head>
<script language="JavaScript">
<!--
   function entra(){
     window.alert("Bem vindo");
   }
   function sai(){
     window.alert("Volte sempre");
   }
-->
</script>
</head>
<body onload="entra();"
      onunload="sai();">
<p>Texto da página...</p>
</body></head>
```

No <head> definem-se duas instruções, a "entra()" e a "sai()". Na primeira função (function), "entra()", faz surgir uma janela de aviso no centro do ecrã quando a página é carregada que mostra uma mensagem, "Bem vindo, exemplo de uma reaccao a um evento do sistema". Na segunda função (function), "sai()", faz surgir uma janela de aviso quando se sai da página com a mensagem "Volte sempre".

As funções só são executadas no momento em que estas são invocadas, por isso as funções presentes no exemplo 3 só são executadas no onload do body e no onunload do body respecivamente.

## Eventos de utilizador

Os eventos podem ser utilizados para interagir com o utilizador, desta forma só são invocados por acção deste, assim é possível tornar a página mais atractiva e modelada para cada utilizador pois não apresenta tudo mas só o que o utilizador pretende.

```
<head>

<script language="Javascript">
function skin1()
{
   document.bgColor='red';
}
function skin2()
{
   document.bgColor='green';
}
function skin3()
{
   document.bgColor='yellow';
}
function skin4()
{
   document.bgColor='#FFFFCC';
}

</script>

</head>

<body>

<form name="cores">
<input type="radio" name="campo"
onclick="skin1();">Fundo vermelho<br>
<input type="radio" name="campo"
onclick="skin2();">Fundo verde<br>
<input type="radio" name="campo"
onclick="skin3();">Fundo amarelo<br>
<input type="radio" name="campo"
onclick="skin4();">Fundo amarelo<br>
</form>

</body>
```

Neste exemplo o visitante da página pode escolher que cor quer dar ao background. No entanto se o utilizador sair da página a sua escolha será perdida. Desta forma é mostrado uma forma de o JavaScript interagir com o visitante.

## Variáveis

Variável é uma posição de memória onde é possível armazenar certas informações. As variáveis são representadas por nomes denominados identificadores que têm uma estrutura definida:

1. Devem iniciar obrigatoriamente por uma letra ou pelo símbolo "$";
2. A partir daí além de letras, "$" pode conter dígitos(0 até 9).

De seguida vai ser mostrado um exemplo de manipulação de variáveis:

```
<html><head></head>
<body>
<script type="text/javascript">
var mensagem="Aqui está a mensagem";
//define a variável mensagem
document.write(mensagem);
document.write(
      "<p>"+mensagem+"</p>");
</script></body></html>
```

## Declaração de Variáveis

É possível declarar uma variável de duas formas mas só iremos abordar uma delas, a mais utilizada, que consiste em utilizar uma palavra reservada "var". Uma variável que não tenha sido inicializada, possui o valor de "undefined", e tenha em atenção que o JavaScript é sensível a maiúsculas e minúsculas, ou seja, letras minúsculas e maiúsculas são diferentes, portanto, undefined e null devem ser escritos sempre em letra minúsculas. Se uma variável é declarada apenas, com o comando "var", o seu conteúdo é "undefined" ou NaN (Not a Number), caso esteja num contexto numérico.

Declarar uma variável é muito simples, como poderá ver no exemplo seguinte.

```
var nome;
var idade;
var pais="Portugal";
```

Como pode observar no exemplo anterior declarar uma variável é bastante simples.

## Tipos de Variáveis

Em JavaScript não é necessário indicar o tipo de variável com a qual se vai trabalhar devido às variáveis em JavaScript poderem armazenar diferentes tipos de dados.

## Conclusão

Nesta primeira parte do artigo de JavaScript foram abordados temas básicos e iniciais da programação com JavaScript. No entanto, não ficamos por aqui pois devido à grande extensão do artigo, este vai ser dividido em duas partes uma delas. A primeira parte é a que acabaram de ler, a segunda parte sairá na próxima edição da revista, na qual vamos abordar operadores, funções, instruções condicionais (if, switch), instruções iterativas (for, while, do-while, continue, brake), programação orientada a objectos (POO) e vectores.

João Pereira


# VisualBasic.NET 3ª Parte

## Arrays

Um array é conhecido como uma lista, no caso de ser uni-dimensional, ou uma matriz no caso de ser bi-dimensional, e é uma das mais importantes estruturas de dados e também uma das mais simples. Um array é constituído por um nome e por um número, denominado de índice. Em VB.NET, a primeira posição do array é sempre no índice 0, ao contrário do que acontecia no VB6, podia-se definir se a posição 0 ou 1 seria a inicial.

- nomeArray(índice)
- nomeArray(0) – Primeira posição
- nomeArray(Ubound(nomeArray)) – Última posição

## Declaração de arrays

### - Uni-dimensional

Dim nomeArray(numero total de elementos – 1) as Tipo de Dados
Ex:

```
Dim nomeArray(3) As String 'Declara
o array com 4 elementos'
```

OU

```
Dim nomeArray() As String =
{"elemento1", "elemento2",
"elemento3", "elemento4"} 'Declara o
array e atribui valores'
```

### - Bi-dimensional

Dim nomeArray(numero total de elementos – 1, numero total de elementos – 1) as Tipo de Dados

Ex:

```
Dim nomeArray(,) As String
```

OU

```
Dim nomeArray(10, 10) As String
'Array com duas dimensões de 10x10'
```

### - Multi-dimensional

Dim nomeArray(numero total de elementos – 1, numero total de elementos – 1, numero total de elementos – 1, etc) as Tipo de Dados

(numero total de elementos – 1) – terá que ser o valor total de elementos menos 1, razão pela qual é indicada como primeira posição, a posição 0. Um array declarado como nomeArray(9) irá suportar 10 posições, ou seja, de 0 a 9.

Quando trabalhamos com arrays, por vezes vemo-nos obrigados a redimensiona-lo, pois não podemos definir uma dimensão fixa para o mesmo. Para fazermos esta operação, utilizamos o ReDim. O ReDim é usado para alterar o tamanho de cada dimensão do array, mas nunca pode alterar o número de dimensões ou tipo de dados do mesmo. O que o ReDim faz é apagar o array actual e criar um novo array com as definições indicadas no ReDim.

```
Dim nomeArray(10, 10) As String
'Array com duas dimensões de 10x10'

ReDim nomeArray(10, 15)
'Redimensiona o array para 15
elementos na 2ª dimensão'
```

Como o array actual é apagado, toda a informação contida é igualmente eliminada. Para podermos manter essa informação, termos de usar juntamente com o ReDim, o Preserve. Um dos inconvenientes na utilização do Preserve, é que apenas poderá ser alterada a ultima dimensão do array, tendo que manter o número de elementos das outras.

```
ReDim Preserve nomeArray(15, 10)
'Redimensiona o array para 15
elementos na 1ª dimensão e mantem a
informação armazenada'
```

Os arrays possuem como qualquer outra classe do VB.NET um conjunto de métodos e propriedades.

## Propriedades

### IsFixedSize
Indica se o array tem um tamanho fixo.

### IsReadOnly
Indica se o array é somente leitura (read-only).

### IsSynchronized
Indica se o acesso ao array é sincronizado (thread-safe).

### Length
Obtém o total de elementos em todas as dimensões do array.

### Rank
Obtém o número de dimensões do array.

### SyncRoot
Obtém um objecto que pode ser usado para sincronizar o acesso ao array.

## Métodos

### Clear
Define o intervalo dos elementos de um array para zero, para false ou para null.

### Clone
Cria uma copia do array.

### Copy
Copia a secção de um array para outro array e realiza o casting e o boxing requerido.

### CopyTo
Copia todos os elementos de array de uma dimensão para outro array.

### CreateInstance
Inicializa uma nova instância da classe array.

### Equals (herdado de Object)
Determina se duas instâncias de objectos são iguais.

### GetLength
Obtém o numero de elementos de uma dimensão especificada de um array.

### GetLowerBound
Obtém o menor índice inferior da dimensão especificada em um array.

### GetType (herdado de Object)
Obtém o tipo da instância actual.

### GetUpperBound (Equivale ao UBound do VB6)
Obtém o índice superior da dimensão especificada num array.

### GetValue
Obtém o valor de um elemento definido no array.

### IndexOf
Retorna o índice da primeira ocorrência de um valor em um array de uma dimensão.

### Initialize
Inicia cada elemento de um array chamando o construtor padrão.

### Reverse
Inverte a ordem dos elementos de um array de uma dimensão.

### SetValue
Define o elemento especificado em um array para um valor definido.

### Sort
Ordena os elementos de um array de uma dimensão.

### ToString (inherited from Object)
Retorna uma string que representa o objecto actual.

(informações retiradas de www.macoratti.net)

## Ficheiros

Para trabalharmos com ficheiros em VB.NET recorremos quase sempre às classes do System.IO, e para isso, temos que fazer o import deste.

```
Import System.IO
```

Para manipularmos um ficheiro, antes necessitamos de criar um canal para esse mesmo ficheiro, e para isso usamos o FileStream. O FileStream permite-nos então a criação desse canal, e poderá ser definido o modo de abertura do ficheiro, ou seja, se irá ser de leitura e/ou escrita.

```
Dim ficheiro as New FileStream
(Nfich,FileMode,FileAccess,FileShare)
```

Para indicarmos o modo de abertura do canal utilizamos o enumerador FileMode. O FileMode é constituido pelos seguintes membros:

### Append

Abre o ficheiro para escrita e posiciona-se no fim do mesmo. Caso não exista o ficheiro então cria um novo.

### Create

Abre o canal para escrita ou leitura e escrita e apaga o conteúdo caso o ficheiro exista, senão cria um novo.

### CreateNew

Cria um novo ficheiro para escrita.Caso exista dá um erro.

### Open

Abre o ficheiro para escrita ou leitura e posiciona-se no inicio do mesmo. Caso não exista dá um erro.

### OpenOrCreate

Abre o ficheiro se ele existir, senão cria-o.

### Truncate

Abre o ficheiro e remove todo o conteudo.

## Ler um ficheiro

Após fazermos o import do namespace IO, temos disponiveis as classes para a manipulação de ficheiros. Para fazer a leitura de um ficheiro iremos recorrer à classe StreamReader. Esta classe possui métodos como o ReadLine (permite-nos ler o ficheiro linha a linha), o ReadToEnd (permite-nos ler o ficheiro do seu inicio até ao fim e carregá-lo numa variavel ou objecto), entre outros.

```
Imports System.IO
Module Module1
  Sub Main()
    Dim strFicheiro As String =
"C:\ficheiro.txt"
    'Abre o ficheiro'
    'Modo = Abertura | Tipo de Acesso
= Leitura | Acesso de outros processos
= Leitura'
    Dim fs As New
FileStream(strFicheiro, FileMode.Open,
FileAccess.Read, FileShare.Read)
    'Carrega o stream no StreamReader'
    Dim sr As New StreamReader(fs)
    'Imprime o conteudo do ficheiro no
ecra'
    Console.Write(sr.ReadToEnd)
    'Liberta o ficheiro'
    sr.Close()
    Console.ReadLine()
  End Sub
End Module
```

## Escrever um ficheiro

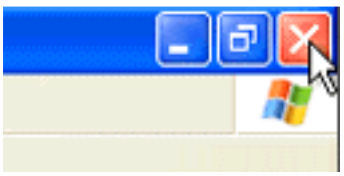
Para escrever num ficheiro utilizamos a classe StreamWriter. Com esta classe podemos escrever num ficheiro usando vários tipos de codificação (UTF-8, ASCII, UNICODE, etc). Os métodos Write e Writeline são os utilizados para a escrita em ficheiros. O Flush e o Close são também métodos importantes da classe StreamWriter. Como os dados estão armazenados em buffers e não directamente no ficheiro, deve-se invocar o método flush para que seja escritos no ficheiro, ou então o método Close para que force a escrita e feche a stream.

```
Imports System.IO
  Module Module1
    Sub Main()
      Dim texto As String
      'chama a funcao para ler o
ficheiro'
      ler_ficheiro()
      Console.WriteLine()
      Console.WriteLine()
      Console.WriteLine("Insira um
texto para adicionar ao ficheiro:")
      texto = Console.ReadLine
      Console.Clear()
      'chama a funcao para escrever
no ficheiro o texto inserido'
      escrever_ficheiro(texto)
      'imprime no ecra o ficheiro ja
actualizado'
      ler_ficheiro()
      Console.ReadLine()
    End Sub

    Private Sub ler_ficheiro()
      Dim strFicheiro As String =
"C:\ficheiro.txt"
      'Abre o ficheiro indicado na
variavel com os atributos'
      'Modo = Abertura | Tipo de
Acesso = Leitura | Acesso de outros
processos = Leitura'
      Dim fs As New
FileStream(strFicheiro,
FileMode.Open, FileAccess.Read,
FileShare.Read)
      'Carrega o stream no
StreamReader'
      Dim sr As New StreamReader(fs)
      'Imprime o conteudo do
ficheiro no ecra'
      Console.Write(sr.ReadToEnd)
      'Liberta o ficheiro'
      sr.Close()
      Console.WriteLine()
    End Sub
```

```
    Private Sub
escrever_ficheiro(ByVal texto As
String)
      Dim strFicheiro As String =
"C:\ficheiro.txt"
      'Abre o ficheiro indicado na
variavel com os atributos'
      'Modo = Adicionar | Tipo de
Acesso = Escrita | Acesso de outros
processos = Leitura'
      Dim fs As New
FileStream(strFicheiro,
FileMode.Append, FileAccess.Write,
FileShare.Read)
      'Carrega o stream no
StreamReader'
      Dim sw As New StreamWriter(fs)
      'Escreve no ficheiro o texto
inserido pelo utilizador'
      sw.WriteLine()
      sw.Write(texto)
      sw.Close()
    End Sub
End Module
```

*Na próxima edição iremos continuar este tutorial. O tema da próxima edição continuará a ser manipulação de ficheiros em VisualBasic.NET. Irão ser abordadas novas classes como Directory, DirectoryInfo, FileInfo, FileAttributes, Path, etc... que vos irão permitir um maior controlo e interacção sobre os vossos ficheiros. Não percam a próxima edição...*



Tiago Salgado & Marcelo Martins

# Segurança em sistemas GNU/Linux

## Introdução

Muita gente pergunta a si mesma porque é que "aqueles tipos dizem que GNU/Linux é mais seguro?". Neste artigo pretendo que pessoas que nunca usaram GNU/Linux e que tenham dúvidas em relação à segurança deste sistema, as esclareçam.

## Permissões

O controlo de segurança mais básico do sistema, mas ainda assim bastante eficaz, é feito ao nível do sistema de permissões. Estas consistem numa estrutura hierárquica, na qual há vários tipos de utilizadores com vários tipos de permissões, podendo estas ser definidas conforme o desejado. Podemos dar permissões para o vizinho do lado ver os meus ficheiros, mas também dar permissões para outro vizinho não ver nada. Isto é muito importante e é esta uma das razões pela qual um vírus não se espalha facilmente em GNU/Linux (tal como em Mac, que tem o mesmo sistema, ambos baseados no Unix).

Para um eventual vírus atacar um destes sistemas com vista a fazer qualquer tipo de estrago, teria que ser o próprio utilizador a executar o vírus, e ainda assim (caso em modo não-root) o único estrago que deveria resultar da execução seria a possível perda dos ficheiros da HOME do utilizador, o único directório no qual um utilizador normal tem permissões totais, visto que lhe pertence, tendo a maioria dos outros directórios permissões restritivas à escrita, e desta forma à alteração ou remoção.

O sistema de permissões pode ser demonstrado numa tabela:

| Dono | | | Grupo | | | Outros | | |
|---|---|---|---|---|---|---|---|---|
| r | w | x | r | w | x | r | w | x |
| 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |

A tabela demonstra exemplifica o sistema de permissões. O 0 significa desligado/permissão não activa, o 1 ligado/permissão activa. O 'r' simboliza a leitura, o 'w' a escrita e o 'x' a execução.

Estas permissões correspondem a três grupos de 3 bits, cada bit correspondendo à leitura, escrita e execução e três grupos, desses tais três bits, referindo-se ao dono (do ficheiro), ao grupo, e aos "outros". Para se perceber melhor o sistema de permissões convém separar cada grupo de três bits individualmente. Desta forma, se quiséssemos ter um ficheiro com permissões totais para o dono do ficheiro, e permissão de leitura e de execução tanto para o Grupo como para os outros teria-mos algo como:

| Dono | | | Grupo | | | Outros | | |
|---|---|---|---|---|---|---|---|---|
| r | w | x | r | w | x | r | w | x |
| 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |

111 101 101

7 em decimal 5 em decimal 5 em decimal

Para que seja mais fácil, apresentamos também uma tabela de conversão de binário para decimal:

| binário | decimal |
|---|---|
| 000 | 0 |
| 001 | 1 |
| 010 | 2 |
| 011 | 3 |
| 100 | 4 |
| 101 | 5 |
| 110 | 6 |
| 111 | 7 |

Portanto, desta forma se quisessemos definir o tal ficheiro com as permissões já referidas, usaríamos o comando chmod, que é o que define as permissões, da seguinte maneira:

```
# chmod 755 ficheiro.txt
```

Como se pode constatar, usou-se o chmod 755, sendo este número a aglutinação da conversão para decimal das permissões demonstradas acima.

Outra maneira, talvez mais simples de exemplificar o processo é dando valores aos tipos de permissão. Deste modo, o r vale 4, o w vale 2 e, por fim, o x vale 1. Se quisermos dar permissões totais teremos r+w+x = 4 + 2 + 1 = 7, sendo este o número que utilizaremos, para o utilizador em questão.

Caso queiramos aplicar a permissão a uma pasta e a todos os seus sub-ficheiros, utilizaremos a extensão -R. Por exemplo:

```
# chmod -R 777 ~/
```

### Notas:

- O ~/ refere-se à home do utilizador.
- Todos os comando têm de ser executados "as root", caso os ficheiros aos quais se vai editar a tabela de permissões não tenham permissões totais para o user em questão.

## A firewall

Uma firewall GNU/Linux visa proteger o computador contra acessos indesejados, tráfego indesejado, proteger os serviços que estejam a correr e filtrar pacotes. A firewall mais usada é a iptables , mais conhecida pelo seu GUI firestarter, que foi introduzida no kernel 2.4 em subtituição da ipchains. A firewall não funciona por si mesma. Não é instalar e esperar ter tudo configurado. É necessário criar as regras necessárias para o sistema.

Então, o que proteger? O melhor a fazer é abrir apenas o necessário. Ou seja ter uma configuração restritiva. É aconselhado bloquear as portas abaixo de 1024 porque estas executam serviços que utilizam o modo de root. Os serviços como rlogin, telnet, ftp, NFS, DNS, LDAP, SMTP, RCP, X-Window devem ser permitidos apenas a máquinas de confiança, uma vez que são potencialmente inseguros podendo ser usados, não só para atacar a máquina, como também como ponte para atacar outros. O hacker pode usar a máquina da vítima para se proteger e lançar o ataque a partir doutra máquina.

## Dicas

-Não ter uma conta com uma password que seja adivinhada facilmente. Password's desse tipo são por exemplo a data de nascimento, o nome, "123456"...

-Nunca fornecer password's a outras pessoas.

-Instala, sempre que houver, actualizações de segurança. Por vezes há bug's em distribuições que podem ser fatais. Felizmente a comunidade da grande parte das distros é bastante rápida a encontrá-los e a disponibilizar patch's de actualização.

-Limita o número de processos na máquina.

-Proteger contra Fork Bombs. Uma Fork Bomb é uma forma de executar um DoS (Denial of Service), um ataque que sobrecarrega o sistema e que faz a maquina andar extremamente lenta. É difícil executar esse ataque se tiver uma firewall bem configurada uma vez que não entram no pc, mas "mais vale prevenir que remediar"... Uma Fork Bomb cria muitos processos até a maquina crashar completamente. Para se protegerem têm de definir no ficheiro limits.config que está em /etc/security/ o número máximos de processos que a máquina pode correr. Adicionem as seguintes linhas:

```
username soft nproc 100
username hard nproc 150
```

Isto previne os utilizadores de terem mais de 150 processos a correr e mostra um aviso se 100 processos estiverem a correr. É apenas um exemplo, poderão definir consoante as necessidades das vossas máquinas.

-Configure a firewall de um modo restritivo, apenas para o necessário. Deste modo estará a dar um grande passo na sua protecção.

Joel Ramos & Miguel Pais

# Sistema de directórios

## Introdução

Uma das primeiras diferenças que salta a vista para quem começa a usar Linux é a estrutura de directórios do Linux, que nada tem a ver com o Windows. Pode parecer um pouco aleatório, mas na realidade existe uma lógica e função para cada directório. Assim, o objectivo deste texto é apresentar e familiarizar com o sistema Linux.

## O Sistema

A primeira coisa que é preciso perceber é que em Linux os discos e partições não aparecem necessariamente como unidades diferentes, como o C:, D:, do Windows. Tudo faz parte de um único directório, chamado de "root", representado por "/", em que tudo fica acessível a partir de vários sub-directórios. (Não confundir com o utilizador 'Root' que é o Administrador)

Um aspecto sobre o sistema Linux que é interessante saber, é que tudo num sistema Linux é um ficheiro, se não é um ficheiro, então é um processo.
Se assim é, então o que é um Directório? São ficheiros com listas de outros ficheiros. Navegando no sistema de ficheiros do Linux, no "/" podemos encontrar os seguintes directórios:

bin/ dev/ home/ lost+found/ proc/ sbin/ usr/ boot/ etc/ lib/ mnt/ root/ tmp/ var/

Segue-se a explicação de para que servem cada um destes directórios:

### /bin
Abreviatura para "Binaries" - Binários. Contém a maioria dos executáveis mais comuns que serão usados pelos utilizadores, administradores e sistema Normalmente contém as shells como bash, csh, assim como comandos, por exemplo: su, tar ou uname.

### /sbin
Abreviatura para "System Binaries" - Binários do Sistema. É semelhante ao /bin, mas contém apenas programas essenciais ao sistema, como administração e manutenção e é essencial para o seu arranque.

### /boot
Contém os ficheiros de arranque e o kernel do Linux, que são acedidos antes do resto do sistema estar montado, e parte dos bootloaders (LILO e GRUB).

### /dev
Contém os ficheiros que representam o hardware presente na máquina. Encontram-se ficheiros como hda1, hda2, cdrom, etc. que representam dispositivos reconhecidos e instalados no Linux.

### /etc
Contém a maioria de todos os ficheiros de configuração e manipulação do sistema Linux. Nele podem encontrar por exemplo ficheiros de configuração de programas, do sistema gráfico X11, scripts de arranque.

### /home
Contém as pastas de cada utilizador (Home Directory). Os utilizadores comuns só conseguem aceder à sua própria pasta (a não ser que lhes especifiquem outro tipo de acessos). Dentro de cada pasta de utilizador também se encontram os

ficheiros de configuração respectivos para cada um. Pode ser comparado ao "Documents and Settings" do Windows

### /lib
Contém as bibliotecas básicas do sistema que podem ser compartilhadas por diversos programas. Este directório pode ser comparado ao directório System32 ou System do Windows.

### /lost+found
Depois de um "crash" do sistema, como por exemplo falta de energia, o Linux irá tentar recuperar o que está corrompido no próximo arranque. Caso um ficheiro esteja corrompido, ele será colocado neste directório.

### /mnt
Abreviatura para "mount point" (ponto de montagem). Contém a maioria das montagens do sistema operactivo. É aqui que irá ter acesso aos Cdroms, Usbs, Partições, etc. Em algumas distribuições, certas unidades como Usbs e cartões de memória são montados em /media. Pode criar os mount points que precisar, pois não existem limitações nos mount points.

### /opt
Este directório contém programas opcionais, como por exemplo KDE. Cada software terá a sua própria pasta.

### /proc
É um pseudo-directório que fornece informações sobre o kernel e sobre os processos que estão activos no momento, além de informações sobre a utilização de alguns dispositivos e do sistema.
Cada sub-pasta representa um processo.
Alguns programas como 'ps' ou 'top' acedem a esta pasta para recolher informação do sistema em vez de comunicar directamente com o kernel.

### /root
Este é o Home Directory do utilizador root ( Administrador ), onde só ele tem acesso.

### /tmp
Este directório armazena os ficheiros temporários. É limpo durante o arranque e shutdown do seu sistema, Não é aconselhável guardar aqui ficheiros importantes.

### /usr
Abreviatura para "Unix System Resourses" Contém todos os comandos, bibliotecas, programas, páginas de manuais e outros ficheiros estáticos que são necessários para o funcionamento normal do sistema.

### Algums sub-directórios:

/usr/doc - Contém grande parte da documentação do Linux

/usr/src/linux - Contém o código fonte do Kernel.

/usr/bin - Contém outros ficheiros binários referentes aos utilizadores, mas que não são essenciais.

### /var
Contém ficheiros que possuem dados variáveis, como ficheiros de logs, ficheiros de configuração de correio electrónico, de impressão, entre outros.

## Conclusão

Este texto é uma apresentação simples do Sistema de Directórios do Linux, mas um conhecimento necessário para poder aproveitar o potencial que este sistema oferece.
A melhor maneira de perceber e aprender é usar.

Marco Silva

http://orionsbelt.zi-yu.com

## Introdução

O Orion's Belt é um jogo web-based elaborado por dois portugueses como trabalho final de bacharelato. Está escrito em C# e funciona em cima de ASP.NET. Este jogo tem várias características semelhantes a outros jogos do género mais conhecidos: Ogame e Darkgalaxy. No OB o jogador começa com um home planet e pode partir à conquista de outros planetas pela galáxia. A gestão de planetas é um dos aspectos mais importantes do jogo, sendo necessária a construcção de edifícios, elaboração de pesquisas, etc, para evoluir tanto tecnologicamente como ecnomicamente.

O aspecto que diferencia o OB dos outros jogos do género é o modelo de combate. Todos os combates são realizados num tabuleiro com as unidades produzidas nos planetas. Este motor de combate está implementado em JavaScript.

Cada unidade tem um conjunto de características como: ataque, defesa, dano, vida, alcance e tipo de movimento. Algumas unidades têm ainda características especiais como: ataque ricochete, ataque catapulta, ataque triplo, bónus de ataque/defesa em certos tipos de terreno ou contra certos tipos de unidades.

Todas estas características fazem com que o combate seja muito táctico, e um dos aspectos mais aliciantes do jogo.

No OB há inclusive torneios à parte em que os jogadores são fornecidos com uma armada, e têm de passar uma fase de grupos e de seguida os playoffs. Estes torneios contam para um sistema de Ranking baseado no ELO Ranking System (usado no Xadrez), e permitem saber quem são os melhores jogadores no tabuleiro.

Finda a introdução, vão ser abordados dois temas da arquitectura do Orion's Belt.

## Sistema de Recursos

O OB conta com um grande número de recursos: 43 edifícios, 25 unidades e 48 pesquisas. A fácil manuntenção/adição de recursos foi um dos primeiros objectivos para o motor de jogo. Para este fim foi criado um idioma XML capaz de representar esta informação.

Este idioma XML permite descrever facilmente todas as capacidades de um qualquer recurso, seja ele edifício, pesquisa ou unidades de combate. É possível descrever o preço, as dependências e características várias.

O uso de XML fez com que o motor seja mais simples e algo genérico. Contudo, há várias outras vantagens em se ter usado XML para descrever os recursos. Através do uso de XSLT (idioma XML que transforma XML noutros formatos) foram criadas templates que pegam no XML dos recursos e o transformam em páginas do manual (formato wiki) e também em ficheiros JavaScript para uso do motor de combate.


O tabuleiro de jogo do Orion's Belt

```
<resource type="Unit" value="ColonyShip">
 <dependencies>
  <resource-ref type="Building" value="StarPort" />
  <resource-ref type="Research" value="AdvancedFlightI" />
 </dependencies>
 <cost>
  <resource-needed type="gold" value="5000" />
  <resource-needed type="mp" value="8000" />
  <resource-needed type="energy" value="1000" />
  <resource-ref type="Building" value="StarPort" />
  <duration value="15" quantity="1" />
 </cost>
 <attributes>
  <attribute type="TeletransportationCost" value="350"/>
 </attributes>
 <oncomplete/>
 <battle unitType="special">
  <attack base="230" range="1" minimumDamage="200" maximumDamage="250"/>
  <defense base="8000" hitPoints="9200" canStrikeBack="true" />
  <movement cost="6" type="all" level="air" />
 </battle>
</resource>
```

*Idioma do Sistema de Recursos*

## Bases de Dados

Uma aplicação web normalmente está associada a um SGBD (sistema de gestão de bases de dados) como o MySQL, PostgreSQL, etc. O OB suporta vários tipos de SGBD através da design pattern bridge. Toda a camada de acesso a dados foi pensada como uma abstracção ao SGBD usado em runtime. Para atingir este fim é criada uma classe base abstracta com as funcionalidades elementares e classes derivadas que contêm especializações para cada SGBD. Por exemplo, para persistir as Alianças, existe a classe AllianceUtility:

```
public class AllianceUtility {
 public abstract Alliance Get(int id);
 public abstract Alliance[] GetAll();
 public abstract void Save(Alliance a);
 ...
};
```

Cada classe Utility usa também o design pattern singleton, para providenciar um único objecto daquele tipo, sempre disponível. É usada uma propriedade static que indica qual o objecto que trata da persistência. É aqui que há a ponte para a implementação corrente. O SGBD alvo é decidido em compile time através de parâmetros no compilador.

Desta forma é possível ter várias implementações de persistência, que são escondidas da restante lógica de aplicação.

## Conclusão

O OB é um projecto já com alguma dimensão. Contudo, certas escolhas bem pensadas no início de desenvolvimento tornaram-se grandes escolhas no futuro do jogo, pois permitem uma excelente facilidade de manuntenção.

Convidamos todos os interessados a experimentar o jogo. O jogo é dividido em rondas e, por volta de Setembro, irá começar uma nova ronda. Para esta nova ronda estão planeadas algumas novidades interessantes: possibilidade de construção de Death Stars, estas luas de combate são inspiradas pela Death Star da Guerra das Estrelas, e combates com quatro jogadores ao mesmo tempo - torneios com combates 1x1x1x1 ou 2x2.

Quem tiver mais curiosidade no jogo pode passar pelo endereço:
http://pt.wiki.zi-yu.com/index.php/Vídeos
Neste endereço estão vídeos de dois combates e um de introdução global ao OB.

Pedro Santos

# Opera Mini

O mais frustrante para os actuais web dependentes é não poder consultar o e-mail ou visitar uma determinada página web só porque está longe de um pc. Hoje em dia existem outras formas de aceder à Internet sem fazer uso de um computador, se pensarmos bem quase todos nós possuímos um telemóvel com gprs com suporte para aplicações java, que permite uma ligação estável de 56kbps, mas o facto é que a maioria deles não possuem um browser html realmente funcional, ou pura e simplesmente não o possuem.

Pois agora já existe uma opção de alta qualidade para quem quer tirar todo o potencial da Internet associada ao vosso telemóvel, pda ou smartphone. E enganem-se aqueles que pensam que são poucas as plataformas e telemóveis suportados, uma vez que a lista de equipamentos é extensa e é constantemente actualizada. Posso dizer que o meu 7250i se tornou um companheiro importante no tempo em que estive longe da Internet, e me permitiu consultar o e-mail no gmail, visitar a minha página e por vezes visitar inclusive a página do fórum portugal-a-programar.org.

Apesar das páginas serem demasiado grandes para qualquer ecrã de um telemóvel, o Opera Mini consegue digerir complicadas páginas e torná-las muito simples de forma a facilitar a sua visualização. E apesar de algumas páginas serem muito grandes e ficarem muito compridas o Opera Mini permite-nos navegar na página com imensa facilidade, clicar em links, fazer scroll e visualizar com pormenor as imagens.

Entre as ferramentas incluídas estão um histórico de páginas visualizadas, um gestor de favoritos, suporte automático para pesquisas no google e alguns dicionários, suporte para download de imagens e um gestor de passwords.

As restantes funcionalidades podem diferir da versão do Opera mini, uma vez que existem duas a Basic e a Advanced. A Advanced possui uma melhor compressão de textos e de imagens o que permite melhorar o tempo de carregamento, assim como um interface melhorado com suporte para icons, scroll mais suave e um relógio. A Advanced consegue também melhores resultados a 'renderizar' páginas grandes criando uma grande página, enquanto que a Basic tem de criar vários segmentos de página.


Como se não bastasse este software é freeware e de fácil instalação bastando aceder a mini.opera.com com o wap do vosso telemóvel e a página indicará os passo necessários para instalarem o Opera Mini, seleccionando a versão correcta. Poderão consultar a lista de dispositivos compatíveis em http://www.opera.com/products/mobile/operamini/phones.

Tendo em conta que o Opera sempre foi um browser interessante, não surpreende que o Opera Mini siga as passadas do seu irmão mais velho, e em dispositivos móveis é actualmente o rei. Fica apenas a faltar o suporte para flash, porque de resto não existe nada a apontar de negativo a este browser.

Actualmente o Opera está também a ser incluído em outros tipos de equipamentos como consolas (Nintendo Wii e DS), Internet Tablets (Nokia 770), Portable Media Players (Archos PMA400), aviões e tv box's. Resta saber qual é o limite deste browser.

Sérgio Matias

# Programação Orientada a Objectos em JAVA 2


O livro fala de programação orientada aos objectos usando a linguagem JAVA. É da autoria de F. Mário Martins, professor associado da Universidade do Minho e coordenador do grupo SIM - Software, Interacção e Multimédia.

Nesta obra começam por ser abordados os principais conceitos associados à programação orientada aos objectos, nomeadamente o que é um objecto, o encapsulamento e a diferença entre instâncias e classes. A maior parte do livro é, contudo, dedicada à tecnologia JAVA, sendo abordados os aspectos mais importantes que esta linguagem suporta (classes, hierarquias, excepções, interface, etc). Possui muitos exemplos de código onde são aplicados os conceitos abordados, facilitando a compreensão dos mesmos. Nota-se também a preocupação de ensinar técnicas de programação que permitam que o software desenvolvido satisfaça propriedades essenciais hoje em dia, como por exemplo a modularidade e a extensibilidade.

Parece-me um livro bastante indicado para quem quer aprender JAVA ou para quem procura aprender metodologias de programação que permitam aumentar a qualidade das aplicações que desenvolve. Não será, no entanto, o livro mais indicado para quem já tem alguma experiência com JAVA e com POO em geral, pois aspectos mais avançados da linguagem não são abordados.

Rui Gonçalves

# Linguagens Web


"Linguagens Web" é um livro de iniciação à programação orientada à Web que reúne todas as principais linguagens de programação utilizadas na Internet, de forma concisa e bem apresentada para possibilitar uma aprendizagem mais rápida e progressiva.

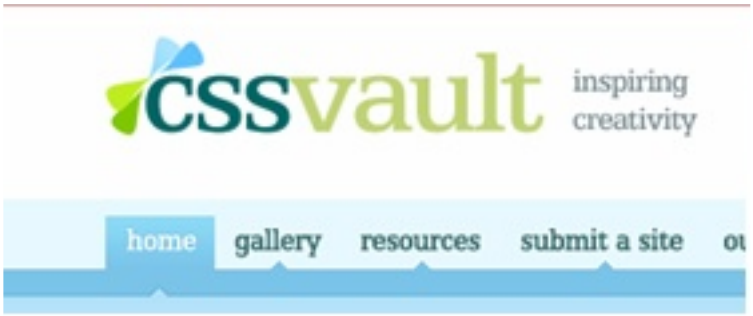
Este livro foi escrito por Alexandre Pereira e Carlos Poupa e editado por "edições Sílabo". Data do ano 2004 e possui cerca de 450 páginas.

As linguagens abordadas são: HTML, CSS, JavaScript, ASP, ASP.NET (VB.NET e C#), PHP e Java.

Fala um pouco sobre cada linguagem onde é abordada a sua estrutura, a sua sintaxe, sua formatação e como a utilizar correctamente.
Esta obra é aconselhada a qualquer programador no entanto é excelente para programadores que se estejam a iniciar pois utiliza uma linguagem clara e de fácil entendimento com exemplos explicados para uma mais rápida e eficiente aprendizagem.

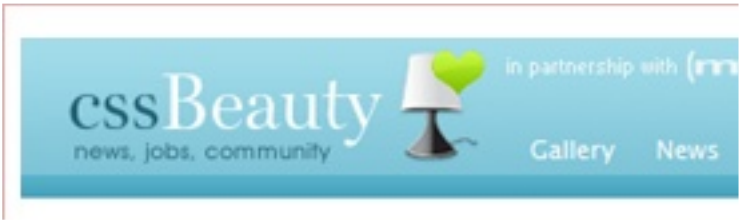
É fornecido apoio on-line a este livro em http://linguagens.mediateca.info

João Pereira




O cssvault disponibiliza uma larga galeria de websites de qualidade, organizados por mês e ano, todos eles de grande qualidade. É sem dúvida um excelente sítio de consulta do que de melhor existe no momento na Web 2.0.
http://cssvault.com

Por seu lado o cssBeauty oferece aos seus utilizadores uma galeria semelhante ao do cssvault, mas também ofertas de emprego, artigos de css, html, javascript e outros de grande interesse para os web-developers.
http://www.cssbeauty.com

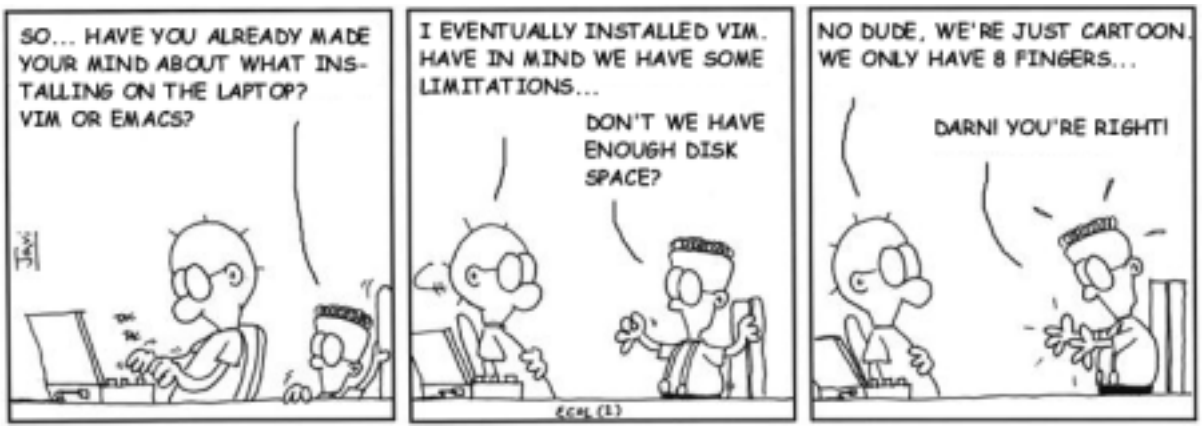



Foi criada em Março deste ano – e só agora descobrimos – uma pequena página com 12 perguntas frequentes de utilizadores que nunca viram Linux na sua vida. Esta página criada pelo Grupo de Linux da Universidade de Aveiro está disponível para consulta em http://glua.ua.pt/Linux.


Muito se tem falado no que é a Web 2.0 realmente, qual os benefícios que poderemos tirar disso. Pois bem, aqui está mais um documento a tentar explicar isso mesmo.

http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html

Daniel Correia




emacs ou vim?


mais um clássico...


o segredo por trás do Pentium 2

Um ano passado da criação da comunidade e, como todos devem saber, as coisas vão de "vento em poupa".

A comunidade conta já com mais de 1800 utilizadores registados e com uma média de visitas diárias a tocar o algarismo dos milhares. Houve nestes últimos tempos a saída de alguns membros do staff (Sara Silva, Bruno Monteiro, Hugo Violante) por motivos profissionais e/ou pessoais, mas contamos já com a participação activa de novas pessoas com vontade de construir e inovar (Fernando Martins, Sérgio Lopes, João Matos e José Oliveira).

Dentro em breve poderão contar com uma nova ferramenta de apoio, o wiki do p@p, que estará pronto e acessível a todos para esclarecer de forma mais prática todas as vossas dúvidas e também para poderem dar as vossas contribuições, que são obviamente parte fundamental de toda a comunidade. Estamos também no início da criação de uma nova plataforma de aprendizagem, baseada no moodle. Ainda temos muito trabalho pela frente, mas possuímos grandes perspectivas para esta plataforma.

Vários pequenos projectos têm vindo a aparecer dentro da comunidade e são algo que realmente merece a pena ver. Portanto, para conhecerem bem a comunidade, nada melhor do que passar por...

www.portugal-a-programar.org

David Pintassilgo

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