PROGRAMAR

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

EDIÇÃO #37 - OUTUBRO 2012

ISSN 1647-0710

EDITORIAL

EQUIPA PROGRAMAR

Contacto
revistaprogramar@portugal-a-programar.org

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

ISSN
1 647-071 0

O que existe do lado de lá da montanha?

O que existe do lado de lá da montanha ? É a pergunta que muitas vezes nos fazemos, referindo-nos "à montanha de trabalho que temos para fazer", ao "montanhoso problema que não se quer resolver", ao "íngreme bug, que teima em não identificar onde se encontra", "à barreira e velocidade imposta pelo hardware", etc…

O que existe do lado de lá dessa montanha, é simples de saber, basta "escalá-la" para descobrir! Sem nunca desistir! Recordo-me com nostalgia do tempo em que se dizia que a "mítica barreira" dos 100mhz não iria ser ultrapassada! Mas foi! Recordo-me como muitos de vós recordarão das promessas e especulações em torno da tecnologia MMX, que acabaram revelando-se na sua maioria infundadas, e infrutíferas, pois a "montanha" parecia ser muito alta, mas na realidade pouco mais era que um "montinho".

Hoje em dia ao ler código, ao ler um qualquer artigo, ao programar, penso, "quantas linhas teria eu escrito em C ou Assembly para conseguir fazer algo como isto… E quantas linhas escrevi agora! A "montanha" deixou de ser "inescalável".

Ainda agora por vezes me deparo, como certamente a maioria dos leitores se depara com "aquele problema chato" que não parece ter solução, ou aquele erro, que já me fez olhar para o código "duzentas vezes" sem ver onde estará ele… Para agora descobrir que afinal é uma vírgula ou uma aspa que me esqueci. Esse "detalhe" que quase nos faz desesperar, que nos leva a pensar em desistir, e que no fim, depois de o resolvermos, descobrimos a alegria de o ter superado! Essa superação será para os programadores "o outro lado da montanha".

É esse alento, essa determinação de "superar mais uma montanha" que move muitos dos programadores, se não "a maioria". Essa vontade de "vencer" que começa a crescer e nos ajuda a fazer "dos grandes problemas" pequenos problemas, da grande complexidade, pequenos "detalhes" e no fim da "jornada", nos faz acreditar que todo o esforço, feito, valeu a pena, pois se não o fizéssemos, nunca saberíamos o que existe "para lá da montanha"

Na Revista, edição após edição uma "nova montanha" se ergue, e edição após edição descobrimos, com a preciosa ajuda dos nossos leitores, o que existe "do lado de lá da montanha".

E agora que lançamos esta edição, preparamo-nos para vencer "a montanha que se segue", como equipa, todos juntos, de nós, para vós! Na certeza que que daremos o nosso melhor por uma edição ainda melhor! De nós, para vós! Por todos nós! Até lá!

António Santos
<antonio.santos@revista-programar.info>

ÍNDICE

TEMA DE CAPA

A PROGRAMAR

COLUNAS

ANÁLISES

COMUNIDADES

NO CODE

EVENTOS

04 Out 2012	Comunicar nas Redes Sociais
04 Out 2012	1ª Edição Rumos Inside Out
11 Out 2012	Oracle Customer Showcase Forum
20 Out 2012	Comemoração do 3º Aniversário da Comunidade NetPonto em Lisboa
07 Nov 2012	Oracle Day 2012
15-17 Nov 2012	Sapo CodeBits VI

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

W3C anuncia plano para distribuir o HTML5 em 2014 e HTML 5.1 em 2016

Quebrar a especificação em pedaços menores,vai permitir a normalização mais rápida.

O World Wide Web Consortium (W3C), o grupo que gere o desenvolvimento das principais especificações utilizadas pela Wev, propos um novo plano que iria ver a especificação HTML 5 posicionada como Recomendação o que em linguagem W3C representa um complete, terminado padrão – pelo final de 2014. O grupo planeia um seguimento, o HTML 5.1, para o final de 2016.

Ao abrigo do novo plano, o Grupo de Trabalho HTML produzirá uma Recomendação Candidata do HTML 5.0 por finais de 2012 que inclui apenas as definições que estão especificadas, estáveis, e implementadas em browsers reais. O que quer que seja controverso ou instável sera excluído desta especificação. O grupo irá também remover qualquer especificação da qual seja conhecido ter problemas de inoperabilidade entre implementações existentes, Esta Recomendação Candidata formará a base da especificação 5.0.

Em conjunto, será desenvolvido um esboço do HTML 5.1. Isto irá incluir tudo da Recomendação Candidata HTML 5.0, e todas as definições instáveis que foram excluídas. Em 2014, isto será submetido a um processo semelhante. Qualquer coisa instável será retirada, para produzir a Recomendação Candidata HTML 5.1, e surgirá um esboço do HTML 5.2, com as partes instáveis incluídas.

Isto continuará depois para o HTML5.3, 5.4, e assim por diante.

Anteriormente, não estava previsto o HTML 5 estar complete até 2022 (sim, a uma década de agora). A Recomendação Candidata deveria ser distribuída por esta altura, com grande parte dos próximos dez anos gastos a desenvolver um conjunto extensivo de testes para permitir testes de conformidade das implementações. O novo HTML 5.1 será menor como um número de tecnologias (tal como Web Woekers e WebSockets) uma vez o foram sob o guarda chuva HTML 5, mas foram agora repartidas em especificações separadas. Também terá requisitos de teste menos rigorosos. Porções da especificação onde a inoperabilidade tenha sido demonstrada “in the wild” não necessitarão de novos testes, mas em vez disso focar-se-ão em novas características.

A padronização HTML 5 tem sido um processo turbulento, com muitos argumentos e discussões enquanto grupos diferentes, com prioridades diferentes se esforçaram para encontrar um terreno comum. O novo plano nota que o "tom negativo da discussão tem sido um problema permanente" e diz que o Grupo de Trabalho terá de ser melhor para combater o comportamento anti-social. O plano proposto não foi, no entanto universalmente bem-vindo. Alguns membros do Grupo de Trabalho ficaram descontentes com o tratamento proposto de suas áreas específicas de actuação.

Para os programadores WEB, o impacto do novo plano poderá ser limitado; os programadores estão já habituados a trabalhar em especificações do zero numa base diária. A consequência mais imediata nessas peças terem sido consideradas estáveis o suficiente para serem incluídas na versão 5.0 deveria adquirir um conjunto de testes mais rico. Por sua vez, isso irá ajudar os programadores de browser a localizar (e, com sorte, remediar) quaisquer bugs ou incompatibilidades restantes.

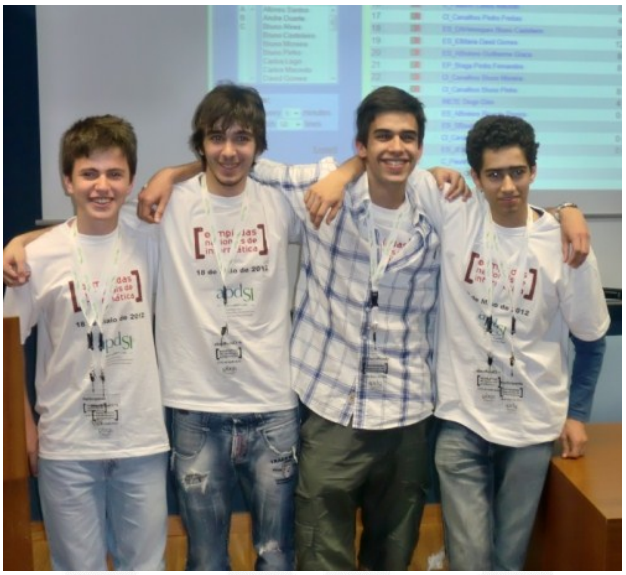
Fonte: ArsTechnica.com

Tradução: Sara Santos

Portugal à conquista das Olimpíadas Internacionais da Informática

A partir do próximo dia 23 de setembro Portugal incia a sua participação na 24ª edição das International Olympiad in Informatics (IOI), que este ano decorre na cidade italiana de Sirmione até 30 de setembro.

A formação portuguesa é composta por quatro estudantes do ensino secundário, nomeadamente Afonso Santos (Escola Técnica e Liceal de S. António do Estoril), Francisco Machado (Escola Secundária Infanta D. Maria), Pedro Paredes e João Ramos (Escola Secundária de Avelar Brotero) e (Escola Secundária de Avelar Brotero), a que se juntam um professor da Universidade do Porto, Pedro Ribeiro, e o estudante Rodrigo Gomes, que participou nas duas últimas edições das IOI.



Os quatro concorrentes portugueses garantiram a presença nas IOI no passado dia 18 de maio, ao ficar nos primeiros lugares das Olimpíadas Nacionais de Informática - ONI'2012.

Com o objetivo de "estimular o interesse dos jovens pela informática e pelas Tecnologias da Informação", a edição deste ano conta a participação de mais de 80 países, fazendo parte do leque de seis olimpíadas internacionais ligadas à ciência que todos os anos se realizam - e que contemplam ainda as disciplinas de Matemática, Física, Química, Biologia e Astronomia.

A participação portuguesa é organizada pela APDSI - Associação para a Promoção e Desenvolvimento da Sociedade da Informação, contando com os patrocínios da Fundação Calouste Gulbenkian e da Secretaria de Estado do Desporto e da Juventude.

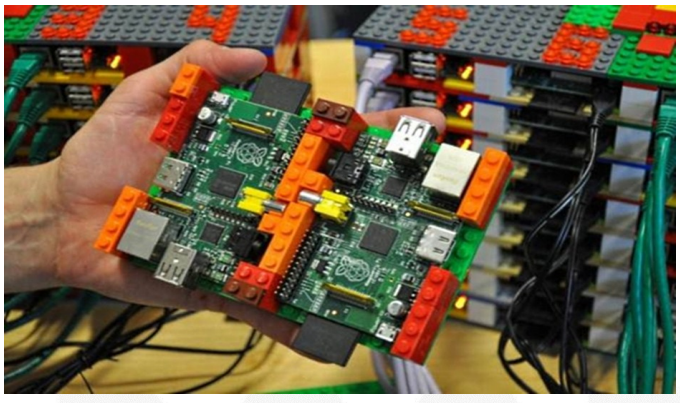
Escrito ao abrigo do novo Acordo Ortográfico

Fonte:TekSapo

Cientistas constroem "supercomputador" com 64 Raspberry Pi e Lego

Juntam-se 64 microcomputadores Raspberry Pi, uma mão-cheia de switches e cabos de rede Ethernet; depois, criam-se estruturas em Lego para suportar todas estas máquinas; por fim, liga-se tudo e adiciona-se software "q.b.", de modo a criar um "supercomputador".
É esta a proposta de um grupo de engenheiros informáticos da Universidade de Southampton, no Reino Unido, liderado pelo Professor Simon Cox e acompanhada por mais oito pessoas - incluindo James Cox, o filho de 6 anos do mentor do projeto, que participou como "especialista" na componente da estrutura em Lego e nos testes ao sistema.

O sistema dá pelo nome de "Iridis-Pi", funciona com uma ligação normal à rede elétrica e é constituído por um total de 64 processadores. A capacidade de armazenamento total é de 1TB, conseguidos através da utilização de um cartão SD com 16GB por cada microcomputador, enquanto a ligação de todas as máquinas entre os vários "nós" é gerida por software MPI (Message Passing Interface).



O custo do sistema criado rondou os 3.200 euros, excluindo os switches.

Quanto ao software para desenvolvimento do sistema, foi usado o plug-in gratuito Python Tools for Visual Studio para a criação de código, a que se juntam algumas aplicações criadas por James Cox em Scratch.

"A equipa pretende ver este sistema low-cost como um ponto de partida, para inspirar e permitir que os estudantes apliquem a computação de elevado desempenho à gestão de dados" refere o Professor Simon Cox no comunicado da Universidade de Southampton, explicando que este "supercomputador" pode ser usado em sistemas complexos de engenharia e em "desafios científicos".

Mais direto nas palavras esteve o filho de Cox, que considerou o Raspberry Pi "muito divertido", destacando o facto de o poder segurar com uma mão, "escrever programas de computador e jogar jogos com ele".

Para os interessados em criar uma máquina semelhante baseada em vários Raspberry Pis, a equipa liderada pelo Professor Simon Cox partilha a informação necessária numa página, onde explica todos os passos necessários, possibilitando ainda a transferência de um manual em PDF.

Escrito ao abrigo do novo Acordo Ortográfico

Fonte: TekSapo

TEMA DE CAPA

Makefiles

Makefiles

Criação de Makefiles

Antes de começar, convém referir que neste artigo assumimos a utilização de um ambiente *nix com acesso a uma shell (bash neste caso) e com o GNU make e gcc instalados. Mais à frente no artigo, é utilizado também o utilitário sed, que implica conhecimento de expressões regulares (regex).

Há quem dispense o IDE a favor da linha de comandos e compile os seus programas manualmente, algo especialmente importante para os iniciados, uma vez que proporciona uma excelente oportunidade para compreender de forma relativamente aprofundada o processo de compilação. Para projectos simples, a compilação directa na linha de comandos é perfeitamente apropriada. No entanto, torna-se conveniente automatizar a compilação de projectos mais complexos, e uma das ferramentas criadas para isso mesmo é o make.

Como introdução a este artigo, começamos com uma pequena secção dedicada à compilação manual de programas escritos em C. Esta secção não pretende ser muito completa, uma vez que não é esse o tema principal do artigo, embora seja importante. Passamos depois para a compilação através da criação de um Makefile, o qual será melhorado sucessivamente até ser capaz de calcular automaticamente as dependências necessárias para a compilação.

Compilação manual com o gcc

Imaginando um projecto com os ficheiros main.c e foo.c (que implementam um programa que simplesmente imprime a string “This is an app.” e sai), podemos compilar o nosso programa para um executável app com o seguinte comando:

```
$ gcc -o app main.c foo.c
$ ./app
```

This is an app.

A flag -o indica que o parâmetro seguinte é o nome do executável a criar. É aconselhável que a compilação seja feita com as flags -Wall -Wextra de forma a emitir avisos pertinentes sobre o código (variáveis não utilizadas, entre outros):

```
$ gcc -Wall -Wextra -o app main.c foo.c
$ ./app
```

This is an app

Em projectos de dimensão maior é benéfico compilar cada ficheiro .c separadamente e ligá-los (fazer linking) apenas no final. Desta forma podemos recompilar apenas os ficheiros .c alterados, diminuindo assim o tempo de compilação total. Para isso, devemos utilizar flag -c e compilar cada ficheiro .c individualmente. Assim:

```
$ gcc -Wall -Wextra -c -o main.o main.c
$ gcc -Wall -Wextra -c -o foo.o foo.c
$ gcc -o app main.o foo.o
$ ./app
```

This is an app.

Atenção: como as flags -Wall e -Wextra afectam apenas a compilação não são utilizadas na ligação dos ficheiros compilados no executável final.

Introdução ao make

Vamos agora automatizar a compilação com o make, um programa que determina que ficheiros precisam de ser recompilados, e emite comandos para essa recompilação.

O make funciona através de um conjunto de regras que contêm os comandos necessários para a compilação de programas (e outras tarefas, como a sua instalação). Existe um conjunto de regras implícitas que permite compilar programas nas linguagens mais populares (neste artigo, C). Imaginando um programa que imprime a string *“Hello world!”* e sai, cujo código se encontra no ficheiro *hello.c*, podemos fazer na linha de comandos:

```
$ make hello
cc hello.c -o hello
$ ./hello
```

Hello world!

No entanto, o potencial do make é permitir ao programador definir as suas próprias regras (explícitas). Estas são lidas do ficheiro Makefile (ou outro, especificado com a flag -f) no formato seguinte:

```
# comentários
alvo: dependência ...
    comando
    ...
```

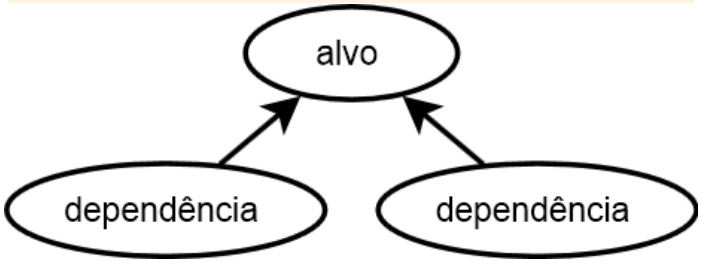


Figura 1: Grafo de dependências do pseudo-Makefile.

Para simplificar, a regra acima apresenta apenas um alvo, mas o make suporta regras com vários alvos (separados por um espaço), tal como na lista de dependências. Cada co-

mando deve ser colocado numa linha, e é muito importante perceber que é *obrigatório* preceder cada comando de *uma tabulação* (e não de espaços).

Quando invocado, o make começa por procurar o ficheiro alvo. Se este existir, são procuradas as dependências e comparadas as suas datas de modificação. Quando as dependências são mais recentes que os alvos (ou quando o alvo não existe), o make actualiza o alvo (executa os comandos). Quando as dependências não são encontradas, o make tenta recriá-las (se houver regras para isso). Caso contrário, a execução termina dizendo que não há nada a fazer (está tudo actualizado). Tomando a Figura 1 como referência (que contém um grafo de dependências para o nosso pseudo-Makefile), podemos verificar que sempre que algum nó do grafo é mais recente que os nós em níveis superiores, estes são considerados desactualizados e serão, portanto, recriados.

A execução do make começa pela primeira regra do ficheiro. Podemos especificar outra regra na linha de comandos, passando o seu alvo como argumento ao make (ex.: make alvo).

O make permite-nos definir várias regras com o mesmo alvo, e isso permite-nos construir uma lista de dependências de forma incremental, uma vez que todas são agregadas pelo make no final. No entanto, é importante saber que não acontece o mesmo com os comandos dessas regras. Quando um alvo está presente em várias regras, apenas os comandos da última regra são utilizados. Este ponto será importante quando tentarmos, mais à frente, importar um conjunto de regras pré-feitas.

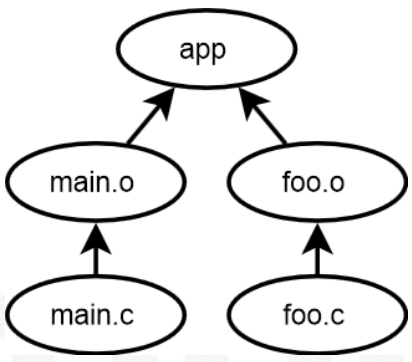
Um exemplo (pouco) real

Imaginemos agora um pequeno projecto com os ficheiros main.c e foo.c. Como o objectivo é fragmentar a compilação, compilando apenas os ficheiros .c que forem alterados, definimos então as regras:

```
app: main.o foo.o
     gcc -o app main.o foo.o

main.o: main.c
     gcc -Wall -Wextra -c -o main.o main.c

foo.o: foo.c
     gcc -Wall -Wextra -c -o foo.o foo.c
```



É notável a repetição neste Makefile: gcc está escrito em todas as regras e -Wall -Wextra em duas delas. Se for necessário trocar de compilador ou alterar as suas flags, será necessário alterar todas as regras manualmente (embora um editor de texto decente faça isto facilmente). Para evitar esse trabalho, o make permite-nos definir variáveis:

```
CC := gcc
CFLAGS := -Wall -Wextra
...
main.o: main.c
    $(CC) $(CFLAGS) -c -o main.o main.c
...
```

Ao invocarmos o make (e executarmos o nosso programa app), obtemos o seguinte output:

```
$ make
gcc -Wall -Wextra -c -o main.o main.c
gcc -Wall -Wextra -c -o foo.o foo.c
gcc -o app main.o foo.o
$ ./app
```

This is an app.

Existem algumas particularidades relativas à definição de variáveis (que o leitor poderá explorar no manual do make); ao contrário do indicado na maioria dos tutoriais, devemos utilizar sempre := e só quando necessário o =. O motivo para tal é simples: nas variáveis definidas com := (atribuição imediata) o valor da variável é determinado imediatamente, e se utilizarmos = (atribuição deferida) o valor é calculado de cada vez que a variável é utilizada. A utilização de := é, portanto, mais previsível.

No news is good news

Como diz o ditado, no news is good news, que significa que a ausência de output é sinal de que tudo correu bem. Como se constata nos exemplos anteriores, o make imprime no ecrã os comandos que executa, e para projectos mais complexos este output pode ser excessivo. É boa ideia eliminá-lo, de forma a que o output proveniente dos programas executados seja facilmente identificado.

Para isso, o make possui uma flag -s que cancela o seu output. Uma alternativa (ainda melhor) é preceder os comandos a esconder do símbolo @, que permite escolher quais os comandos a não mostrar.

```
main.o: main.c
    @$(CC) $(CFLAGS) -c -o main.o main.c
...
```

Desta vez, a invocação do make não produz nenhum output:

```
$ make
$ ./app
This is an app.
```

Ainda assim, o leitor poderá querer ter algum tipo de feedback do make relativamente à sua actividade. Podemos, por exemplo, utilizar o utilitário echo para imprimir as nossas mensagens:

```
main.o: main.c
    @echo "A compilar main.c"
    @$(CC) $(CFLAGS) -c -o main.o main.c
...
```

Desta forma, o make irá emitir um output mais agradável:

```
$ make
A compilar main.c
A compilar foo.c
A ligar app
$ ./app
This is an app.
```

Variáveis automáticas

Começamos novamente a reparar que há vários padrões de repetição no nosso Makefile. Nas regras que geram os ficheiros .o, por exemplo, o nome do alvo está repetido no echo e no comando de compilação. Além disso, as próprias regras são uma repetição umas das outras, com alvos e dependências diferentes, mas seguem o padrão de "regras que compilam ficheiros .c em ficheiros .o". O make permite-nos generalizar estas regras. Comecemos pela primeira regra, que gera app:

```
...
OBJECTS := main.o foo.o

app: $(OBJECTS)
    @echo "A ligar $@"
    @$(CC) -o $@ $^
```

Foram introduzidas três novas variáveis: OBJECTS é simplesmente uma lista de dependências. O facto de estar separada da regra facilita a sua alteração. $@ contém o nome do alvo (app) e $^ contém a lista de dependências (main.o foo.o) – o mesmo que OBJECTS; a regra faz o mesmo que antes, mas sem repetição dos nomes de ficheiros. Relativamente às regras que geram os ficheiros .o, o caso é mais complexo porque queremos, com uma só regra, gerar qualquer ficheiro .o a partir do seu correspondente .c.

```
%.o: %.c
    @echo "A compilar $*.c"
    @$(CC) $(CFLAGS) -c -o $@ $<
```

Surgiram duas novas variáveis automáticas e um novo símbolo (%). O símbolo % é utilizado para construir padrões e representa o mesmo texto no alvo e nas dependências. A variável $* contém o valor representado por % e a variável $< contém a primeira dependência. Podíamos, obviamente, ter utilizado $@ em vez de $*.c, e $^ em vez de $< (pois só temos uma dependência), mas o objectivo foi dar também a conhecer outras variáveis automáticas.

Funções úteis do make

O nosso Makefile permite que o make compile qualquer ficheiro .c, mas não sabe que ficheiros devem ser compilados. Por outras palavras, sempre que temos um novo .c no nosso projecto, é necessário adicionar o .o correspondente à variável OBJECTS. Como seria de esperar, o make permite-nos automatizar o processo:

```
SOURCES := $(wildcard *.c)
OBJECTS := $(patsubst %.c, %.o, $(SOURCES))
```

Estamos a utilizar duas funções do make: wildcard e patsubst. Na primeira linha estamos a atribuir a SOURCES uma lista de ficheiros .c na directoria actual. Na segunda linha, estamos a substituir todas as ocorrências de .c por .o (preservando o texto representado por %) presentes na variável SOURCES e a atribuir o resultado da substituição à variável OBJECTS. O mesmo resultado seria obtido por:

```
OBJECTS := $(patsubst %.c, %.o, $(wildcard *.c))
```

Desta forma, todos os ficheiros .c presentes na directoria do projecto serão compilados para .o correspondentes, dependências de app.

Se quisermos atribuir a uma variável o output de um comando, podemos utilizar a função shell, seguida do comando a executar:

```
VAR := $(shell date)
```

Aqui podemos voltar à problemática da atribuição imediata vs deferida. Com atribuição imediata (símbolo :=), o valor de VAR será calculado uma única vez, e assim permanecerá (a menos que explicitamente alterado). Se definíssemos VAR com atribuição deferida (símbolo =), o seu valor seria calculado sempre que fosse necessário utilizá-lo e, dada a natureza do comando date, seria sempre diferente a cada utilização.

Hierarquia de um projecto

Podemos querer organizar os ficheiros do nosso projecto em directorias com significado: bin para os binários executáveis, obj para os .o, src para os .c, entre outros. Alteremos o nosso Makefile de acordo:

```
CC := gcc
CFLAGS := -Wall -Wextra
BINDIR := bin
OBJDIR := obj
SRCDIR := src
SOURCES := $(wildcard $(SRCDIR)/*.c)
OBJECTS := $(patsubst $(SRCDIR)/%.c,
$(OBJDIR)/%.o, $(SOURCES))

$(BINDIR)/app: $(OBJECTS)
    @echo "A ligar $@"
    @$(CC) -o $@ $^

$(OBJDIR)/%.o: $(SRCDIR)/%.c
```

```
    @echo "A compilar $*.c"
    @$(CC) $(CFLAGS) -c -o $@ $<
```

Coloca-se agora uma questão importante: se as directorias bin e obj não existirem, não será possível compilar o programa. Podemos adicioná-las como dependências, mas também isso traz problemas, pois quando um ficheiro é alterado, também a data de modificação da directoria é alterada e todos os ficheiros lá dentro (que dela dependem) ficam desactualizados, obrigando a uma recompilação completa. Para estas situações o make tem um tipo especial de dependências para as quais apenas se verifica a sua existência (não são comparadas as datas de modifcação). São listadas à direita das dependências normais, precedidas de uma barra (|):

```
...
$(BINDIR)/app: $(OBJECTS) | $(BINDIR)
...
$(OBJDIR)/%.o: $(SRCDIR)/%.c | $(OBJDIR)
...
$(BINDIR) $(OBJDIR):
    @mkdir $@
```

Quando as directorias não existirem, o make irá invocar a regra para as criar. Repare-se que esta última regra tem dois alvos e nenhum pré-requisito. Neste caso, a variável $@ irá conter apenas o alvo a ser gerado no momento.

Alvo .PHONY

A maioria dos Makefiles contém uma regra que permite limpar as directorias do projecto (eliminar ficheiros binários e .o, por exemplo). Habitualmente, esta regra chama-se clean. No entanto, o que acontece se existir um ficheiro clean na directoria? Como seria de esperar, a regra não será executada porque o seu alvo já existe. O make permite a criação de regras que são sempre executadas mesmo que o alvo exista. Para isso, devem ser definidas como dependências do alvo .PHONY (atenção ao ponto):

```
...
clean:
    @rm -f $(BINDIR)/*
    @rm -f $(OBJDIR)/*
.PHONY: clean
```

Como lidar com #includes

Esta secção do artigo é mais complexa que as anteriores e dirige-se a quem já tenha alguma experiência com sistemas *nix, e se sinta confortável com a linguagem C e compilação manual de programas. Para aqueles que ainda não estão nesse patamar, esta secção poderá ser um ponto de retorno no futuro, quando o nível de conhecimento for mais confortável.

Os programas escritos em C contêm muitas vezes ficheiros com a extensão .h, conhecidos como include files. Estes ficheiros servem de interface a outros ficheiros C e ajudam a fragmentar o nosso programa em partes lógicas.

A adição de ficheiros .h ao projecto coloca alguns problemas relacionados com a definição de dependências: embora possamos facilmente instruir o make a recompilar ficheiros .c cujo .h correspondente foi alterado, cada ficheiro .h pode ser dependência de qualquer ficheiro .c (não apenas do seu homónimo) ou até de outros ficheiros .h. No nosso projecto imaginário, tanto main.c como foo.c fazem #include de foo.h.

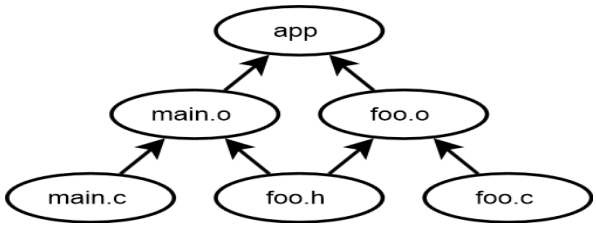


Figura 3: Grafo de dependências após inclusão de foo.h.

Como traduzir esta relação para o Makefile sem intervenção manual (Figura 3)?

Antes de mais, precisamos de conseguir obter a lista de ficheiros .h que são incluídos por um .c. O gcc (e clang) permite-nos fazer o seguinte:

```
$ gcc -MM -MG main.c
main.o: main.c foo.h
$ gcc -MM -MG foo.c
foo.o: foo.c foo.h
```

Ou seja, o gcc permite fazer output de uma regra para Makefiles com as dependências de um determinado ficheiro .c. Isto significa que podemos gerar um ficheiro auxiliar com as dependências para cada ficheiro .c no nosso projecto e incluí-lo no nosso Makefile através do comando include (e o make irá acrescentar as dependências importadas às definidas no Makefile, refinando-as).

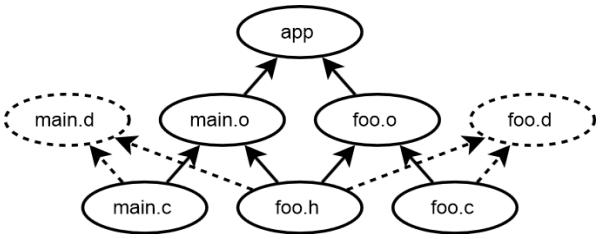


Figura 4: Grafo de dependências com ficheiros auxiliares .d.

O próximo passo exige alguma ginástica mental: vamos criar um ficheiro com a extensão .d (esta escolha de extensão é completamente arbitrária) para cada ficheiro .c no nosso projecto. Cada ficheiro .d irá conter as dependências necessárias para gerar um ficheiro .o a partir do ficheiro .c homónimo, utilizando o gcc, como acima referido. Além disso, o próprio ficheiro .d terá como dependências as mesmas dos ficheiros .o, obrigando a novo cálculo sempre que o código seja alterado (Figura 4).

Para tal, criaremos uma nova directoria .depend (definida pela variável DEPDIR) na qual serão colocados os ficheiros .d. Estes serão lidos pelo make através da instrução:

```
include $(DEPENDS)
```

A definição da variável DEPENDS é deixada como exercício para o leitor (dica: assemelha-se à definição da variável OBJECTS, com a diferença de que a directoria será definida pela variável DEPDIR e a extensão será .d). Esta instrução deverá ser colocada após a primeira regra (a que cria o executável principal), pois é essa que queremos que seja utilizada automaticamente (caso contrário, estaríamos a definir como primeira regra uma das definições de dependências que importamos).

Inicialmente, estes ficheiros .d não existirão, e como tal, será emitido um aviso pelo make. Podemos silenciá-lo prefixando a instrução include com um hífen: -include. Ao tentar importar ficheiros inexistentes, o make tentará criá-los, e por isso é necessário criar uma regra para que o make possa criá-los:

```
$(DEPDIR)/%.d: $(SRCDIR)/%.c | $(DEPDIR)
    gcc -MM -MG $< | sed 's!^\(.\+\).o:!$(DEPDIR)/
\1.d $(OBJDIR)/\1.o:!' > $@
```

O leitor sem grande prática num sistema com Linux ou equivalente estranhará esta linha. Mas passo a explicar: O gcc vai gerar a lista de dependências como visto acima, passar esse output ao programa sed, que o irá alterar ligeiramente, e finalmente escrever o resultado no alvo da regra. A parte complicada do comando (sed ...) encarrega-se de adicionar o ficheiro .d ao alvo e de acrescentar caminhos correctos. A parte final (> $@) é uma operação de redireccionamento de output típica da shell em *nix e neste caso envia o output para o alvo (variável automática $@). Fica fora do âmbito deste artigo explicar o funcionamento do sed e os redireccionamentos da shell, mas importa saber que a sua utilização nos permite converter o seguinte output do gcc:

```
main.o: src/main.c src/foo.h
```

para um output assim, e guardá-lo num ficheiro main.d na directoria .depend:

```
obj/main.o .depend/main.d: src/main.c src/
foo.h
```

Desta forma, se o ficheiro foo.h for alterado, tanto main.d como main.o estarão desactualizados, obrigando assim à geração dos mesmos através das regras definidas anteriormente.

Aqui fica o Makefile completo:

```
CC := gcc
CFLAGS := -Wall -Wextra
BINDIR := bin
OBJDIR := obj
SRCDIR := src
DEPDIR := .depend
SOURCES := $(wildcard $(SRCDIR)/*.c)
OBJECTS := $(patsubst $(SRCDIR)/%.c,
$(OBJDIR)/%.o, $(SOURCES))
DEPENDS := $(patsubst $(SRCDIR)/%.c,
$(DEPDIR)/%.d, $(SOURCES))

$(BINDIR)/app: $(OBJECTS) | $(BINDIR)
      @echo "A ligar $@"
      @$(CC) -o $@ $^

-include $(DEPENDS)

$(OBJDIR)/%.o: $(SRCDIR)/%.c | $(OBJDIR)
      @echo "A compilar $*.c"
      @$(CC) $(CFLAGS) -c -o $@ $<

$(DEPDIR)/%.d: $(SRCDIR)/%.c | $(DEPDIR)
      @$(CC) -MM -MG $< | sed 's!^\(.\+\).o:!
$(DEPDIR)/\1.d $(OBJDIR)/\1.o:!' > $@

$(DEPDIR) $(BINDIR) $(OBJDIR):
      @mkdir $@

clean:
      @rm -rf $(BINDIR)/*
      @rm -rf $(OBJDIR)/*
.PHONY: clean
```

Agora, sempre que um ficheiro do nosso projecto for alterado, seja ele um ficheiro .c ou .h, apenas os ficheiros relevantes serão recompilados, poupando assim tempo e preocupações ao programador. Outra vantagem do nosso Makefile é a sua capacidade de acrescentar novos ficheiros .c à sua lista sem intervenção manual do programador. Em suma, o que fizemos foi mover toda a lógica de determinação de dependências para um programa especializado nisso mesmo, o make, deixando o programador livre para fazer apenas uma coisa: escrever o seu programa, seguro de que será compilado da forma mais rápida e fiável possível.

Saber mais

Bastantes aspectos ficaram fora deste artigo por não serem, à partida, tão úteis a quem está a começar. No entanto, o leitor poderá investigar a seu gosto tudo o que o GNU make oferece no seu manual..

AUTOR

Escrito por António Pedro Cunha (pwseo)

Médico natural de Guimarães, formado na Universidade do Minho.

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

Website: http://pwseo.aloj.net

A PROGRAMAR

Interface Gráfica de Termómetro Usando Processing

SEO: Search Engine Optimization – Primeiros passos

Algoritmos de pathfinding

Profiling tools ! Usar ou não usar... Os 5 minutos que mudam a experiência!

Interface Gráfica de Termómetro Usando Processing

Introdução:

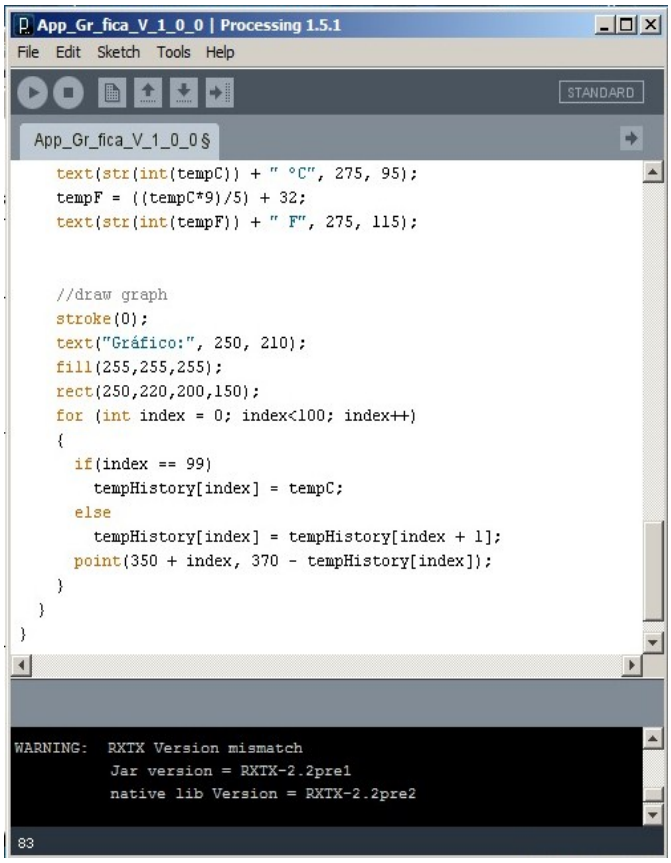
Este artigo foi escrito como continuação do artigo “Estação de Controlo de Temperaturas Usando Arduíno e Texas Instruments LM335A” publicado na edição anterior.

O objectivo deste novo artigo é criar uma interface gráfica que simplifique a leitura das temperaturas instantâneas lidas pelo sensor de temperatura Texas Instruments LM335A e interpretados pelo micro-controlador Arduino.

A interface criada é muito simples, assemelha-se a um termómetro comum e foi criada recorrendo à linguagem de programação Processing.

Processing:

Processing é uma linguagem de programação open-source aliada a um IDE que permite criar interfaces gráficas. Inicialmente a linguagem Processing destinava-se a ajudar aqueles que se iniciavam na arte de programar num contexto gráfico, sendo que actualmente esta linguagem de programação é usada desde estudantes até cientistas.



(Figura 1 – Processing)

A sintaxe desta linguagem de programação é em muito semelhante à linguagem C, pelo que é de fácil compreensão.

Existe muita informação e tutoriais sobre esta linguagem de programação e suas potencialidades espalhados pela internet, mas o website oficial possui bastante informação e possibilita o download do IDE: http://processing.org/

Código-Fonte para o Arduino:

Para que o Arduino possa ler e interpretar as informações lidas pelo sensor de temperatura LM335A é necessário usar o seguinte código-fonte:

```
//Declaração de Variáveis
int tempPin = 0;
float temperatura;
void setup()
{
  Serial.begin(9600);
}

void loop()
{
  //Leitura dos Valores do Sensor
  temperatura = analogRead(tempPin);
  //Conversão dos Valores Analógicos para ºC
  temperatura = ((((temperatura / 1023)*5)*100)-
273.15);
  //Envio de Dados para PC
  Serial.print((byte)temperatura);
  delay(1000);
}
```

O que este pedaço de código faz é ler os dados analógicos lidos no sensor de temperatura, convertê-los em dados digitais a cada segundo e enviar esses dados para o computador, recorrendo à conexão USB.

Carregar uma Imagem Com Processing

Na aplicação criada foi utilizada uma imagem de um termómetro em .JPG de modo a tornar a aplicação um pouco mais atractiva visualmente.

Para que possamos usar uma imagem de uma fonte externa na nossa aplicação, é necessário recorrer a uma série de passos. Inicialmente temos de guardar o nosso projecto numa pasta, e de seguida criar dentro dessa mesma pasta uma outra pasta com o nome “data” e então inserir a imagem dentro dessa pasta.

Após colocarmos a imagem que queremos na pasta “data”, vamos passar ao código necessário para que possamos exportar a imagem para a aplicação propriamente dita. O código necessário para efectuar este passo é o seguinte:

Após efectuar a criação da pasta “data”, inserir a respectiva

imagem dentro dessa pasta e, usando o código acima indicado, deverá então aparecer na janela da aplicação a imagem previamente seleccionada.

```
// Declarar a variável Pimage
PImage imagem;

void setup()
{

// Defenir o tamanho da Janela
size(500,500);

  // Criar uma nova Instância para a Variável PIma-
ge
  imagem = loadImage("termometro.jpg");
}

void draw()
{

//Definir a cor de Fundo da Aplicação
 background(255);

 // Desenhar a Imagem na Janela da Aplicação na
Coordenada (0,0)
 image(img,0,0);
}
```

Criar uma Aplicação Gráfica usando a Linguagem de Programação Processing

Agora que já sabemos como importar uma imagem para a nossa aplicação gráfica, vamos programar a restante aplicação de modo a estabelecer uma conexão entre os dados lidos do sensor de temperatura Texas Instruments LM335A e interpretados pelo micro-controlador Arduino e apresentá-los na janela da aplicação gráfica.

O código da pequena aplicação gráfica que desenvolvi, usando a linguagem de programação Processing, é o seguinte:

```
//Importar a Biblioteca Serial Communication
import processing.serial.*;

// Declaração de Variáveis
Serial commPort;
PImage imagem;
int yDist;
float tempC;
float tempF;
float[] tempHistorico = new float[100];

void setup()
 {

// Defenir o tamanho da Janela
 size(500,500);

// Carregar a Imagem do Termómetro
imagem = loadImage("img_termometro.jpg");

//Definir a cor de Fundo da Aplicação
background(255);

// Desenhar a Imagem na Janela da Aplicação na Co
//ordenada (0,0)
image (imagem, 0, 0);

  //Inicializar a Serial Communication Port
  commPort = new Serial(this, "COM3", 9600);

  //Preencher o Vector TempHistorico com 0
  for(int index = 0; index<100; index++)
{
   tempHistorico[index] = 0;
}

}

void draw()
{
  while (commPort.available() &rt; 0)
  {
    tempC = commPort.read();

    //Actualizar para Eliminar os Dados Antigos
    background(123);

    // Carregar a Imagem do Termómetro
    imagem = loadImage("img_termometro.jpg");
    background(255);
    image (imagem, 0, 0);

    //Desenhar o Trinângulo Apontador
    yDist = int(370 - (370 * (tempC * 0.01)));
    stroke(0);
    triangle(187, yDist + 10, 197, yDist + 5, 197,
                                          yDist + 15);
    fill(0,0,0);

    //Escrever os Valores de Referência
    fill(0,0,0);
    textAlign(RIGHT);
    text("212 F / 100 ºC", 145, 65);
    text("167 F / 75 ºC", 145, 140);
    text("122 F / 50 ºC", 145, 215);
    text("77 F / 25 ºC", 145, 290);
    text("32 F / 0 ºC", 145, 365);

    //Converter e Escrever as Temperaturas Actuais
    stroke(0);
    textAlign(LEFT);
    text("Temperatura Actual:", 250, 60);
    fill(255,255,255);
    rect(250,70,200,55);
    fill(0,0,0);
    text(str(int(tempC)) + " ºC", 275, 95);
    tempF = ((tempC*9)/5) + 32;
    text(str(int(tempF)) + " F", 275, 115);

    //Desenhar o Gráfico do Histórico
    stroke(0);
    text("Gráfico:", 250, 210);
    fill(255,255,255);
    rect(250,220,200,150);
    for (int index = 0; index<100; index++)
    {
      if(index == 99)
{
            tempHistorico[index] = tempC;
}
      Else
{
            tempHistorico[index] = tempHistorico
[index + 1];
          point(350 + index, 370 - tempHistorico
[index]);
}
    }

  }
```

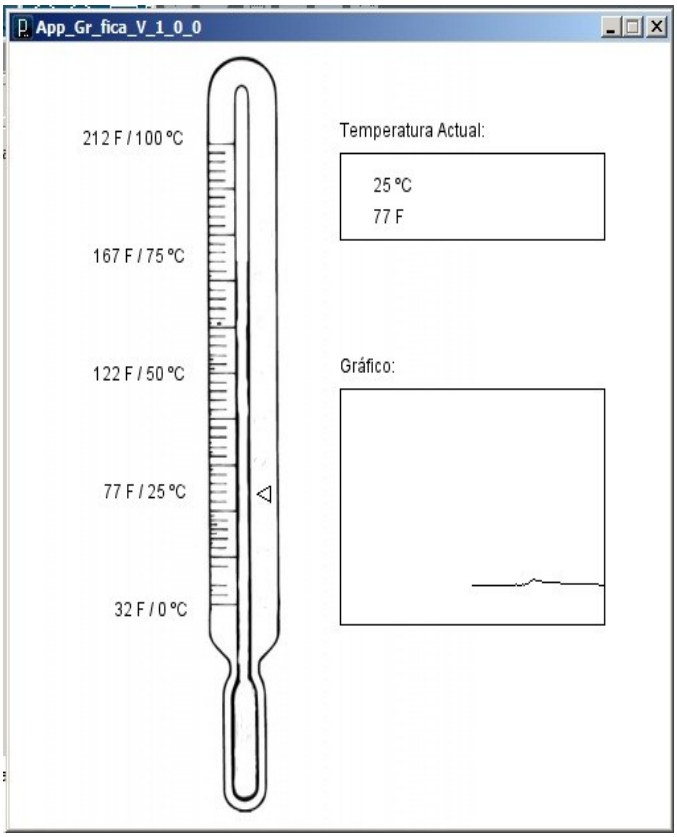


Figura 2 - Aplicação Gráfica)

A aplicação criada é visualmente muito simples, mas é suficiente para ser clara e fácil de ler. Como é possível constatar pelo código fonte acima colocado, a sintaxe da linguagem de programação Processing é muito semelhante à linguagem C, pelo que será simples de entender para quem já tenha alguma experiência em programação.

Uma particularidade da programação gráfica usando a linguagem de programação Processing é a necessidade de desenhar e ajustar as imagens carregadas para a aplicação por meio de programação de coordenadas no código fonte, o que por vezes pode atrasar e complicar um pouco a parte gráfica da aplicação.

Conclusão:

Como podemos constatar, criar uma aplicação gráfica capaz de interagir com o micro-controlador Arduino é relativamente simples, havendo muita informação, recursos e ferramentas disponíveis na Internet.

Para quem é adepto de interfaces gráficas, a linguagem de programação Processing é uma boa linguagem para criar pequenos projectos, facilitando a vida a quem já tiver alguma experiência com a linguagem de programação C, pois a sua sintaxe é muito semelhante.

AUTOR

Escrito por Nuno Santos

Curioso e autodidacta com uma grande paixão pela programação e robótica, frequenta o curso de Tecnologias de Informação e Comunicação na UTAD mas não esconde o sonho de ainda vir a ser Engenheiro Informático. Estudante, Blogger, e moderador no fórum Lusorobótica são algumas das suas actividades. Os seus projectos podem ser encontrados em: http://omundodaprogramacao.com/

SEO: Search Engine Optimization – Primeiros passos

Cá estamos de volta para mais um artigo sobre este fascinante tema que é o ***Search Engine Optimization* – SEO**. Para os que lêem este artigo pela primeira vez, e em especial os que desconhecem o tema, convido-os desde já para uma breve leitura das Edições n.º 34 e n.º 36 da **Programar**, mais concretamente os meus dois artigos que fazem a introdução ao conceito de SEO.

Agora que já estamos mais familiarizados com o SEO, iremos finalmente começar por falar mais a sério sobre o tema e colocar as mãos à obra. Ao escrever este terceiro artigo parto do princípio que já conhecem conceitos como ***PageRank***, **SERPs**, ***robots***, ***crawlers***, etc. Uma vez mais recomendo algum *background,* pois as técnicas de *Search Engine Optimization* são muitas e diversas, e apenas combinando bem estes tópicos é que conseguimos que a sua "magia" funcione.

Mas por onde começar?

Bem, como em tudo na vida, pelo princípio! No SEO também devemos começar pelo princípio, ou pelo menos quando isso ainda é possível, o que nem sempre acontece, como já irão perceber. Quando me refiro ao início, refiro-me claro está à construção de um site. O SEO é, como já vimos anteriormente, um conjunto de técnicas que aplicamos em websites visando a melhoria do seu posicionamento e visibilidade face ao Google e a potenciais visitantes oriundos dos seus resultados das pesquisas na internet. Nada melhor do que aplicar todas estas técnicas na génese do projecto, algo que nós Webdesigners e Consultores de SEO sabemos que infelizmente raramente acontece, pois os responsáveis pelos portais só "acordam" para esta necessidade quando os sites já estão em produção. Neste último caso apenas conseguimos intervir no que ainda é possível, embora fiquem descansados que ainda podemos resolver e optimizar um grande número de "variáveis" SEO. Tal como na construção de um edifício, onde se conseguirmos optimizar a sua eficiência energética quando o projecto ainda está no papel, poupamos imensos recursos e certamente que teremos que partir menos paredes. Perceberam a ideia? No SEO o problema é semelhante, se bem que neste caso as nossas paredes são páginas web e os tijolos são as linhas de código que lhes dão vida.

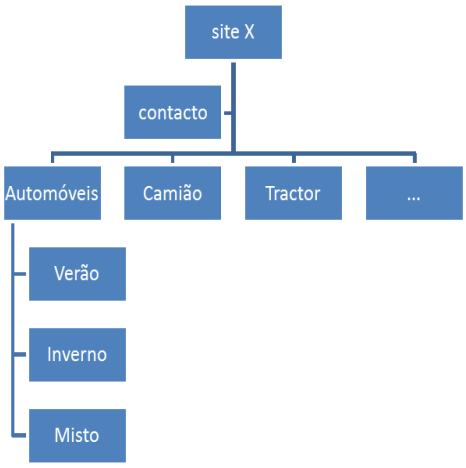
Isto tudo far-vos-á mais sentido se vos der exemplos concretos. Imaginem dois simples casos-cenário: No primeiro, iremos desenvolver um portal de vendas *online* de pneus, com tudo o que tem direito em termos de SEO; No segundo caso iremos optimizar um já existente, cujo domínio já fora escolhido pelo seu empresário, adoptando as iniciais do seu nome, ou as mesmas que constam no registo da sua empresa, a Miguel Lopes e Filhos Lda. (apenas a título de exemplo…não tenho nada contra este senhor, ficando: www.ml-filhos.pt. Acham mesmo que este domínio irá trazer muito "sumo" SEO para o site? Para quem procurar nos motores de busca por "filhos", talvez mais tarde venha a tropeçar muitas vezes neste site, mas ao visitá-lo, depressa se arrependerá, fazendo disparar a **Taxa de rejeições**, outro termo que iremos abordar nesta série de artigos, que muito simplisticamente significa que regista a percentagem de visitantes que depressa abandonaram o site, por não ser de todo o que procuravam. Sim, esta é uma taxa muito má! Tal como muitas outras taxas que por aí se anunciam, mas não entremos por aí agora…

Voltando agora ao primeiro cenário, aquele que nos possibilita desenvolver o portal desde o seu início, e onde por exemplo podemos escolher o seu nome do domínio. Já estão a ver onde quero chegar? É aqui que reside a grande vantagem de começar tudo do zero, pois mesmo a simples escolha de um nome de domínio, rico em *keywords* (palavras chave que descrevem o seu conteúdo), terá a prazo um enorme impacto no potencial SEO dos vossos sites. Diria até que a escolha do nome do domínio é um dos passos mais críticos para o sucesso da optimização SEO de um *website*.

Voltando então ao **cenário 1**, para uma loja de pneus convém adoptar um domínio que descreva da melhor forma o propósito do site: **vender pneus**. Não me querendo alongar muito nas *keywords*, até porque irei dedicar um artigo completo sobre elas, o que deveríamos de fazer em primeiro lugar seria analisar muito bem o foco da actividade comercial do nosso cliente. O consultor de SEO tem que, e em primeiro lugar, conhecer muito bem o negócio do seu cliente e a actividade ou o propósito do site que irá optimizar. Esta é uma das regras de ouro no SEO. Outra boa prática não menos importante, é a de conhecer o público-alvo. Não digo que isto signifique que, no caso particular de uma loja de peças auto, tenhamos que conhecer, quer seja pela sua medida ou especificação, cada pneu colocado à venda, mas temos que ter uma visão ampla do ramo, pois iremos necessitar desse conhecimento para orquestrar toda a nossa estratégia de SEO. E isto não tem apenas importância na escolha do domínio, mas em toda a estrutura portal, como vamos perceber de seguida.

Neste caso específico, ou em outros projectos de SEO, convém realizar uma série de entrevistas ao cliente que nos encomendou o serviço de optimização, visitar o armazém, as instalações, etc. Sejam meticulosos, exaustivos e até "chatos" se necessário! Falem com funcionários e clientes! Possuir um espírito pró-activo é meio caminho andado para o sucesso.

Imaginando que neste cenário fiz esse trabalho de casa, fiquei certamente a saber que uma loja de pneus os vende para vários modelos de viaturas: Automóvel; Camião; Tractor, Empilhador, etc. Só com este conhecimento básico já consigo montar uma estrutura simples para o meu site, no que diz respeito às suas categorias, que poderá ter a seguinte forma:



Reparem agora na "magia SEO" que poderá surgir de seguida. Ainda não escolhemos o nome do domínio e sabemos já que temos de escolher um nome que o enriqueça de *keywords*, mas que seja curto o suficiente para que não perca na vertente de usabilidade (outro factor importantíssimo que não podemos descurar) e resulte em termos de *marketing*. Por vezes as melhores soluções são as mais simples e por isso vou pegar na palavra "pneu", o foco essencial do negócio. Mas o que falta ponderar? A vertente do lado do utilizador que procura pneus para camião, nos motores de busca da internet! Ora colocando-nos no seu lugar, o que introduzimos no campo de pesquisa do motor de busca, quando procuramos por lojas de pneus? Talvez: "**pneus para camião**"...mas não podemos escolher este nome, pois se ficasse **www.pneus-para-camiao.pt**, (nota: caso algum destes exemplos já exista, é por mera coincidência – tento sempre escolher exemplos que ainda não existam, pelo menos na data da escrita do artigo) por exemplo, deixaríamos de fora os que procuram pneus para outros tipos de viatura. E que tal **www.pneus-para.pt**? Pode parecer uma ideia sem nexo, mas reparem que se pensarmos um pouco mais "longe", como pode resultar este domínio conjugado com a estrutura definida anteriormente: www.**pneus-para**.pt/**camiao**.html; www.**pneus-para**.pt/**tractor**.html; www.**pneus-para**.pt/**automovel**.html...etc. Perceberam a jogada? A estrutura do site, em conjunto com o nome rico em palavras-chave, gera frases que são muito próximas aos termos introduzidos no campo de pesquisa, maximizando a hipótese do motor de busca escolher o nosso site/página, e de o apresentar em primeiro lugar ao cibernauta.

Para além de possuirmos um nome de domínio rico em palavras-chave relativas ao nosso negócio, este domínio conjugado com a estrutura do site, traduz-se num aumento de potencial enorme no que diz respeito ao SEO das páginas. Agora convém também testar a vertente de usabilidade e de *marketing*. Será que o nome cativa? Bem, aqui fica ao critério de cada um. Nada como propor uma reunião com o departamento de *marketing* da empresa e realizar um bom *brainstorming*. Mas atenção, como consultores SEO, tentem sempre estar presentes e alertar os *marketeers* ou os decisores, para os prós e contras de cada opção escolhida. Pesem sempre bem as vantagens e desvantagens e escolham uma opção de compromisso. Existem ferramentas que nos ajudam na escolha das *keywords* com mais "peso" para os motores de busca, assim como as que possuem uma maior concorrência, mas deixarei este tópico para o artigo específico, pois iria alongar muito este. Depois de perceberem como tiramos partido dessas ferramentas para a escolha das melhores palavras-chave, ainda se tornará mais fácil escolher o nome de domínio para o vosso projecto.

Domínios e TLDs

Agora que já esclarecemos este ponto crítico no nosso processo de SEO, o da escolha do nome do domínio, convém também falar dos **TLDs** (*top-level domains*). Os TLDs são os domínios que estão no topo ao nível hierárquico do ***Domain Name System*** (DNS). São exemplos de TLDs o **.pt**; **.com**; **.co.uk**, etc.

SEO – Search Engine Optimization – Primeiros passos

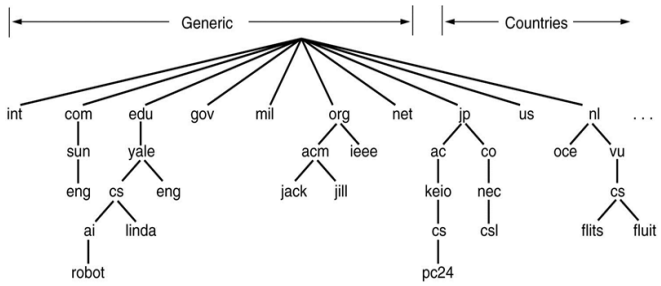


Ilustração 1 - TLDs (www.computing.dcu.ie)

Como podem visualizar na imagem anterior, estes podem dividir-se em dois grandes grupos, um relativo aos TLDs genéricos, e o outro aos TLDs que representam países. Porque vos falo neste tema que mais parece saído de uma aula de redes? Porque a escolha do nome do domínio implica a escolha de um TLD e este também tem importância para o SEO do nosso site.

Se pretendemos criar um site que tenha um público-alvo específico de um determinado país, devemos escolher o seu TLD respectivo, o que no caso de clientes residentes em **Portugal** seria o **.pt**. Se por outro lado pretendêssemos um site mais virado para um público inglês, a escolha mais acertada seria talvez o **.co.uk**. Digo talvez porque ainda temos o grupo dos TLDs genéricos e estes podem ocupar o lugar dos anteriores, se quisermos indicar aos motores de busca e potenciais clientes qual a natureza do nosso site, de acordo com a seguinte notação, que embora não seja obrigatória, deve ser respeitada:

TLDs genéricos (ou gTLD)	Descrição
.com	É o mais popular e generalista. Pode ser utilizado por qualquer tipo de site. Também indicado para sites de comércio.
.net	Também é generalista e costuma ser escolhido quando a opção .com já se encontra escolhida.
.org	Destinado a organizações, quer sejam com ou sem fins lucrativos ou até comerciais.
.edu	Muito utilizado pelas organizações de ensino, em especial as dos EUA.
.gov	Reservado a instituições governamentais
.biz	Utilizado por organizações orientadas a actividades comerciais e de negócios.

Existem ainda outros, como o **.info**, ou **.mil**, sendo que na internet poderão facilmente encontrar mais informação sobre assunto. Aqui a minha chamada de atenção vai para o facto de que ao contrário do que muitas pessoas pensam, existem mais opções do que o tradicional .com ou .pt, e que estas deverão ser escolhidas com parcimónia.

Por fim quero falar de outros aspectos a ter em conta, como o serviço de **alojamento *web*** a contratualizar, bem como a sua localização geográfica. Uma vez mais, devemos pensar a médio/longo prazo na escolha da empresa que nos irá fornecer o alojamento do nosso portal. Geralmente os decisores optam pelas soluções mais baratas, o que nem sempre será a opção mais acertada. Escolham sempre empresas que vos dêem garantias de fiabilidade e confiabilidade, bem como não podem ser descuradas as questões ao nível da performance do servidor e velocidade de acesso/latência da ligação ao mesmo. Um serviço barato pode por vezes esconder uma velocidade de acesso lenta, ou um servidor *web* "atascado" (*overselling*), traduzindo-se numa má experiência para o utilizador e numa possível penalização por parte dos motores de busca, como iremos também ver em artigos futuros, pois estes também medem a performance destas métricas anteriormente faladas.

Existem ainda outros, como o **.info**, ou **.mil**, sendo que na internet poderão facilmente encontrar mais informação sobre assunto. Aqui a minha chamada de atenção vai para o facto de que ao contrário do que muitas pessoas pensam, existem mais opções do que o tradicional .com ou .pt, e que estas deverão ser escolhidas com parcimónia.

Por fim quero falar de outros aspectos a ter em conta, como o serviço de **alojamento *web*** a contratualizar, bem como a sua localização geográfica. Uma vez mais, devemos pensar a médio/longo prazo na escolha da empresa que nos irá fornecer o alojamento do nosso portal.

Geralmente os decisores optam pelas soluções mais baratas, o que nem sempre será a opção mais acertada. Escolham sempre empresas que vos dêem garantias de fiabilidade e confiabilidade, bem como não podem ser descuradas as questões ao nível da performance do servidor e velocidade de acesso/latência da ligação ao mesmo. Um serviço barato pode por vezes esconder uma velocidade de acesso lenta, ou um servidor *web* "atascado" (*overselling*), traduzindo-se numa má experiência para o utilizador e numa possível penalização por parte dos motores de busca, como iremos também ver em artigos futuros, pois estes também medem a performance destas métricas anteriormente faladas.

Existe um último assunto para o qual vos quero alertar, já que falamos na escolha do serviço de alojamento *web*, que é o tipo de alojamento/servidor que vamos contratar. Desaconselham-se serviços partilhados (*shared hosting*). Estes servidores concentram em si várias outras contas e sites, de clientes diferentes, o que pode afectar tanto a performance como a reputação de um site. Um bom exemplo disso é o que acontece quando temos o azar de alojar um site no mesmo servidor onde estão alojados sites menos "fidedignos". Como partilhamos por exemplo o mesmo endereço IP, corremos o risco de "pagar o justo pelo pecador", podendo até num cenário extremo, ver o nosso site *blacklisted* ou os emails barrados por *spam* no destinatário.

Muito mais havia a esmiuçar sobre todos estes conceitos que hoje aqui vos falei. Como já perceberam o SEO é um tema tão vasto e interessante, que muitas páginas se podem escrever sobre ele. Muitas mais tenho para vos dar a conhecer, em próximas edições da Programar! Até lá, sempre que precisem de uma opinião ou de ajuda nalgum problema ou projecto, sintam-se livres de me contactar.

AUTOR

Escrito por Miguel Lobato

Licenciado em Tecnologias da Informação e Comunicação (TIC) e é Consultor de Search Engine Optimization – SEO e Administração de Sistemas. É também Webdesigner, Webdeveloper e um apaixonado por tudo o que é relacionado com as novas tecnologias.

https://www.facebook.com/MiguelLobatoSEO - @MiguelLobatoSEO

Algoritmos de pathfinding

Quem, como eu, gostar de vídeojogos de estratégia e similares, não pode deixar de pensar na quantidade e complexidade de algoritmos que estão por detrás da AI deste tipo de entretenimento. Vamos focar durante este artigo num tipo particular de algoritmos: ***pathfinding***. Na sua essência mais básica, o pathfinding pesquisa um *graph* começando num vértice e explorando os nodes adjacentes até encontrar o node de destino, sempre com o objectivo de encontrar o caminho mais curto entre o node de origem e o node de destino.

Introdução

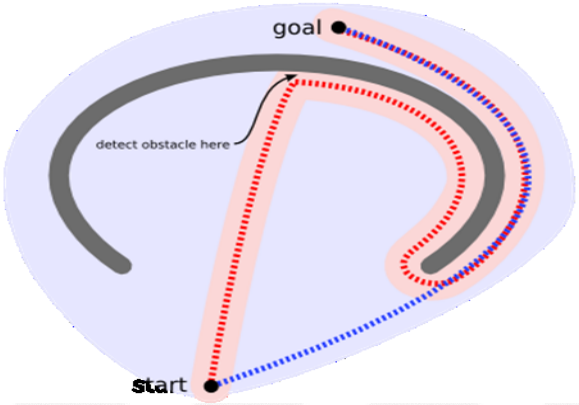
Pathfinding ou pathing refere-se à determinação, por uma aplicação informática, da rota mais curta entre dois pontos. É uma variante mais prática na resolução de labirintos. Esse campo de pesquisa é fortemente baseado no algoritmo de Dijkstra para encontrar o caminho mais curto em um graph ponderado.

Um graph é uma estrutura de dados comum, constituída por um conjunto de nós contendo dados e arestas que ligam os nós uns com os outros.

Pathfinding permite determinar se, e como, podemos chegar a um nó de outro. Além disso, podemos considerar não só se dois nós estão conectados, mas também atribuir algum tipo de custo para viajar entre eles. Podemos, então, procurar não só o caminho mais curto, mas também o mais barato, mais rápido, ou mais seguro.

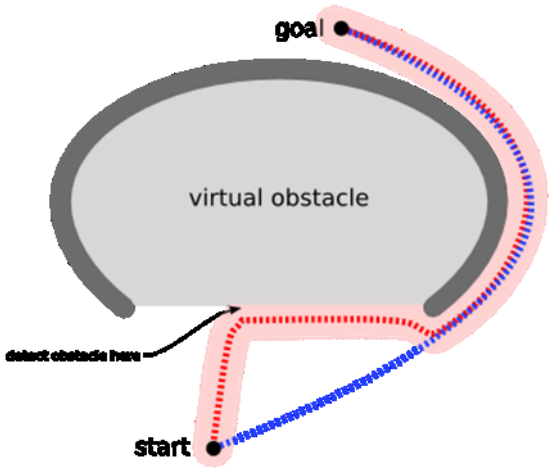
Determinar como nos deslocarmos num mapa é uma problemática interessante. Existem muitas soluções possíveis, desde a mais simples (irmos em frente até encontrarmos um obstáculo) até as mais complexas (algoritmos de pathfinding com heurísticas). Iremos começar por analisar os algoritmos mais simples, A*, até os mais atuais, navigation meshes para pathfinding.

Por que se preocupar com pathfinding? Considere a seguinte situação:



A unidade está inicialmente na parte inferior do mapa e quer chegar ao topo. Não há nada na área que verifica (mostrado em rosa) para indicar que a unidade não deve mover-se, por isso continua em seu caminho. Perto do topo, ele detecta um obstáculo e muda de direção. Em seguida, ele encontra o seu caminho em torno do obstáculo em forma de "U", seguindo o caminho vermelho. Em contraste, um desbravador teria verificado uma área maior (mostrado em azul claro), mas encontrou um caminho mais curto (azul), nunca enviando a unidade para o obstáculo de forma côncava.

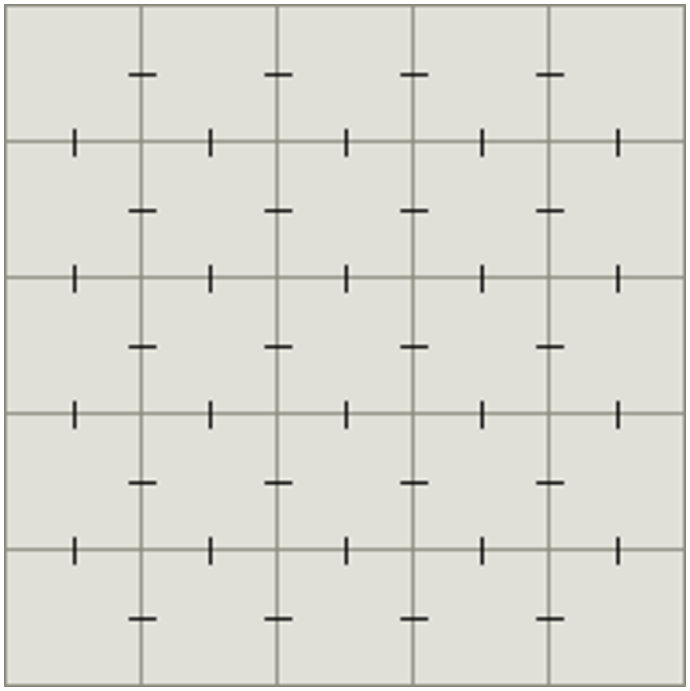
No entanto, pode estender um algoritmo de movimento para contornar armadilhas como a mostrada acima. Ou evitar a criação de obstáculos côncavas ou marcar o interior convexo como perigoso (a ser entrado somente se o objetivo é para dentro):



Pathfinders deixam você olhar em frente e fazer planos em vez de esperar até o último momento para descobrir que há um problema. Há uma troca entre o planejamento com pathfinders e reagir com algoritmos de movimento. Planejamento geralmente é mais lento, mas dá melhores resultados; movimento é geralmente mais rápido, mas pode ficar preso. Se o mundo do jogo está mudando, muitas vezes, o planejamento prévio é menos valioso. Eu recomendo usar ambos: pathfinding para a imagem grande, lento ao contornar obstáculos e caminhos longos; e movimento para a área local, em rápida mudança, e caminhos curtos.

Algoritmos

Os algoritmos de pathfinding de livros didáticos de ciência da computação trabalham em gráficos no sentido matemático, um conjunto de vértices com arestas que os conectam. Um mapa do jogo de azulejos pode ser considerado um gráfico com cada telha sendo um vértice e as bordas desenhada entre as telhas que são adjacentes uns aos outros:



A maior parte dos algoritmos de pathfinding ou algoritmos de pesquisa de AI são projetados para gráficos arbitrários em vez de jogos baseados em redes. Gostaria de encontrar algo que pode tirar proveito da natureza grade do mapa. Há algumas coisas que considero senso comum, mas que os algoritmos não entendem. Por exemplo, saber algo sobre direções: sei que, em geral, como duas coisas ficam mais distantes, ele vai demorar mais para passar de um para o outro, e sei que não há buracos secretos que permitam que você se teletransporte a partir de um ponto no mapa para o outro. (Bem, eu suponho que não há nenhum, se existem, torna-se muito difícil encontrar um bom caminho, porque você não sabe para onde olhar primeiro.)

Neste primeiro artigo irei abordar o algoritmo A *. A * é a escolha mais popular para pathfinding, porque é bastante flexível e pode ser usado numa grande variedade de contextos.

Este algoritmo pode parecer complicado para principiantes mas na realidade é bem simples de se entender e implementar.

Como linguagem de eleição irei utilizar o C++ nos nossos exemplos de implementação e sempre que for possível comentarei o código.

A *

O algoritmo A * tem três propriedades importantes:

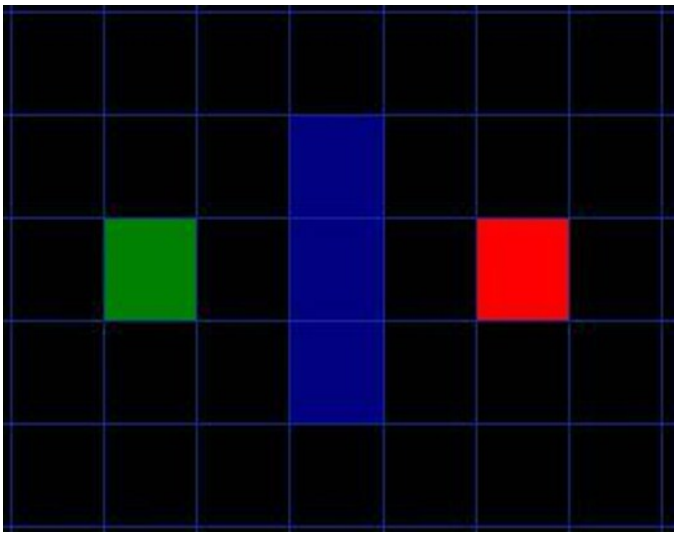
- Ele sempre devolverá o caminho menos caro se existe um caminho para o destino, outros algoritmos podem encontrar um caminho mais rápido, mas não é necessariamente o "melhor" caminho que pode levar;
- A * usa uma heurística (uma "suposição") para procurar os nós considerados mais susceptíveis de conduzir ao primeiro destino, permitindo-nos encontrar muitas vezes o melhor caminho sem ter que procurar o mapa inteiro e fazer o algoritmo muito mais rápido;
- A * é baseado na ideia de que cada nó possui um custo associado a ele. Se os custos para todos os nós são os mesmos, então o melhor caminho devolvido por A * será também o caminho mais curto, mas pode facilmente A * permitir adicionar custos diferentes para se mover através de cada nó.

A * cria duas listas de nós; uma lista fechada contendo todos os nós que temos totalmente explorados, e uma lista aberta contendo todos os nós que estamos trabalhando no momento (o perímetro da pesquisa).

Como o algoritmo é executado o valor F de um nó diz quão caro que acho que vai ser para alcançar o objetivo por meio desse nó.

A área de procura

Vamos assumir que temos alguém que quer ir do ponto A ao ponto B. Vamos supor que uma parede separa os dois pontos. Isto é ilustrado abaixo, com verde sendo o ponto de partida A, e vermelho sendo o ponto final B, e os quadrados azuis cheios sendo a parede entre ambos.



A primeira coisa que você deve observar é que dividimos a área de pesquisa em uma grade quadrada. Simplificar a área

de pesquisa, como fizemos aqui, é o primeiro passo em pathfinding. Este método particular reduz a área de pesquisa para uma matriz dimensional dois simples. Cada item na matriz representa um dos quadrados na grade, e seu estado é registrado como passável ou não-passável. O caminho é encontrado por descobrir quais quadrados que devemos tomar para ir de A para B. Uma vez que o caminho está encontrado, a nossa pessoa se move do centro de um quadrado até o centro do próximo até que o alvo é atingido.

Estes pontos centrais são chamados "nós". Quando você lê sobre pathfinding em outro lugar, muitas vezes você vai ver as pessoas discutindo nós. Porque não é só chamá-los quadrados? Porque é possível dividir a sua área de pathfinding em outra coisa que quadrados. Eles poderiam ser retângulos, hexágonos, triângulos, ou qualquer outra forma, realmente. E os nós podem ser colocados em qualquer lugar dentro das formas - no centro ou ao longo das bordas, ou em qualquer outro lugar. Nós estamos usando esse sistema, no entanto, porque é o mais simples.

Iniciando a procura

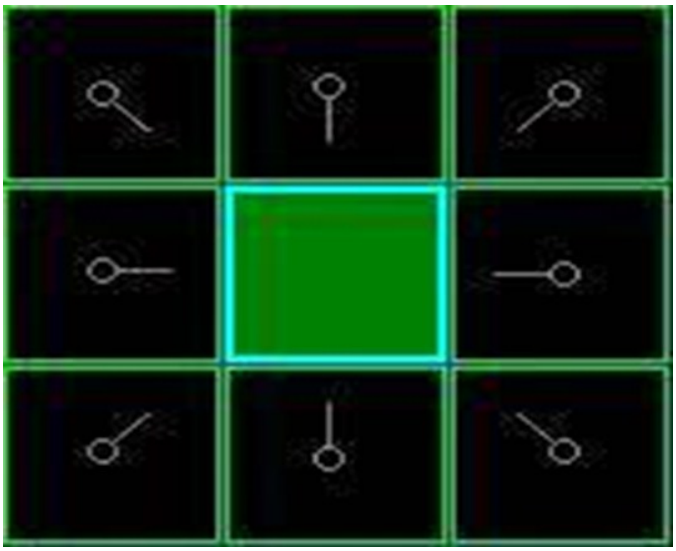
Uma vez que nós simplificamos a área de pesquisa em um número gerenciável de nós, como temos feito com o layout da grade acima, o próximo passo é a realização de uma pesquisa para encontrar o caminho mais curto. Fazemos isso a partir de um ponto, verificando os quadrados adjacentes, e geralmente pesquisando fora até que encontramos o nosso destino.

Começamos a busca fazendo o seguinte:

1. Comece no ponto de partida A e adicione-o a uma "lista aberta" de quadrados a serem considerados. A lista aberta é como uma espécie de lista de compras. Neste momento, há apenas um item na lista, mas mais tarde teremos mais. Ele contém quadrados que podem cair ao longo do caminho que pretende tomar, mas talvez não. Basicamente, esta é uma lista de quadrados que necessitam de ser verificados;
2. Olhe para todos os quadrados alcançáveis ou passáveis adjacentes ao ponto de partida, ignorando quadrados com paredes, água, ou outro terreno ilegal. Adicione-os à lista aberta, também. Para cada um desses quadrados, salvar o ponto A como seu "quadrado pai". Este material quadrado pai é importante quando queremos traçar o nosso caminho. Será explicado mais tarde;
3. Largue o quadrado inicial A da sua lista aberta, e adicione-o a uma "lista fechada" de quadrados que você não precisa novamente de olhar agora.

Neste ponto, você deve ter algo parecido com a ilustração abaixo. Nesta ilustração, o quadrado verde escuro no centro é o seu quadrado de partida. Ele é delineado em azul claro para indicar que o quadrado foi adicionado à lista fechada.

Todos os quadrados adjacentes estão agora na lista aberta de quadrados a serem verificados, e eles são descritos em verde claro. Cada um tem um ponteiro cinza que aponta de volta para seu pai, que é o quadrado de partida.



Em seguida, nós escolhemos um dos quadrados adjacentes na lista aberta e mais ou menos repetimos o processo anterior, como descrito abaixo. Mas qual quadrado nós escolhemos? Aquele com o menor custo de F.

Pontuação do caminho

A chave para determinar quais `F = G + H` quadrados a usar quando descobrir o caminho é a seguinte equação:

Cada nó terá três valores associados a ele, F, G e H. Cada nó também precisa estar ciente de seu pai, para que possa estabelecer a forma como chegar a esse nó.

- G - o custo exato para chegar a este nó a partir do nó de partida;
- H - o custo (heurística) estimado para chegar ao destino a partir daqui;

Nosso caminho é gerado repetidamente passando por nossa lista aberta e escolhendo o quadrado com a menor pontuação de F. Este processo será descrito em mais pormenor um pouco mais adiante no artigo. Primeiro, vamos olhar mais de perto como calculamos a equação.

Como descrito acima, G é o custo de movimento para mover -se do ponto de partida a um dado quadrado usando o caminho gerado para chegar. Neste exemplo, vamos designar um custo de 10 para cada quadrado horizontal ou vertical movido, e um custo de 14 para um movimento diagonal. Nós usamos estes números, porque a distância real a se mover na diagonal é a raiz quadrada de 2 (não se

assuste), ou cerca de 1,414 vezes o custo de se mover horizontalmente ou verticalmente. Nós usamos 10 e 14 por causa da simplicidade. A relação é aproximada, e evita ter que calcular raízes quadradas e evitamos decimais. Isto não é porque somos burros e não gostamos de matemática. Usando números inteiros como estes é muito mais rápido para o computador, também. Como você vai descobrir logo, pathfinding pode ser muito lento se você não usar atalhos como estes.

Uma vez que estamos calculando o custo de G ao longo de um caminho específico para um dado quadrado, a maneira para descobrir o custo G daquele quadrado é levar o custo de G de seu pai, e depois adicionar 10 ou 14, dependendo se ele é diagonal ou ortogonal (não-diagonal) do quadrado pai. A necessidade deste método irá tornar-se evidente um pouco mais adiante neste exemplo, quando ficamos mais de um quadrado de distância do quadrado de partida.

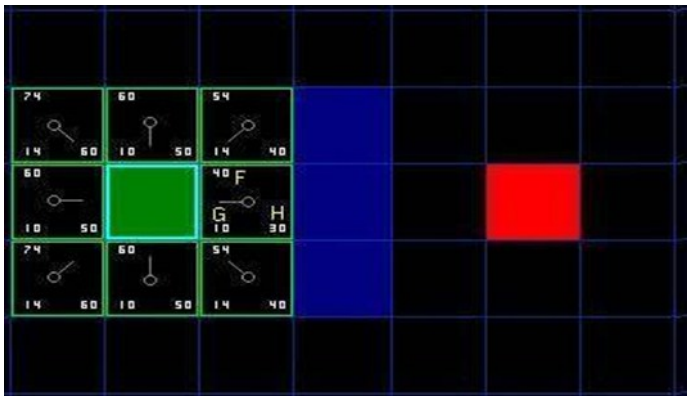
H pode ser estimado de várias maneiras. O método que usamos aqui é chamado de ***método de Manhattan***, onde você calcula o número total de quadrados movidos horizontalmente e verticalmente para alcançar o quadrado alvo do quadrado atual, ignorando movimento diagonal, e ignorando quaisquer obstáculos que possam estar no caminho. Em seguida, multiplica o total por 10, o nosso custo para mover um quadrado horizontalmente ou verticalmente. Este é (provavelmente) chamado o método de Manhattan porque é como calcular o número de blocos da cidade de um lugar para outro, onde você não pode atravessar o bloco na diagonal.

Lendo esta descrição, você pode imaginar que a heurística é apenas uma estimativa aproximada da distância restante entre o quadrado corrente e o alvo "em linha recta". Este não é o caso. Na verdade, estamos tentando estimar a distância restante ao longo do caminho (que normalmente é mais distante). Quanto mais perto a nossa estimativa é à distância restante real, o algoritmo mais rápido será. Se superestimar essa distância, no entanto, não é garantido nos dar o caminho mais curto. Em tais casos, temos o que é chamado de "heurística inadmissível".

Tecnicamente, neste exemplo, o método de Manhattan é inadmissível, pois ligeiramente superestima a distância restante. Mas vamos usá-lo de qualquer maneira, porque é muito mais fácil compreensível para os nossos propósitos, e porque é apenas uma ligeira superestimação. Nas raras ocasiões em que a trajetória resultante não é o mais curto possível, será quase tão curto.

F é calculado somando G e H. Os resultados do primeiro passo de nossa pesquisa pode ser visto na ilustração abaixo. O F, G, H e pontuações são escritas em cada quadrado. Como é indicado no quadrado imediatamente à direita do quadrado inicial, F é impresso no canto superior esquerdo, G é impresso no canto inferior esquerdo, e H é impresso na parte inferior direita.

Então, vamos olhar para alguns desses quadrados. No quadrado com as letras, G = 10. Isto é porque fica apenas



um quadrado do quadrado inicial em uma direção horizontal. Os quadrados imediatamente acima, abaixo e à esquerda do quadrado de partida têm todos a mesma pontuação de G 10. Os quadrados diagonais têm contagens de G, de 14.

As contagens de H são calculadas estimando a distância Manhattan ao quadrado alvo vermelho, movendo-se somente na horizontal e na vertical e ignorando a parede que se encontra no caminho. Usando este método, o quadrado imediatamente à direita do início é de 3 quadrados do quadrado vermelho para uma pontuação H de 30. O quadrado logo acima deste quadrado é de 4 quadrados de distância (lembre-se, apenas mova horizontalmente e verticalmente) para uma pontuação H de 40. Provavelmente, você pode ver como as contagens de H são calculadas para os outros quadrados.

A pontuação de F para cada quadrado, novamente, é simplesmente calculada somando G e H em conjunto.

Continuando a procura

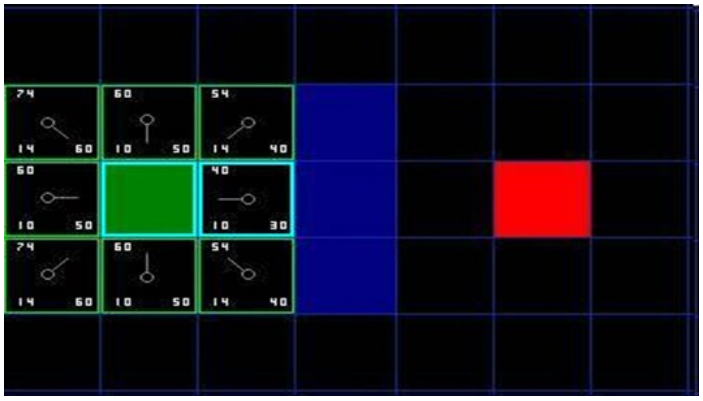
Para continuar a pesquisa, nós simplesmente escolhemos o quadrado com menor pontuação F de todos os que estão na lista aberta. Em seguida, faça o seguinte com o quadrado selecionado:

4. Largue-o da lista aberta e adicione-o à lista fechada;
5. Verifique todos os quadrados adjacentes. Ignorando aqueles que estão na lista fechada ou não-passável (terreno com paredes , água, ou outros terrenos ilegal), adicione quadrados à lista aberta, se eles não estão já na lista aberta. Faça o quadrado selecionado o "pai" dos novos quadrados;
6. Se um quadrado adjacente já está na lista aberta, verifique se este caminho para aquele quadrado é um melhor. Em outras palavras, verifique se a pontuação G para aquele quadrado é mais baixa, se usarmos o quadrado atual para chegar lá. Se não, não faça nada.

Por outro lado, se o custo de G do novo caminho é menor, mude o pai do quadrado adjacente ao quadrado seleccionado (no diagrama acima, alterar o sentido do ponteiro para apontar para o quadrado selecionado). Finalmente, recalcular tanto as notas F e G daquele quadrado. Se isso parece confuso, você vai ver que ilustrado abaixo.

Ok, então vamos ver como isso funciona. Dos nossos 9 quadrados iniciais, temos 8 restantes na lista aberta após o quadrado inicial ter sido transferido para a lista fechada. Destes, o único com o menor custo de F é o imediatamente à direita do quadrado inicial, com uma pontuação F de 40. Então, nós selecionamos este quadrado como o nosso próximo quadrado. É realçado em azul na ilustração a seguir.

Primeiro, vamos soltá-lo da nossa lista aberta e adicioná-lo à nossa lista fechada (é por isso que agora está destacado em azul). Então, vamos verificar os quadrados adjacentes. Bem, os imediatamente à direita são quadrados de parede, de modo que vamos ignorar aqueles. O imediatamente à esquerda é o quadrado de partida, que está na lista fechada, assim nós o ignoramos, também.

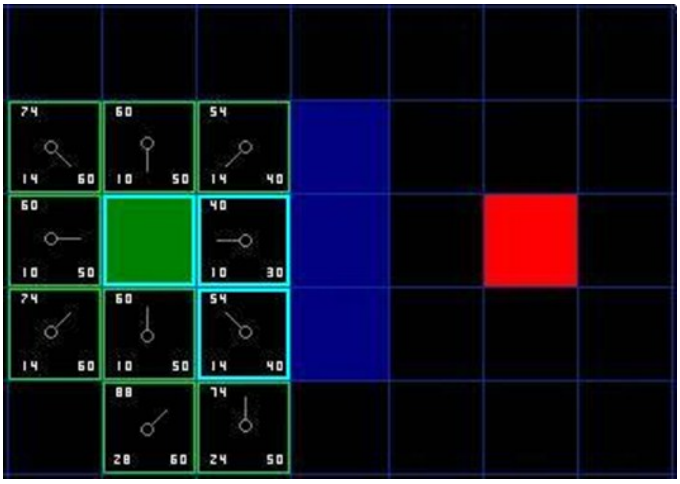


Os outros quatro quadrados já estão na lista aberta, por isso precisamos de verificar se os caminhos para esses quadrados são melhores usando este quadrado para chegar lá, usando contagens de G como nosso ponto de referência. Vamos olhar para quadrado logo acima do nosso quadrado selecionado. Sua pontuação G atual é 14. Se em vez disso passou pelo quadrado atual para chegar lá, a pontuação G seria igual a 20 (10, que é a contagem de G para chegar ao quadrado atual, mais 10 para ir verticalmente àquele um pouco acima dele). A pontuação G de 20 é maior que 14, então isso não é um caminho melhor. Isso deve fazer sentido se você olhar para o diagrama. É mais direto chegar a esse quadrado do quadrado inicial simplesmente movendo uma casa na diagonal para chegar lá, em vez de mover horizontalmente um quadrado, e depois verticalmente um quadrado.

Quando repetir esse processo para todos os 4 quadrados adjacentes já na lista aberta, descobrimos que nenhum dos caminhos são melhores, passando pelo quadrado atual, de modo que não muda nada. Portanto, agora que olhamos para todos os quadrados adjacentes, estamos terminados com este quadrado, e prontos para passar para o próximo quadrado.

Então vamos para a lista de quadrados em nossa lista aberta, que está agora em 7 quadrados, e escolher aquele com o menor custo de F. É interessante notar que, neste caso, existem dois quadrados com uma pontuação de 54. Então, qual escolher? Isso realmente não importa. Para efeitos de velocidade, pode ser mais rápido escolher o último que você adicionou à lista aberta.

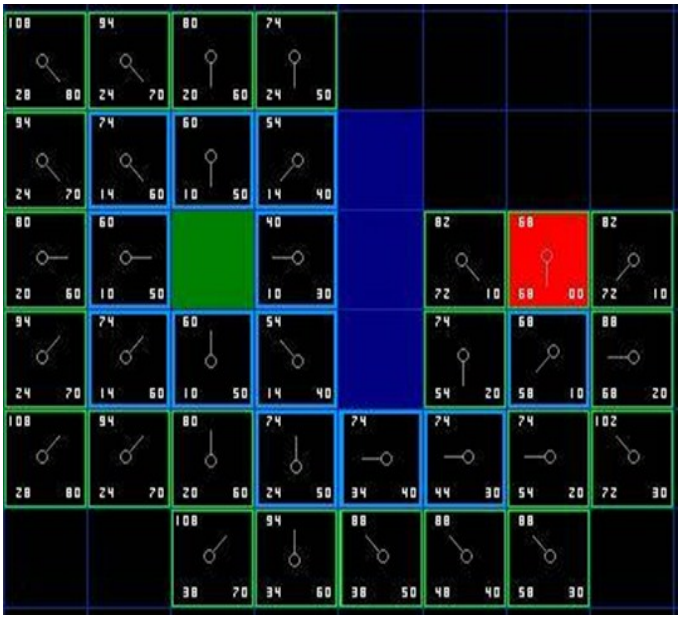
Portanto, vamos escolher o logo abaixo e para a direita do quadrado inicial, como é mostrado na figura a seguir.



Desta vez, quando verificamos os quadrados adjacentes, observamos que um imediatamente à direita é um quadrado de parede, assim nós o ignoramos. O mesmo vale para o que está acima. Nós também ignoramos o quadrado logo abaixo da parede. Por quê? Porque você não pode chegar a esse quadrado diretamente do quadrado atual sem cortar através do canto da parede próxima. Você realmente precisa ir para baixo primeiro e depois passar para esse quadrado, movendo-se em torno do canto no processo. (Nota: Esta regra de cantos de corte é opcional. O seu uso depende de como os nós são colocados)

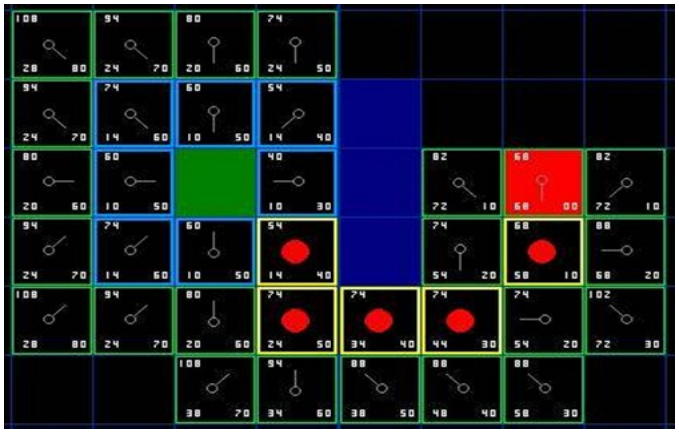
Isso deixa cinco outros quadrados. Os outros dois quadrados abaixo do quadrado atual ainda não estão na lista aberta, de modo que adiciona-mo-los e o quadrado atual se torna seu pai. Dos outros três quadrados, dois já estão na lista fechada (o quadrado inicial, e um pouco acima do quadrado atual, ambos destacados em azul no diagrama), de modo que ignora-mo-los. E o último quadrado, imediatamente à esquerda do quadrado atual, é conferido para ver se a pontuação G é mais baixa se você passar pelo quadrado atual para chegar lá. Nada feito. Então, está pronto para verificar o quadrado próximo na nossa lista aberta.

Nós repetimos este processo até adicionar o quadrado alvo à lista fechada, cujo ponto é algo como a ilustração abaixo.



Note que o quadrado pai para as duas casas abaixo do quadrado inicial mudou desde a figura anterior. Antes tinha uma pontuação G de 28 e apontou para o quadrado acima e à direita. Agora tem uma pontuação de 20 pontos e aponta para o quadrado logo acima dele. Isso aconteceu no meio do caminho em nossa busca, onde a pontuação G foi verificada e acabou por ser mais baixa usando um novo caminho - por isso o pai foi transferido e as contagens de G e F foram recalculadas. Enquanto esta mudança não parece muito importante, neste exemplo, há uma abundância de possíveis situações onde esta verificação constante vai fazer toda a diferença para determinar o melhor caminho para o seu destino.

Então, como vamos determinar o caminho? Simples, basta começar no quadrado alvo vermelho, e trabalhar para trás passando de um quadrado para o seu pai, seguindo as setas. Isto acabará por levá-lo de volta para o quadrado inicial, e esse é o seu caminho. Ele deve ser semelhante à ilustração a seguir. Se movendo do quadrado inicial A para o quadrado de destino B é simplesmente uma questão de se mover a partir do centro de cada quadrado (o nó) para o centro do próximo quadrado do caminho, até chegar ao destino.



Pseudo-code A *

```
create the open list of nodes, initially con-
taining only our starting node
   create the closed list of nodes, initially
empty
   while (we have not reached our goal) {
       consider the best node in the open list
(the node with the lowest f value)
       if (this node is the goal) {
           then we're done
       }
       else {
           move the current node to the closed
list and consider all of its neighbors
           for (each neighbor) {
               if (this neighbor is in the
closed list and our current g value is lower)
{
                   update the neighbor with
the new, lower, g value
                   change the neighbor's par-
ent to our current node
               }
               else if (this neighbor is in
the open list and our current g value is low-
er) {
                   update the neighbor with
the new, lower, g value
                   change the neighbor's par-
ent to our current node
               }
               else this neighbor is not in
either the open or closed list {
                   add the neighbor to the
open list and set its g value
               }
           }
       }
   }
```

http://www.boost.org/libs/graph

http://www.quarktet.com/PulseView.html

AUTOR

Escrito por João Ferreira

Autodidacta na área de programação e electrónica, comecei a programar em RM/Cobol. Com a massificação do Windows e interfaces gráficas interessei-me por C/C++, Uso como linguagem principal Pascal na sua vertente open-source (Lazarus). Programa Assembly na optimização de routinas e procedimentos de baixo nível. Na electrónica estou a divertir-me neste momento com o Raspberry-Pi e com outros projectos que estão em curso no HitSpace .

Profiling tools ! Usar ou não usar... Os 5 minutos que mudam a experiência!

Este artigo pretende apresentar as vantagens da utilização de ferramentas de Profiling no desenvolvimento de aplicações web em PHP, bem como justificar a afirmação feita no título para ele pensado: “Os 5 minutos que mudam a experiência!”.

Introdução

Frequentemente, ao desenvolvermos aplicações web, deparamo-nos com problemas de desempenho. No entanto, ao tentarmos cegamente resolver esses problemas, podemos não obter resultados concretos e estáveis. Assim, para auxiliar e optimizar o nosso trabalho, dispomos de uma técnica a que damos o nome de Profiling.

O Profiling consiste na utilização de ferramentas e técnicas que permitam obter dados sobre a execução do código, por exemplo: tempo de execução de cada função, linha, ou pedido, quanta memória utilizada, tempo de execução de queries, etc.

Desta maneira pode-se avaliar todo o desempenho de uma aplicação, identificar possíveis “bottlenecks” no código, tudo isto antes de colocar a aplicação em “produção”, permitindo assim uma optimização do código, para se obter a sua melhor performance da aplicação e se evitarem desperdícios de tempo tanto na execução como na identificação posterior dos “bottlenecks” e subsequente optimização do código.

Existem várias ferramentas para realizar estas tarefas. Neste artigo optei por utilizar a Xdebug, complementada por algu-

ENIGMAS DO C#: #: ASYNC/AWAIT E THREADS

por Paulo Morgado

Com o recente lançamento do Visual Studio 2012 e Framework .NET 4.5, foi introduzido na linguagem de programação C# duas novas palavras-chave (**async** e **await**) destinadas a programação assíncrona.

Com estas novas instruções (que tiram partido de funcionalidades já existentes na Task Parallel Library da Framework .NET 4.0 e em novas funcionalidades) a programação assíncrona torna-se muito mais simples e com aspeto de programação síncrona.

E para comemorar este lançamento, nesta edição temos um conjunto de enigmas relacionados com as palavras-chave **async** e **await**.

Enigma

Dado o seguinte programa:


```
static class Program
{
    class MyButton : Button
    {
        public MyButton(
            string text,
            EventHandler clickHandler)
        {
            this.Text = text;
            this.Margin = new Padding(4);
            this.AutoSize = true;
            this.AutoSizeMode
                    = AutoSizeMode.GrowAndShrink;
            this.Click += clickHandler;
        }
    }

    class TracingSynchronizationContext
        : SynchronizationContext
    {
        readonly SynchronizationContext sc;
        public TracingSynchronizationContext(
                    SynchronizationContext sc)
        {
            this.sc = sc;
        }
        public override void Post(
            SendOrPostCallback d,
            object state)
        {
            Console.WriteLine("Post");
            this.sc.Post(d, state);
        }
    }
    [STAThread]
    static void Main()
    {
        Application.EnableVisualStyles();
     Application
       .SetCompatibleTextRenderingDefault(false);

        Thread.CurrentThread.Name = "UI Thread";
```

```
        var panel = new FlowLayoutPanel
        {
            Dock = DockStyle.Fill,
            AutoSize = true,
            AutoSizeMode
                = AutoSizeMode.GrowAndShrink,
            FlowDirection
                = FlowDirection.LeftToRight,
            Controls =
            {
                new MyButton(
                    "1", Button1Click),
                new MyButton(
                    "2", Button2Click),
                new MyButton(
                    "3", Button3Click),
                new MyButton(
                    "4", Button4Click),
                new MyButton(
                    "5", Button5Click)
            }
        };
        var form = new Form
        {
            FormBorderStyle
                = FormBorderStyle.FixedToolWindow,
            AutoSize = true,
            AutoSizeMode
                = AutoSizeMode.GrowAndShrink,
            ClientSize = panel.Size,
            Controls = { panel }
        };
        form.Load += (s, e) =>
            SynchronizationContext
                .SetSynchronizationContext(
           new TracingSynchronizationContext(
           SynchronizationContext.Current));
        Application.Run(form);
    }
    static async void Button1Click(
        object sender,
        EventArgs e)
    {
        Console.WriteLine();
        Console.WriteLine(">>> 1");
        TraceThread();
        await DoWorkAsync();
        TraceThread();
        await DoWorkAsync();
        TraceThread();
        await DoWorkAsync();
        TraceThread();
        Console.WriteLine("<<< 1");
        Console.WriteLine();
    }
    static async void Button2Click(
        object sender,
        EventArgs e)
    {
        Console.WriteLine();
        Console.WriteLine(">>> 2");
        TraceThread();
        await SequenceAsync("2");
        TraceThread();
        Console.WriteLine("<<< 2");
        Console.WriteLine();
    }
    static async void Button3Click(
        object sender,
        EventArgs e)
    {
        Console.WriteLine();
        Console.WriteLine(">>> 3");
        TraceThread();
        await Sequence2Async("3");
        TraceThread();
        Console.WriteLine("<<< 3");
        Console.WriteLine();
    }
    static async void Button4Click(
        object sender,
        EventArgs e)
    {
        Console.WriteLine();
        Console.WriteLine(">>> 4");
        TraceThread();
        await Task.Factory.StartNew(
            () => SequenceAsync("4"),
            TaskCreationOptions.HideScheduler)
                                    .Unwrap();
        TraceThread();
        Console.WriteLine("<<< 4");
        Console.WriteLine();
    }
    static async void Button5Click(
        object sender,
        EventArgs e)
    {
        Console.WriteLine();
        Console.WriteLine(">>> 5");
        TraceThread();
        await Task.Run(
            () => Sequence2Async("5"));
        TraceThread();
        Console.WriteLine("<<< 5");
        Console.WriteLine();
    }
    static async Task SequenceAsync(
        string id)
    {
        Console.WriteLine(">>> {0}'", id);

        TraceThread();
        await DoWorkAsync();
        TraceThread();
        await DoWorkAsync();
        TraceThread();
        await DoWorkAsync();
        TraceThread();
        Console.WriteLine("<<< {0}'", id);
    }
    static async Task Sequence2Async(
        string id)
    {
        Console.WriteLine(">>> {0}'", id);
        TraceThread();
        await DoWorkAsync()
            .ConfigureAwait(false);
        TraceThread();
        await DoWorkAsync();
        TraceThread();
        await DoWorkAsync();
        TraceThread();
        Console.WriteLine("<<< {0}'", id);
    }
    static Task DoWorkAsync()
    {
        Console.WriteLine(
        "    DoWorkAsync()");
        return Task.Delay(1);
    }
    static void TraceThread()
    {
     Console.WriteLine(Thread.CurrentThread.Name
                                ?? "Worker Thread");
    }
}
```

Qual é o resultado de premir os botões 1, 2, 3, 4 e 5?

Veja a resposta e explicação na página 46

mas outras ferramentas sobre as quais vou falando ao longo do artigo.

Ambiente utilizado:

Sistema Operativo GNU/Linux, distribuição Ubuntu 10.04 LTS, a correr num PC comum, com Apache2 e respectivas mods e MySql.

Instalação:

Primeiramente instala-se o **Siege,** que é uma ferramenta complementar (ferramenta de teste de carregamento) que me permite ver quantos pedidos por segundo a minha aplicação recebe, e quanto tempo demora a responder.

Para instalar devo primeiramente fazer o download da ultima versão do Siege, para a home directory, através do endereço http://www.joedog.org/pub/siege/

Uma vez feito o download, executo o seguinte conjunto de comandos na consola, na minha home directory:

```
tar -xvf siege-latest.tar.gz
cd siege-2.70/
./configure
make
sudo make install
siege.config
```

Ao executar o siege.config, irá ser criado um ficheiro .siegerc. Posso editar este ficheiro para alterar o local onde será armazenado o ficheiro de log do siege. Caso não o faça, os ficheiros de log serão armazenados em /var/siege.log. É recomendável colocar os ficheiros de log a serem armazenados na nossa home directory, e altamente desaconselhável executar estas aplicações como root.

Existem outras configurações possíveis e interessantes que não serão abordadas, para não estender muito o artigo. Estarão sempre acessíveis no ficheiro .siegerc, onde podem ser activadas e desactivas

Uma vez instalado, utilizamos o seguinte comando, para executar o siege durante 60 segundos com 100 ligações simultâneas (simuladas):

```
$ siege -c 100 -b -t60s http://localhost
```

O output deverá parecer-se com isto:

```
** SIEGE 2.70
** Preparing 5 concurrent users for battle.
The server is now under siege...
Lifting the server siege.. done.
Transactions:      491 hits
Availability:      100.00 %
Elapsed time:      59.70 secs
Data transferred:  0.51 MB
Response time:     1.02 secs
Transaction rate:  4.81 trans/sec
Throughput:  0.01 MB/sec
Concurrency:       4.92
Successful transactions: 491
Failed transactions:    0
Longest transaction:    1.45
Shortest transaction:   0.82
```

O output do siege fica armazenado no ficheiro log, o que permite que se possa ir acompanhando a evolução do desenvolvimento, bem como perdas ou aumentos de performance sempre dentro de um ambiente de testes. Também se torna útil para se compararem resultados de testes, com mais ou menos utilizadores.

Com o siege instalado, avançamos para a instalação do Xdebug, que além de ser uma ferramenta de debug bastante poderosa, é também uma excelente ferramenta de profiling.

Para instalar o Xdebug, executa-se o seguinte código na consola:

```
sudo apt-get install php5-xdebug
```

Agora instalado o Xdebug, é necessário proceder à respectiva configuração editando o ficheiro xdebug.ini que se encontra em /etc/php5/conf.d/. Para o fazer basta na consola introduzir o seguinte comando:

```
sudo nano /etc/php5/conf.d/Xdebug.ini
```

Existem muitos outros editores de texto em modo de texto, como o Vi, Emacs, etc… Neste caso optei por utilizar o Nano, pela sua simplicidade.

No ficheiro Xdebug.ini insira as linhas que se seguem no seu final:

```
xdebug.show_mem_delta=On
xdebug.trace_format = 0
xdebug.auto_trace = On
xdebug.trace_output_dir = /tmp/tests
xdebug.trace_output_name = trace.%c.%p
xdebug.collect_params = 4
xdebug.collect_includes = On
xdebug.collect_return = On
xdebug.show_mem_delta = On
```

Esta configuração, como se pode ver, indica ao Xdebug que deve registar a stacktrace dos erros, define a localização onde deve armazenar os dados, bem como o número de parâmetros a armazenar, entre outras opções que poderão ser aprofundadas no próprio site da ferramenta Xdebug.

Terminando este passo, devemos reiniciar o httpd, neste caso o apache, com o seguinte comando na consola:

```
sudo /etc/init.d/apache2 restart
```

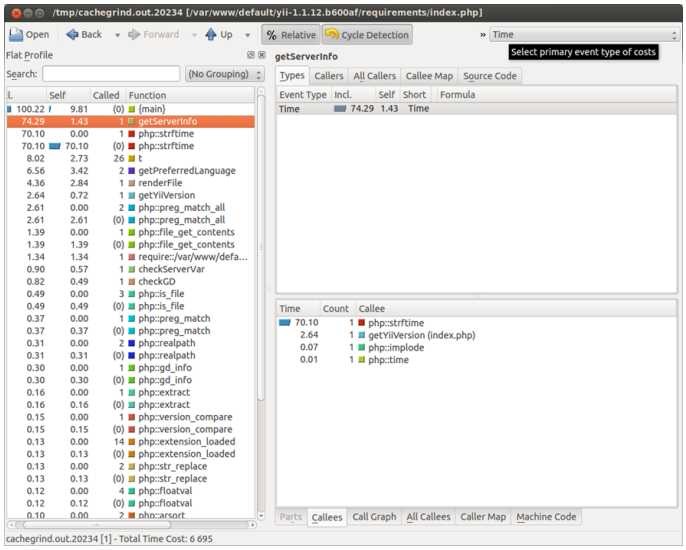
Com todas estas operações concluídas apenas falta instalar uma ferramenta que nos facilite a interpretação gráfica dos dados obtidos pelo Xdebug. Existem várias ferramentas gráficas para o fazer. Neste caso vamos usar o KCachegrind. Esta ferramenta pode ser instalada executando o seguinte comando na consola:

```
sudo apt-get install kcachegrind
```

Se preferir utilizar um ambiente web pode utilizar o Webgrind que é bastante semelhante. Existe ainda uma versão para Windows: o WinCacheGrind.

> **Os programadores devem utilizar ferramentas para analisar seus programas e identificar seções críticas de código.**

O KCacheGrind permite visualizar e personalizar a forma gráfica de apresentação dos dados de profiling produzidos pelo Xdebug. Desses dados destacam-se os relativos ao consumo de recursos de cada função, de cada ficheiro, de cada método (http Post e Get), de cada ligação à base de dados, e de cada query executada). - As funções que se encontrem em ficheiros que sejam chamados por require() , include() , require_once(), também são registados, podendo ser visualizados de forma gráfica, nos respectivos indicadores.



O uso desta ferramenta, dado o seu nível de personalização, é preferível em detrimento do desenvolvimento de uma ferramenta nova que faça o mesmo trabalho, pois seria repetição de trabalho.

Agora bastam apenas alguns minutos para se utilizar o Xdebug em todos os ficheiros, bastando para isso, no código PHP, inserir a linha:

Por exemplo:

```
xdebug_start_trace();
```

Conclusão:

```
<?php
xdebug_start_trace();
for ($i = 0; $i < 1000; $i++)
{
      echo '<p>Portugal-a-Prograr !</p>';
}
xdebug_stop_trace();
?>
```

Estas ferramentas e esta técnica permitem testar uma aplicação durante o seu desenvolvimento, evitando assim "dissabores" quando se coloca a aplicação ou website em produção. São 5 minutos que se gastam a configurar o conjunto de ferramentas, o qual não é explorado ao seu limite neste breve artigo, mas são horas que se ganham com a sua utilização, em tempo poupado entre identificação de "engarrafamentos" e sua resolução, tempo que se poupa ao não ter o site ou aplicação offline, e tempo que não se gasta a tentar empurrar as culpas para qualquer outro programa, que não a aplicação que desenvolvemos.

Atenção: O Xdebug é desaconselhado para uso em produção. Para esse efeito, existe uma outra ferramenta (XHProf), esta sim, adequada ao uso em servidores de produção.

Referências:

Documentação oficial do Xdebug (http://xdebug.org/)

IBM Developers Network - Profile your PHP application (http://ibm.co/d7PS0F)

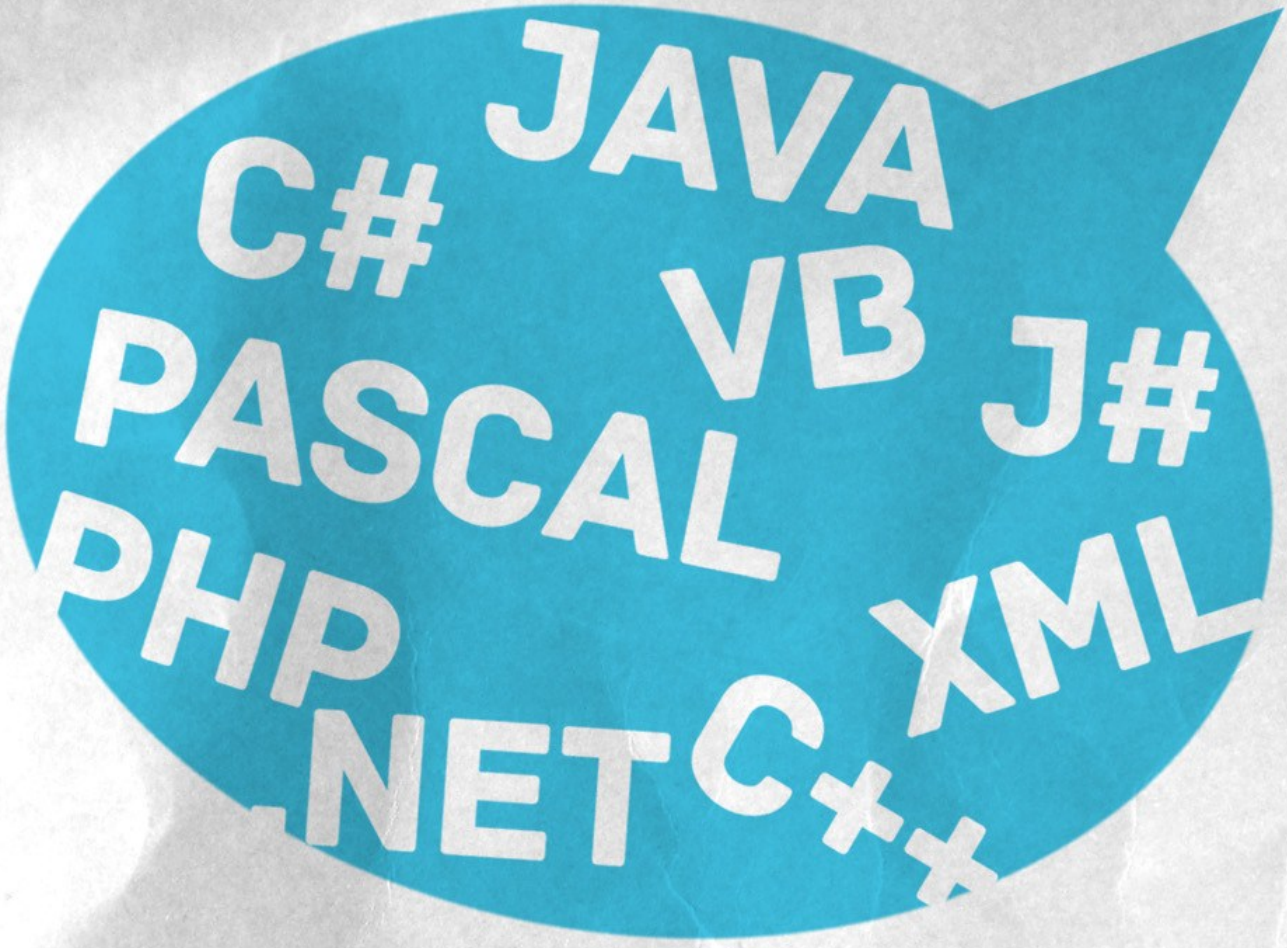
Php.net (http://www.bit.ly/WoTDJN)

AUTOR

Escrito por António Santos

Entusiasta da tecnologia desde tenra idade, cresceu com o ZX Spectrum, autodidacta, com uma enorme paixão por tecnologia, tem vasta experiência em implementação e integração de sistemas ERP, CRM, ERM. Membro da Comunidade Portugal-a-Programar desde Agosto de 2007, é também membro da Sahana Software Foundation, onde é Programador Voluntário. Neste momento é aluno no Instituto Politécnico de Viana do Castelo, na Escola Superior de Tecnologia e Gestão.

http://linkd.in/SqI3Dc



ENTÃO, SÓ FALAS EM CÓDIGO?

TEMOS O REMÉDIO CERTO PARA TI!

portugal-a-programar.pt

A MAIOR COMUNIDADE PORTUGUESA DE PROGRAMAÇÃO, APARECE!

COLUNAS

Visual (NOT) Basic - Organismos! Do zero ao mercado (1/2)

Enigmas de C# - Async/Await e Threads

CoreDump - Core Dump [8] - Fora de Horas

Kernel Panic - A importância da formação no ensino superior numa carreira dentro da área de segurança informática

Organismos! Do zero ao mercado (1/2)

Arregaçar as mangas

Agora que estiveram atentos às edições 34 e 35, onde aprenderam a fazer um jogo para PC recorrendo à XNA, vou tentar captar a vossa atenção novamente: o que me dizem de escrever um jogo para Windows Phone 7.5 ?

Esta é a primeira de duas partes, onde vamos focar essencialmente todos os elementos que orbitam em torno do núcleo do jogo, nomeadamente a ideia, os ecrãs de arranque e o menu. Para a segunda parte deixamos o ciclo do jogo, os pormenores e a colocação no mercado.

Como já deixei claro nos artigos anteriores, o meu principal objectivo não é fazer um jogo bom, ou o próximo best-seller do mercado. O meu objectivo é fazer passar a mensagem básica que vos permita fazer sentir que o vosso jogo de sonho não está assim tão distante e que o podem começar já hoje, agora.

Vou assumir muita coisa. Vou assumir principalmente que o leitor acompanhou edições anteriores e tem algumas noções daquilo que constituí uma aplicação Windows Phone. Poderei saltar passos básicos, ou que já tenham sido bem acompanhados nos últimos artigos de XNA. Poderei não incluir alguns detalhes finais, mas o ZIP que acompanha o artigo terá sempre uma versão completa e funcional. Os ficheiros de solução foram gravados com o Microsoft Visual Studio 2010 Express for Windows Phone.

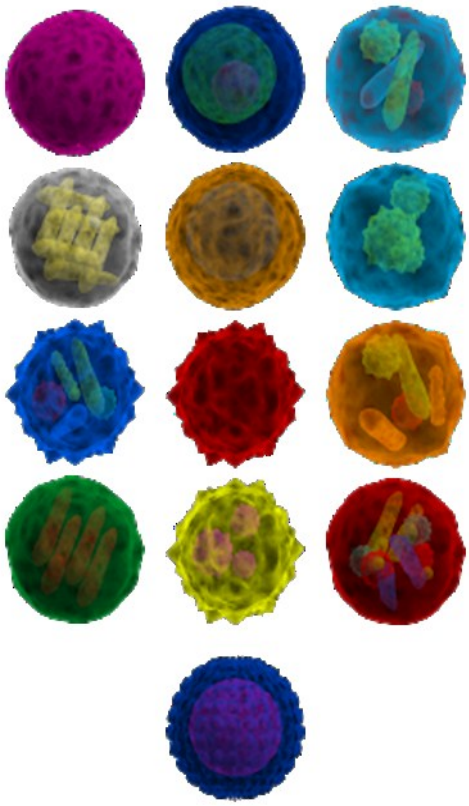
Vão reparar que os poucos bocados de texto do jogo estão em Inglês. Como se trata de um jogo que acompanha um artigo de uma revista de alcance internacional, só faria sentido preparar o jogo para ser lançado em todos os mercados.

Delinear o básico

A plataforma alvo de um jogo pode ser aquilo que determina o seu sucesso ou o que o condena de início. Nem todos os tipos de jogos funcionam em todas as plataformas, mas há certamente formas criativas de dar a volta. Já que estamos a lidar com um jogo para smartphone, optei por vos mostrar uma forma de aproveitar as duas interacções mais naturais: o toque e o movimento.

O jogo consiste em pequenos organismos, onde combinamos timing e perícia para nos desviarmos dos organismos maiores, e "comermos" os organismos mais pequenos. À medida que se vão "comendo" mais organismos, vamos crescendo, até ao ponto de ser ridículo, e passarmos de fase.
O objectivo do jogo é a pontuação. Não existem vidas. Apenas penalidades de pontuação e tamanho. O jogador que conseguir a pontuação mais alta até se fartar… está a ganhar. É simples como isso.

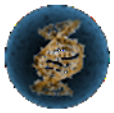
O organismo do jogador encontra-se sempre ao centro. A inclinação do smarpthone dita a direcção e velocidade dos restantes organismos, o que confere uma ilusão de movimento com o mínimo de operações de acompanhamento de um pivot (o jogador).

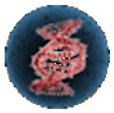


Todos os organismos intervenientes

No menu, serão usados os mesmos organismos sobre a influência do acelerómetro, para enfeite.

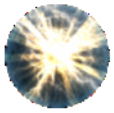
Para além dos organismos, podem aparecer esporadicamente bolhas de efeitos, que generalizei "PowerUps", mesmo que nem todas surtam efeitos positivos.
As bolhas não obedecem à regra do tamanho, e podem ser "comidas" a qualquer altura.
Os efeitos das bolhas não são acumuláveis. Um efeito anula o anterior.


Inversão genética

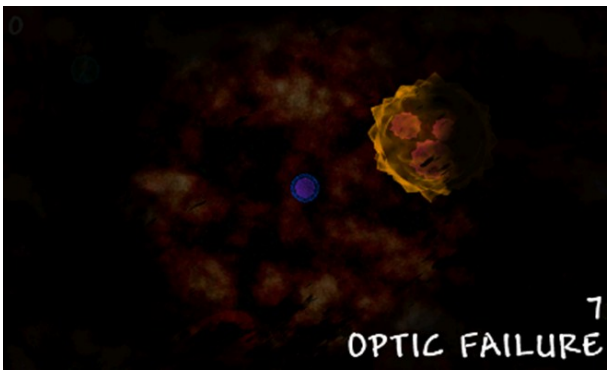

Agitação genética

Dobro dos pontos

Triplo dos pontos

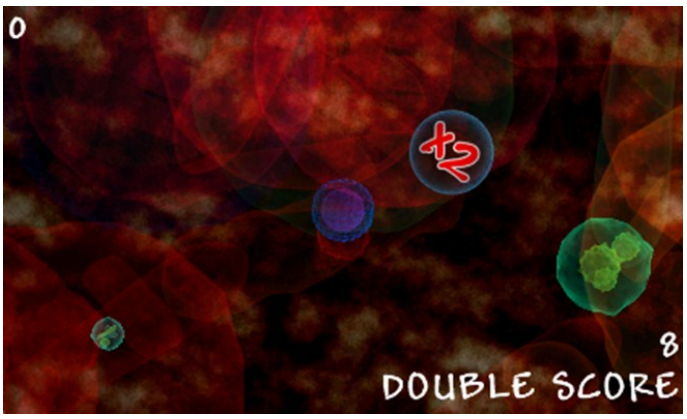

Falha óptica

Inversão genética faz com que durante o seu efeito se inverta a lógica do jogo, podendo assim "comer" os organismos maiores que nós, mas os pequenos não.
A agitação genética multiplica o movimento de todos os organismos, o que se pode traduzir num aumento da sensibilidade do acelerómetro. Dobro e triplo explicam-se por eles e por último, a falha óptica, trata-se de um obstáculo visual que reduz a nossa capacidade de ver o que nos rodeia.

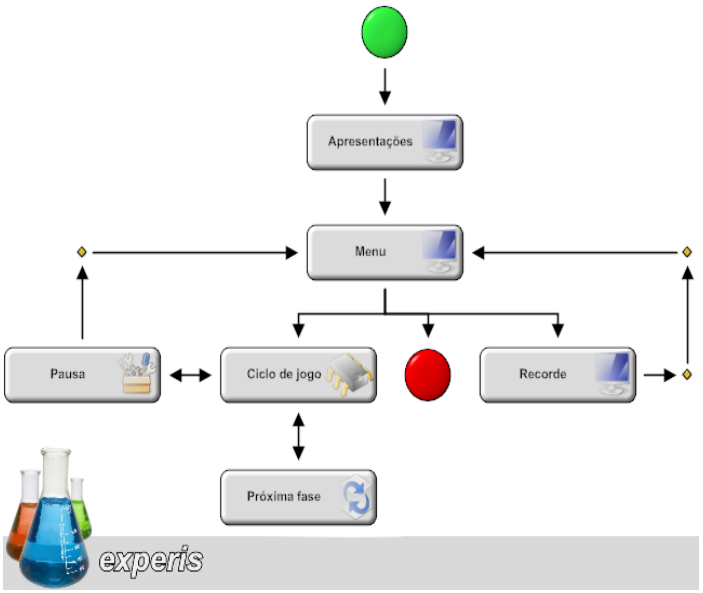


A informação permanente no ecrã será mínima. Vamos apenas manter a pontuação a um canto. Quando se está sob o efeito de uma bolha, aparecerá num outro canto o nome do efeito em vigor bem como o tempo que resta para terminar, em segundos.

Por fim, quando estivermos perigosamente próximo de um organismo que não vamos conseguir "comer", surge um símbolo de aviso num canto inferior.



O fluxo do jogo pode-se traduzir no seguinte:



Preparar recursos

Existem várias abordagens no que toca a organizar e estruturar o código do jogo. O método mais fácil, mas também errado, é escrever toda a lógica na classe principal, neste caso a classe Jogo. Para não complicar o conceito e manter as coisas simples, vou adoptar uma abordagem semelhante à que adoptei no anterior jogo de XNA: vamos criar várias classes de cada categoria, mas declaradas como "Partial", de forma a estarmos a trabalhar na mesma classe, mas com um pouco mais organização.

Comecemos por criar as classes:

Jogo.Actualizacoes
Contém todo o código relacionado com chamadas do Update.

Jogo.Desenho
Contém todo o código relacionado com chamadas do Draw.

Jogo.Fluxo
Contém a lógica para mostrar as imagens de apresentação iniciais nos tempos correctos e algumas verificações temporais.

Jogo.Inicializacoes
Contém todos os métodos que podem ser utilizados para produzir ou inicializar objectos.

Jogo.Recursos
Contém todos os métodos que carregam conteúdos da pipeline para memória bem como as respectivas colecções.

Central
Contém todas as referências ou recursos globais que queremos que estejam acessíveis em qualquer âmbito.

Utilidades
Contém todos os métodos que realizam pequenos cálculos recorrentes ou fracamente relacionados com o núcleo do jogo.

Objectos\Objecto
A unidade das entidades do jogo. Trata-se de uma classe não instanciável que compila todas as características comuns a todas as entidades do jogo.

Objectos\Organismo
Representa uma entidade organismo, que herda do objecto. Em termos práticos implementa apenas uma propriedade adicional, o Tamanho do organismo.

Objectos\Sprite
Uma camada de âmbito menos alargado sobre a unidade Objecto usada para representar elementos gráficos orientáveis, como imagens ou elementos de menu

Objectos\Som
Identificador de um som através de um par de chave/som. É usado para identificar vários SoundEffect diferentes sobre uma mesma chave, para possibilitar por exemplo a reprodução de um som aleatório para uma mesma acção.

Código, código, código

A primeira coisa que vamos fazer é dar umas directivas importantes quando a classe do Jogo é instanciada.

Ficheiro **Jogo.vb**

```
Public Sub New()
        graphics = New GraphicsDeviceManager(Me)
        graphics.IsFullScreen = True
        graphics.SupportedOrientations = Dis-
playOrientation.LandscapeLeft
        PhoneApplicationServi-
ce.Current.UserIdleDetectionMode = IdleDetec-
tionMode.Disabled
        Content.RootDirectory = "Content"
        TargetElapsedTime = TimeSpan.FromTicks
(333333)
        InactiveSleepTime = TimeSpan.FromSeconds(1)
    End Sub
```

Primeiro indicamos ao gestor de gráficos que queremos a aplicação em ecrã completo e que a orientação será única e exclusivamente “LandscapeLeft”. Poderíamos inverter os eixos do acelerómetro para suportar “LandscapeRight”, mas vamos manter isto simples e limitar a rotação apenas ao LandscapeLeft, que obtêm inclinando o smarpthone para a esquerda, até deitar.

A linha seguinte é extremamente importante. Como se trata de um jogo que usa em 95% do tempo o acelerómetro, não existe interacção humana, por assim dizer. Não há toques no ecrã nem em nenhuma tecla física. Independentemente do acelerómetro estar a variar, ou não, para o sistema operativo isto tem a mesma validade de deixar o telefone em cima da mesa e ir embora: eventualmente, o ecrã desliga e tranca. Isto não é bom para o jogo.

Ao desactivarmos o IdleDetection, estamos a indicar que não queremos que o sistema operativo detecte se o telefone está ao abandono e que se desligue: queremos que se mantenha ligado sempre.

De seguida, vamos já preparar a nossa classe central com todas as variáveis que prevemos ser necessárias. Neste caso, vamos já colocar todas as variáveis a usar, mesmo que não sejam ainda usadas nesta parte. Explicarei mais tarde.

Ficheiro **Central.vb**

```
Imports Organisms.Utilidades
Imports Microsoft.Advertising.Mobile.Xna

Public Class Central
    Private Shared _leituraAccel As Vector3 =
                                    Vector3.Zero
    Public Shared PublicidadeAC As AdGameComponent
    Public Shared PublicidadeDA As DrawableAd
    Public Shared FundoJogo1 As Sprite
    Public Shared FundoJogo2 As Sprite
    Public Shared Letras As SpriteFont
    Public Shared EcraIntro_A As Sprite
    Public Shared EcraIntro_B As Sprite
    Public Shared Menu As Sprite
    Public Shared Botao_Jogar As Sprite
    Public Shared Botao_Recorde As Sprite
    Public Shared Botao_Sair As Sprite
    Public Shared Botao_Menu As Sprite
    Public Shared Botao_Continuar As Sprite
    Public Shared Jogador As Organismo
    Public Shared CAMADA_INFERIOR As New List(Of
                                        Objecto)
    Public Shared CAMADA_EFECTIVA As New List(Of
                                        Objecto)
    Public Shared CAMADA_SUPERIOR As New List(Of
                                        Objecto)
    Public Shared POWERUPS_EFECTIVOS As New List(Of
                                        Objecto)
    Public Shared PowerUpActivo As Powerup
    Public Shared MascaraPU As Sprite
    Public Shared ORGANISMOSMENU As New List(Of
                                        Objecto)
    Public Shared AleatorioGlobal As New Random
    Public Shared Nivel As Integer = 1
```

```
    Public Shared Pontos As Long = 0
    Public Shared PontuacaoMaxima As Long
    Public Shared PontuacaoMaximaGlobal As String
    Public Shared Alarme As Boolean = False
    Public Shared Safo As Boolean = False
    Public Shared OffsetAcelerometro As Single =
                                            0.5

    Public Enum PowerUps
        INVERSAOGENETICA
        DOBRO
        TRIPLO
        AGITACAOGENETICA
        FALHAOPTICA
    End Enum

    Private Shared WithEvents ACCEL As New
            Microsoft.Devices.Sensors.Accelerometer

    Public Shared ReadOnly Property LeituraAcel As
                                            Vector3
        Get
            Dim EixoX As Single = 0.4
            EixoX += _leituraAccel.X

            Return New Vector3(Grampo(EixoX, -0.6F,
                    0.6F), Grampo(_leituraAccel.Y,
                                -0.6F, 0.6F), 0)
        End Get
    End Property

    Private Shared Sub ACCEL_CurrentValueChanged
    (sender As Object, e As
      Microsoft.Devices.Sensors.
                            SensorReadingEventArgs
                    (Of Microsoft.Devices.Sensors.
                            AccelerometerReading))
 Handles ACCEL.CurrentValueChanged
        _leituraAccel =
       e.SensorReading.Acceleration
    End Sub

    Public Shared Sub PararLeituraAcelerometro()
        ACCEL.Stop()
    End Sub

    Public Shared Sub IniciarLeituraAcelerometro()
        ACCEL.TimeBetweenUpdates =
        _TimeSpan.FromMilliseconds(30)
        ACCEL.Start()
    End Sub

End Class
```

São necessários imports e referências a Advertising.Mobile.Xna e também a Devices.Sensors para ganharmos acesso ao acelerómetro.

Todas as variáveis são declaradas como “Shared” para acesso central, sem instância e cada uma representa uma entidade do jogo, quer sejam os botões do menu ou o próprio jogador.As colecções são conjuntos de powerups ou organismos, pois como são sempre de número variável, e cada um possuí as suas propriedades, fazemos ocorrer as mesmas influências sobre cada um, em lote. As camadas inferior, superior e efectiva serão explicadas mais tarde pois pertencem ao ciclo de jogo. A colecção ORGANISMOSMENU vai ser usada para afectar em lote todos os organismos que se poderão ver no menu. Avista-se no meio a Enum para identificação dos tipos de powerup disponíveis e seguem-se algumas variáveis de controlo, que serão explicadas quando forem utilizadas.

Por fim seguem-se um conjunto de métodos e membros destinados a activar/desactivar o acelerómetro, bem como a fazer as leituras dos eixos.

De volta à classe do Jogo, passamos para o método de inicialização, que vamos aproveitar para inicializar um módulo de publicidade e ler a pontuação pessoal máxima, armazenada no dispositivo, se alguma.

Ficheiro **Jogo.vb**

```
    Protected Overrides Sub Initialize()
        InicializarPublicidade()
        CarregarPontuacaoPessoal()
        MyBase.Initialize()
    End Sub
```

O método InicializarPublicidade encontra-se nas inicializações:

Ficheiro **Jogo.Inicializacoes.vb**

```
    Public Sub InicializarPublicidade()
        AdGameComponent.Initialize(Me, "APPID")
        Central.PublicidadeAC =
                            AdGameComponent.Current
        Components.Add(Central.PublicidadeAC)
        Central.PublicidadeAC.Visible = False
        Central.PublicidadeDA =
           Central.PublicidadeAC.CreateAd("PUBID",
                  New Rectangle(160, 300, 480, 80))
    End Sub
```

Inicializamos um componente de jogo, como indicado na documentação da Microsoft Advertising e fornecemos as referências do que acabámos de criar à classe Central. Onde se lê APPID, deverá ser substituído por o APPID que vos é fornecido quando se registam em https://pubcenter.microsoft.com/ que é necessário para que as vossas unidades de publicidade funcionem e rentabilizem. O mesmo se aplica ao PUBID, mas refere-se ao ID de uma unidade de publicidade em específico, pois é possível registar várias unidades para a mesma aplicação.

O método CarregarPontuacaoPessoal, por se tratar de uma operação mais ligada ao dispositivo, foi passada para a classe Utilidades:

Ficheiro **Utilidades.vb**

```
    Public Shared Sub CarregarPontuacaoPessoal()
        Dim IST As IsolatedStorageFile = Isolated-
StorageFile.GetUserStoreForApplication()
        If Not IST.FileExists("organismos.pontos")
Then Central.PontuacaoMaxima = 0 : Exit Sub
        Using rF As New IO.StreamReader(New Isolat-
edStorageFileStream("organismos.pontos", File-
Mode.OpenOrCreate, FileAccess.Read, IST))
            Central.PontuacaoMaxima = Long.Parse
(rF.ReadToEnd())
        End Using
    End Sub
```

Organismos! Do zero ao mercado (1/2)

A lógica é simples: se o ficheiro organismos.pontos não existir, então a pontuação máxima será zero. Caso contrário, carrega-se o conteúdo do ficheiro para o Long da pontuação.

Neste momento já possuímos algum código, mas na prática ainda não acontece nada ao correr. Aliás, nem é possível correr. Foram referenciados alguns membros e classes que ainda não existem. Podemos começar por incluir as classes mais importantes do jogo. As que definem as entidades.

Ficheiro **Objectos\Objecto.vb**

```
Imports Organisms.Utilidades

Public MustInherit Class Objecto
    Private _destruir As Boolean

    Public Posicao As Vector2 = Vector2.Zero
    Public Escala As Vector2 = Vector2.One
    Public Rotacao As Single
    Public Opacidade As Single
    Public Grafico As Texture2D = Nothing
    Public OffsetCanais As Color = New Color
                                        (0, 0, 0, 0)

    Public ReadOnly Property Destruir As Boolean
        Get
            Return _destruir
        End Get
    End Property

    Public Sub OrdenarDestruicao()
        _destruir = True
    End Sub

    Public ReadOnly Property Largura As Integer
        Get
            Return CInt(Grafico.Width * Escala.X)
        End Get
    End Property

    Public ReadOnly Property Altura As Integer
        Get
            Return CInt(Grafico.Height * Escala.Y)
        End Get
    End Property

    Public ReadOnly Property Rectangulo As
                                        Rectangle
        Get
            Return New Rectangle(CInt
(Me.Posicao.X), CInt(Me.Posicao.Y), Largura,
                                        Altura)
        End Get
    End Property

    Public Function Distancia(Alvo As Objecto) As
                                        Single
        Dim dX As Double = ((Posicao.X + Centro.X)
                - (Alvo.Posicao.X + Alvo.Centro.X))
        Dim dY As Double = ((Posicao.Y + Centro.Y)
                - (Alvo.Posicao.Y + Alvo.Centro.Y))
        Return CSng(Math.Sqrt((Math.Pow(dX, 2) +
                                Math.Pow(dY, 2))))
    End Function

    Public Function CentroReal() As Vector2
        Return New Vector2(Posicao.X + Centro.X,
                            Posicao.Y + Centro.Y)
    End Function


    Private Function Centro() As Vector2
        Return New Vector2(Largura / 2.0F, Altura /
                                        2.0F)
    End Function

    Sub New(Grafico As Texture2D, Posicao As
                                        Vector2,
            Optional Rotacao As Single = 0,
                Optional Opacidade As Single = 1)
        Me.Grafico = Grafico
        Me.Posicao = Posicao
        Me.Rotacao = Rotacao
        Me.Opacidade = Opacidade

    End Sub

    Public Overridable Sub Actualiza(GT As
                                        GameTime)
        Opacidade = Grampo(Opacidade, 0.0F, 255.0F)
    End Sub

    Public Sub Desenha(SB As SpriteBatch)
        Dim C As Color = New Color(CByte(Grampo
(CInt(Me.Opacidade - Me.OffsetCanais.R), 0, 255)),
                                    CByte(Grampo
(CInt(Me.Opacidade - Me.OffsetCanais.G), 0, 255)),
                                    CByte(Grampo
(CInt(Me.Opacidade - Me.OffsetCanais.B), 0, 255)),
                                    CByte(Grampo
(CInt(Me.Opacidade - Me.OffsetCanais.A), 0, 255)))

        SB.Draw(Grafico,
                New Rectangle(CInt(Posicao.X +
                        Centro.X), CInt(Posicao.Y +
                        Centro.Y), Largura, Altura),
                Nothing, C,
                Rotacao,
                New Vector2(CSng(Grafico.Width /
                     2), CSng(Grafico.Height / 2)),
                SpriteEffects.None, 0)
    End Sub

End Class
```

Não há muito para explicar neste bloco. Trata-se de matemática relativamente simples, algumas variáveis para definir propriedades como a opacidade da entidade, rotação ou gráfico.
Encontramos também métodos para calcular distâncias entre dois Objectos e os métodos de actualização e desenho, que transmitem para o objecto ou para o spritebatch os valores necessários para comunicar o estado da sua representação.

Podemos enfatizar, no entanto, alguns elementos menos explícitos:

OffsetCanais trata-se de uma cor de apoio que fará desviar por os seus valores as intensidades dos canais da cor original do gráfico. Na prática, isto vai fazer com que um amarelo não seja absolutamente amarelo, mas que possa ser aleatoriamente vários tons de amarelo diferentes.

Como se trata da unidade da entidade, não se podem assumir transformações no método Actualiza. Garantimos apenas que a opacidade não sai dos valores aceitáveis e deixamos que este método seja “Overridable” para que camadas superiores que desta classe herdem possam implementar os seus próprios métodos de actualização com a lógica que lhes

compete, como por exemplo a classe Sprite:

Ficheiro **Objectos\Sprite.vb**

```
Imports Organisms.Utilidades

Public Class Sprite
    Inherits Objecto

    Public Velocidade_Angular As Single
    Public Desvanecimento As Single
    Public Crescimento As Vector2 = Vector2.Zero
    Public Velocidade As Vector2 = Vector2.Zero
    Public FactorVelocidade As Vector2 = Vec-
tor2.Zero

    Public Overrides Sub Actualiza(GT As GameTime)
        Posicao = Vector2.Add(Posicao, Vec-
tor2.Multiply(Velocidade, FactorVelocidade))
        Rotacao += Velocidade_Angular
        Opacidade += Desvanecimento
        MyBase.Actualiza(GT)
    End Sub

    Sub New(Grafico As Texture2D, Posicao As Vec-
tor2,
            Optional Rotacao As Single = 0, Optio-
nal Opacidade As Single = 255)
        MyBase.New(Grafico, Posicao, Rotacao, Opa-
cidade)
    End Sub
End Class
```

A classe Sprite já controla mais alguns aspectos do objecto. Está capaz de armazenar valores de velocidade e crescimento para afectar ao longo do tempo na posição, rotação e escala do objecto.

Para tal, implementa o seu próprio método Actualiza primeiro, e no fim o método do objecto.

Acima do Sprite, a classe Organismo implementa o mesmo por herança e ainda acrescenta uma variável para controlar o tamanho do organismo.

Ficheiro **Objectos\Organismo.vb**

```
Public Class Organismo
    Inherits Sprite

    Public Property Tamanho As Single
        Get
            Return MyBase.Escala.X
        End Get
        Set(value As Single)
            MyBase.Escala = New Vector2(value)
        End Set
    End Property

    Sub New(Grafico As Texture2D, Posicao As
                                        Vector2,
            Optional Rotacao As Single = 0,
                Optional Opacidade As Single = 255)
        MyBase.New(Grafico, Posicao, Rotacao,
                                        Opacidade)
    End Sub

End Class
```

O tamanho é de extrema importância pois a lógica básica do jogo passa por determinar se o organismo em que estamos a tocar é maior ou menor do que nós.

Por fim, implementando todas estas propriedades temos o objecto Powerup:

Ficheiro **Objectos\PowerUp.vb**

```
Imports Organisms.Utilidades

Public Class Powerup
    Inherits Organismo

    Public Tipo As Central.PowerUps
    Public Tempo As Integer = CInt(Seg_Frames(15))

    Sub New(Grafico As Texture2D, Posicao As
                                        Vector2,
            Optional Rotacao As Single = 0,
                Optional Opacidade As Single = 255)
        MyBase.New(Grafico, Posicao, Rotacao,
                                        Opacidade)
    End Sub

    Public Sub DescontaTempo()
      Tempo -= 1
      If Tempo <= 0 Then MyBase.OrdenarDestruicao()
    End Sub

End Class
```

Acrescentam-se ainda variáveis para determinar o tipo de powerup e a duração do efeito. Por fim, acrescenta-se o método para descontar o tempo da duração, garantindo que se levanta a flag da destruição quando o tempo termina. Verificaremos mais tarde de que forma é que se recolhem os objectos com flags levantadas.

Ainda à pasta dos objectos, acrescentamos também uma representação de som.

Uma estrutura deste tipo permite ser inequivocamente colocada numa colecção, mas ao mesmo tempo permite também repetir chaves. Assim, podemos agrupar vários itens a um tipo de chave.

```
Public Class Som
    Private _chave As String
    Private _media As SoundEffect

    Public ReadOnly Property Chave As String
        Get
            Return _chave
        End Get
    End Property

    Public ReadOnly Property Media As SoundEffect
        Get
            Return _media
        End Get
    End Property

    Sub New(Chave As String, Media As SoundEffect)
        _chave = Chave
        _media = Media
    End Sub
End Class
```

Organismos! Do zero ao mercado (1/2)

Ainda não conseguimos correr o projecto pois ainda existem erros de compilação. Podem verificar em mais do que um sítio o recurso a um método chamado Grampo, que ainda não existe.

É uma boa oportunidade para escrever o resto dos membros da classe Utilidades.

Ficheiro **Utilidades.vb**

```
    Public Shared Function MSeg_Frames(Milisegundos
                                  As Integer) As Long
        Return CLng((Milisegundos * 30) / 1000)
    End Function

    Public Shared Function Seg_Frames(Segundos As
                                      Integer) As Long
        Return CLng(Segundos * 30)
    End Function
```

Os métodos MSeg_Frames e Seg_Frames servem para converter valores de frames para segundos e vice-versa, tendo como premissa o framerate alvo de um jogo para Windows Phone: 30 fps. Assim, a qualquer altura o aparelho tentará render 30 frames do vosso jogo em 1 segundo e permite-nos tirar conclusões como 60 frames = 2 segundos, ou 3 segundos = 90 frames.

Isto é útil para evitar ter que arrancar a calculadora sempre que quisermos temporizar algo em unidades que nos façam mais sentido, neste caso o segundo.

```
Public Shared Function Grampo(Valor As Integer, Min
             As Integer, Max As Integer) As Integer
        If Valor > Max Then Return Max
        If Valor < Min Then Return Min
        Return Valor
    End Function

    Public Shared Function Grampo(Valor As Single,
Min As Single, Max As Single) As Single
        If Valor > Max Then Return Max
        If Valor < Min Then Return Min
        Return Valor
    End Function
```

O método Grampo é um método simples mas de extrema importância.
Este método garante que determinado valor oscila apenas dentro de um determinado alcance. A opacidade, por exemplo, só pode oscilar do 0 ao 255. Qualquer valor que esta variável tente assumir acima de 255 torna-se 255, e abaixo de 0 torna-se 0. Caso não exceda o alcance, assume o valor que tentou assumir.

```
Public Shared Sub ComunicarPontuacao()
  If Not
    Net.NetworkInformation.NetworkInterface.
                GetIsNetworkAvailable Then Exit Sub
    Dim Pontos As Long = Central.PontuacaoMaxima
    Dim WC As New Net.WebClient
             WC.DownloadStringAsync(New Uri
               ("http://um.url.qualquer?pontos=" &
                                  Pontos.ToString))
    End Sub
```

O método comunicar pontuação servirá para passar a pontuação com que o jogador ficou depois de desistir. Neste caso é responsabilidade do script do lado do servidor determinar se a pontuação enviada é superior à que existe centralizada e registar se for.

Para este sistema simples, não existe conceito de identidade. Existe um valor de pontuação global que pode ser comprovado com o valor máximo obtido no próprio aparelho. Por exemplo, eu não posso alegar que sou campeão se no meu dispositivo indica 300 pontos máximos mas o valor global aponta 600. Isto significa que alguém algures possuí 600 no registo de pontos do aparelho. O URL de comunicação, neste caso, é falso para evitar fraude por parte dos leitores do artigo (malandros!) pois não investi em métodos de assegurar que a pontuação é impossível de adulterar. (malandro!)

```
Public Shared Sub ActualizarPontuacaoGlobal()
  If Not
       Net.NetworkInformation.NetworkInterface.
                          GetIsNetworkAvailable Then
       Central.PontuacaoMaximaGlobal = "no
                                       connection..."
            Exit Sub
        End If
        Central.PontuacaoMaximaGlobal = "attempting
                                    to get score..."
        Dim WC As New Net.WebClient
        AddHandler WC.DownloadStringCompleted, Sub
                          (sender As Object, e As
              Net.DownloadStringCompletedEventArgs)

          Central.PontuacaoMaximaGlobal = e.Result
                                             End Sub
        WC.DownloadStringAsync(New Uri
                ("http:// um.url.qualquer?pontos"))
    End Sub
```

O método ActualizarPontuacaoGlobal tenta comunicar com o servidor para obter o valor de pontos máximo. O URL também é falso.

```
Public Shared Sub GravarPontuacaoPessoal()
 Dim IST As IsolatedStorageFile =
   IsolatedStorageFile.GetUserStoreForApplication()
   Using wF As New IO.StreamWriter(New
     IsolatedStorageFileStream("organismos.pontos",
     FileMode.OpenOrCreate, FileAccess.Write, IST))
            wF.Write
              (Central.PontuacaoMaxima.ToString)
        End Using
        ComunicarPontuacao()
    End Sub
End Class
```

Por fim, o método GravarPontuacaoPessoal escreve o valor actual de pontuação máxima para o ficheiro no dispositivo, e tenta comunicar para o servidor também.

Agora já é possível compilar e correr... mas não é muito interessante.

Apesar de já termos escrito grande parte estrutural do jogo, ainda não escrevemos nada que se possa reflectir visualmente.

Para corrigir isso, vamos começar por escrever a classe parcial responsável por carregar conteúdos da content pipeline:

Ficheiro **Jogo.Recursos.vb**

```
Partial Public Class Jogo

    Public Intro_A As Texture2D
    Public Intro_B As Texture2D
    Public Menu As Texture2D
    Public BT_Jogar As Texture2D
    Public BT_Recorde As Texture2D
    Public BT_Sair As Texture2D
    Public BT_Resumir As Texture2D
    Public BT_Menu As Texture2D
    Public Jogador As Texture2D
    Public Fundo1 As Texture2D
    Public Fundo2 As Texture2D
    Public Mascara As Texture2D
    Public Organismos As New List(Of Texture2D)
    Public SFX As New List(Of Som)
    Public Musica As New List(Of Song)
    Public Powerups As New Dictionary(Of
                       Central.PowerUps, Texture2D)

    Public Sub CarregarRecursos()
        Intro_A = Content.Load(Of Texture2D)
        ("intro1")
        Intro_B = Content.Load(Of Texture2D)
        ("intro2")
        Menu = Content.Load(Of Texture2D)("menu")
        BT_Jogar = Content.Load(Of Texture2D)
        ("btjogar")
        BT_Recorde = Content.Load(Of Texture2D)
        ("btrecorde")
        BT_Sair = Content.Load(Of Texture2D)
        ("btsair")
        BT_Resumir = Content.Load(Of Texture2D)
        ("btresumir")
        BT_Menu = Content.Load(Of Texture2D)
        ("btmenu")
        Jogador = Content.Load(Of Texture2D)
        ("jogador")
        Central.Letras = Content.Load(Of
                          SpriteFont)("Principal")
        Fundo1 = Content.Load(Of Texture2D)
        ("fundo1")
        Fundo2 = Content.Load(Of Texture2D)
        ("fundo2")
        Mascara = Content.Load(Of Texture2D)
        ("mascara")

        Powerups(Central.PowerUps.INVERSAOGENETICA)
        = Content.Load(Of Texture2D)("powerup1")
        Powerups(Central.PowerUps.AGITACAOGENETICA)
        = Content.Load(Of Texture2D)("powerup2")
        Powerups(Central.PowerUps.DOBRO) =
             Content.Load(Of Texture2D)("powerup3")
        Powerups(Central.PowerUps.TRIPLO) =
             Content.Load(Of Texture2D)("powerup4")
        Powerups(Central.PowerUps.FALHAOPTICA) =
             Content.Load(Of Texture2D)("powerup5")

        For b As Integer = 1 To 12
            Organismos.Add(Content.Load(Of
                           Texture2D)("bac" & b))
        Next
        Organismos.Add(Content.Load(Of Texture2D)
                                    ("extra1"))
        Organismos.Add(Content.Load(Of Texture2D)
                                    ("extra2"))
        Organismos.Add(Content.Load(Of Texture2D)
                                    ("extra3"))
        Organismos.Add(Content.Load(Of Texture2D)
                                    ("extra4"))

        For Each Tipo As String In {"comer",
"comido"}
        For i As Integer = 1 To 5
         SFX.Add(New Som(Tipo.ToUpper, Content.Load
                   (Of SoundEffect)("sfx\" & Tipo &
                                    i.ToString)))
            Next
        Next
        SFX.Add(New Som("POWERUP", Content.Load(Of
                      SoundEffect)("sfx/powerup")))
        SFX.Add(New Som("TOQUE", Content.Load(Of
                        SoundEffect)("sfx/toque")))
        SFX.Add(New Som("SAFO", Content.Load(Of
                        SoundEffect)("sfx/safo1")))

        For i As Integer = 1 To 5
            Musica.Add(Content.Load(Of Song)
                    ("musica/musica" & i.ToString))
        Next
    End Sub
End Class
```

Aqui declaramos todos os tipos de recursos XNA que precisamos para dar vida aos tipos criados por nós. Temos uma série de Texture2D para as várias imagens usadas, listas de Texture2D para os vários organismos disponíveis, efeitos de som, músicas e fonte.

A maioria das Texture2D são carregadas directamente com o Content.Load, para outras utilizo ciclos para evitar escrever todas as linhas.

Isto só por si, continua a não fazer nada. Vamos ter de começar a dar vida à classe Jogo, e começar a unir as pontas.

Unir pontas soltas

Para começarmos a ver resultados, voltemos à classe Jogo.

Ficheiro **Jogo.vb**

```
    Protected Overrides Sub LoadContent()
        spriteBatch = New SpriteBatch
(GraphicsDevice)
        CarregarRecursos()
        InicializarEntrada()
        InicializarMenu()
        InicializarJogador()
        TocarMusica(0)
    End Sub
```

No método LoadContent vamos carregar todos os nossos recursos e começar as inicializações.

Ficheiro **Jogo.Inicializacoes.vb**

```
Public Sub InicializarEntrada()
        Central.EcraIntro_A = New Sprite(Intro_A,
                                  Vector2.Zero, 0, 0)
        Central.EcraIntro_B = New Sprite(Intro_B,
                                  Vector2.Zero, 0, 0)
        Central.Menu = New Sprite(Menu,
                                  Vector2.Zero, 0, 0)
        Central.FundoJogo1 = New Sprite(Fundo1,
                                  Vector2.Zero, 0, 0)
        Central.FundoJogo2 = New Sprite(Fundo2,
                                  Vector2.Zero, 0, 0)
        Central.MascaraPU = New Sprite(Mascara,
                                  Vector2.Zero, 0, 0)
    End Sub
```

Organismos! Do zero ao mercado (1/2)

Para inicializar a entrada, vamos preparar objectos Sprite com as Texture2D dos dois ecrãs de apresentação e a imagem de menu. Aproveitamos também esta inicialização para carregar as duas imagens de fundo do jogo e a máscara do powerup "Falha óptica"

```
    Public Sub InicializarJogador()
        Dim ORG As New Organismo(Jogador,
                                 Vector2.Zero, 0, 255)
        ORG.Escala = New Vector2(1, 1)
        ORG.FactorVelocidade = New Vector2(0, 0)
        ORG.Velocidade_Angular = 0
        ORG.Rotacao = 0
        ORG.Opacidade = 255
        ORG.Tamanho = 0.04F
        Central.Jogador = ORG
    End Sub
```

O jogador é uma instância do objecto Organismo, com os valores do canal alpha originais, sem qualquer valor de transformação, e com 4% do seu tamanho gráfico original. Esta instância fica centralizada na referência Central.Jogador.

```
Public Sub InicializarMenu()
        Central.Botao_Jogar =
            New Sprite(BT_Jogar,
                    New Vector2(LARGURA -
               BT_Jogar.Width - 10, 180), 0, 0)
        Central.Botao_Recorde =
            New Sprite(BT_Recorde,
            New Vector2(LARGURA -
            BT_Recorde.Width - 10, 280), 0, 0)
        Central.Botao_Sair =
            New Sprite(BT_Sair,
            New Vector2(LARGURA - BT_Sair.Width -
                                  10, 380), 0, 0)
        Central.Botao_Continuar =
            New Sprite(BT_Resumir,
            New Vector2(CSng(LARGURA / 2) -
                CSng(BT_Resumir.Width / 2), 280),
                                            0, 255)
        Central.Botao_Menu =
            New Sprite(BT_Menu,
            New Vector2(CSng(LARGURA / 2) - CSng
                 (BT_Menu.Width / 2), 380), 0, 255)
    End Sub
```

O menu inicializa-se com sprites para todos os botões possíveis, não só no menu inicial mas também no ecrã de pausa e recorde.
Estes sprites são imediatamente colocados nas suas posições finais, que são estáticas. Todas estas instâncias vão ficar centralizadas nas respectivas referências na classe Central.

Para colocarmos os elementos iniciais e determinar as fases, vamos escrever a classe parcial do fluxo:

Ficheiro **Jogo.Fluxo.vb**

```
Imports Organisms.Utilidades

Partial Public Class Jogo

    Private Enum Fases
        INTRO
        MENU
        ENTRADA_NIVEL
        JOGO
        PAUSA
        RECORDE
    End Enum

    Private FRAME As Long
    Private Fase As Fases = Fases.INTRO

    Private Sub VerificarLinhaDeTempo()
      Select Case FRAME
        Case MSeg_Frames(0)
          Central.EcraIntro_A.Desvanecimento = 10

        Case MSeg_Frames(2000)
          Central.EcraIntro_A.Desvanecimento = -10
          Central.EcraIntro_B.Desvanecimento = 10

        Case MSeg_Frames(4000)
          Central.EcraIntro_B.Desvanecimento = -10
               Central.Menu.Desvanecimento = 10

        Case MSeg_Frames(4500)
           Dim R As New Random
            For bac As Integer = 1 To 65
             Central.ORGANISMOSMENU.Add
             (CriarOrganismosMenu(New Vector2
                (R.Next(0, 800), R.Next(0, 480))))
            Next

        Case MSeg_Frames(5000)
           Central.IniciarLeituraAcelerometro()
           Fase = Fases.MENU
           Central.Botao_Jogar.Desvanecimento = 25

        Case MSeg_Frames(5200)
         Central.Botao_Recorde.Desvanecimento = 25

        Case MSeg_Frames(5400)
           Central.Botao_Sair.Desvanecimento = 25

       End Select
    End Sub

End Class
```

Podemos verificar a Enum que identifica as diferentes fases do jogo e de seguida um método que permite simular uma linha de tempo a partir de uma contagem de frames global. Isto facilita a entrada de objectos em cena dados os tempo, em frames, para a realização dos acontecimentos. Com esta verificação conseguimos coreografar a entrada com desvanecimentos das duas imagens de apresentação e do ecrã de menu.

Quando a variável FRAME assume determinados valores, são alteradas as propriedades de transformação dos sprites de forma a que se tornem visíveis ou não. Isto é apenas um exemplo do que se pode coreografar. As possibilidades são infinitas.

Quando existem frames suficientes para 4500 milissegundos (4,5 segundos) criamos todos os organismos enfeite que vão dar vida ao menu, e aos 5 segundos trocamos a fase do jogo de INTRO para MENU.

Agora só precisamos que esta "timeline" seja verificada em todos os frames, e que os seus intervenientes são devidamente desenhados no spritebatch, mas antes vamos escre-

ver o método CriarOrganismoMenu e o método TocarMusica na classe parcial de Inicializações, para descansar o interpretador dos erros:

Ficheiro **Jogo.Inicializacoes.vb**

```
Public Function CriarOrganismosMenu(posicao As
                                    Vector2) As Sprite
        Dim R As Random = Central.AleatorioGlobal
        Dim Num As Integer = R.Next(0,
                             Organismos.Count - 1)
        Dim B As New Sprite(Organismos(Num),
                                            posicao)
        Dim tmpEscala As Single = CSng(R.NextDouble
                                    () * 0.3) + 0.2F
        Dim tmpVelocidade As Single = Grampo(CSng
                  (R.NextDouble() * 8), 1.0F, 8.0F)
        B.Escala = New Vector2(tmpEscala)
        B.FactorVelocidade = New Vector2
                                        (tmpVelocidade)
        B.Velocidade_Angular = Grampo(CSng
              ((R.NextDouble() * 0.032F) - 0.016F),
                                    -0.016F, 0.016F)
        B.Rotacao = CSng(R.NextDouble * Math.PI)
        B.Opacidade = CSng(R.Next(100, 150))

        Dim tmpR, tmpG, tmpB As Integer
        tmpR = R.Next(0, 120)
        tmpG = R.Next(0, 150)
        tmpB = R.Next(0, 150)
        B.OffsetCanais = New Color(0, tmpR, tmpG,
                                                tmpB)

        Return B
    End Function
```

O método CriarOrganismosMenu consiste em atribuir valores aleatórios às propriedades que definem um organismo, desde o seu tamanho até ao seu tipo ou tonalidade, e devolver o organismo devidamente formatado. Este método, como se pode observar antes, é chamado 65 vezes num ciclo que acumula organismos na colecção ORGANISMOSMENU, que mantém obviamente todos os organismos que circularão no fundo do menu.

```
Public Sub TocarMusica(Optional Forcar As Integer =
                                                -1)

        If Not MediaPlayer.GameHasControl Then Exit
Sub

        Dim tmpMusica As Song = Nothing

        If Forcar = -1 Then
            Dim R As Random =
                            Central.AleatorioGlobal
            tmpMusica = Musica(R.Next(0,
                                    Musica.Count))
        Else
            tmpMusica = Musica(Forcar)
        End If
        MediaPlayer.Stop()
        MediaPlayer.IsRepeating = True
        MediaPlayer.Play(tmpMusica)

    End Sub
```

O método TocarMusica inicia uma música aleatória, e toca-a em repetição eterna. Caso o utilizador já esteja a ouvir música, o que se reflecte no membro MediaPlayer.GameHasControl, das duas, uma: ou não fazemos nada, ou perguntamos ao utilizador se deseja parar a música que está actualmente a ouvir. Neste caso, não vamos fazer nada. Se decidirem ignorar esta verificação e exista a possibilidade do vosso jogo interromper a música que o utilizador estiver a ouvir no mediaplayer do smartphone, **o mecanismo de certificação vai recusar o vosso jogo**. Não se pode interromper a experiência do utilizador sem o seu consentimento.

Com o fluxo escrito, existência de fases e a coreografia de entrada, podemos começar a escrever os métodos Update e Draw da classe Jogo:

Ficheiro **Jogo.vb**

```
Protected Overrides Sub Update(ByVal gameTime As
GameTime)

        Select Case Fase
            Case Fases.INTRO
                ActualizarIntro(gameTime)

            Case Fases.MENU
                ActualizarOrganismosMenu(gameTime)
                ActualizarMenu(gameTime)
                DetectarToques(gameTime)

            Case Fases.RECORDE
                ActualizarOrganismosMenu(gameTime)
                DetectarToques(gameTime)

        End Select

        FRAME += 1
        If FRAME = Long.MaxValue - 1 Then FRAME = 1
        MyBase.Update(gameTime)
    End Sub
```

Como no artigo anterior, gosto de separar as acções por fases.
Na fase INTRO, foi escrito um único método que controla, que se encontra naturalmente na classe parcial das actualizações:

Ficheiro **Jogo.Actualizacoes.vb**

```
Private Sub ActualizarIntro(GT As GameTime)
        VerificarLinhaDeTempo()
        Central.EcraIntro_A.Actualiza(GT)
        Central.EcraIntro_B.Actualiza(GT)
        Central.Menu.Actualiza(GT)
    End Sub
```

Como seria de esperar, na fase INTRO estamos a chamar o método VerificaLinhaDeTempo que vai preparar todos os elementos gráficos para um determinado frame. De seguida, chamam-se simplesmente os métodos de actualização dos elementos intervenientes na introdução: as imagens de apresentação e a imagem do menu.

Logo de seguida, no Update da classe Jogo, os intervenientes da fase MENU também são chamados da classe parcial de actualizações:

Organismos! Do zero ao mercado (1/2)

```
    Private Sub ActualizarMenu(GT As GameTime)
        VerificarLinhaDeTempo()
        Central.Menu.Actualiza(GT)
        Central.Botao_Jogar.Actualiza(GT)
        Central.Botao_Recorde.Actualiza(GT)
        Central.Botao_Sair.Actualiza(GT)
    End Sub
```

De forma muito semelhante à introdução, aqui também se verifica a linha do tempo, pois os botões do menu também são colocados de acordo com a coreografia, e todos os botões intervenientes são actualizados a partir das suas respectivas referências aos objectos.

```
Private Sub ActualizarOrganismosMenu(GT As
                                     GameTime)
   For Each B As Sprite In Central.ORGANISMOSMENU
    B.Velocidade = New Vector2
       ((Central.LeituraAcel.Y * 5) * -1,
        (Central.LeituraAcel.X * 5) * -1)
        B.Actualiza(GT)
        If B.Posicao.X < (B.Largura * -1) * 2
         Then B.Posicao.X = 800 + (B.Largura * 2)
         If B.Posicao.Y < (B.Altura * -1) * 2
         Then B.Posicao.Y = 480 + (B.Altura * 2)
         If B.Posicao.X > 800 + (B.Largura * 2)
         Then B.Posicao.X = (B.Largura * -1) * 2
         If B.Posicao.Y > 480 + (B.Altura * 2)
         Then B.Posicao.Y = (B.Altura * -1) * 2
        Next
End Sub
```

Os organismos do menu, que se encontram todos na lista central ORGANISMOSMENU, são todos afectados por as mesmas condições.

Neste caso, estamos a alterar o vector da velocidade com as leituras do acelerómetro. Assim, enquanto o menu aguarda por decisão do utilizador, os organismos apresentados como fundo respondem à inclinação do smartphone. Na criação, cada organismo adquiriu um valor de velocidade base, que é multiplicado por a leitura do acelerómetro. Isto dá a ilusão de que os organismos se encontram em camadas desfasadas, movimentando-se uns mais depressa do que outros, para as mesmas leituras de acelerómetro.

É possível detectar as inclinações necessárias para este tipo de jogo apenas em dois eixos.

Se imaginarmos o acelerómetro como uma esfera na ponta de uma mola sujeita à força da gravidade, é relativamente fácil de calcular a orientação do aparelho medindo as deslocações da esfera em diferentes planos, e fica claro que eixos podemos utilizar. Não nos interessa fazer leituras em "profundidade".

Na figura podemos observar as elipses que nos interessam e a que eixos se referem.

Como queremos aproveitar o ecrã em Landscape, as leituras

para os movimentos acompanham, obviamente, a orientação do aparelho. Assim, As leituras no eixo X vão indicar-nos a inclinação vertical (no nosso ponto de vista) e Y a horizontal.

As leituras em Z poderiam ser utilizadas para melhorar a detecção dos ajustes necessários a fazer no X e Y para garantir que se poderia jogar até a fazer o pino.

Para adoptar uma postura de jogo mais confortável, adicionamos um valor de "offset" ao eixo X de 0.4 (lembremo-nos que os valores do acelerómetro são dados em "gravidades": 1 = eixo completamente a favor da gravidade, 0 = eixo indiferente à gravidade e -1 = eixo completamente oposto à gravidade. Este último obtém-se tipicamente num movimento brusco. Movimento bruscos podem também resultar em valores acima de 1 ou abaixo de -1), esta adição força a posição de estabilidade para uma posição perto dos 45º de inclinação em relação à gravidade, que é a que se adopta quando se joga sentado.

Para além disto, quando um dos organismos sai do campo visual por um determinado eixo, é reposicionado para o seu extremo oposto, o que dá uma ilusão de organismos infinitos, mesmo sendo apenas 65 como sabem.

```
Private Sub DetectarToques(GT As GameTime)
 Dim TC As TouchCollection = TouchPanel.GetState
   If TC.Count = 0 Then Exit Sub
   Dim TL As TouchLocation = TC(0)
    If TL.State = TouchLocationState.Released Then
            Dim ToqueRect As New Rectangle(CInt
       (TL.Position.X), CInt(TL.Position.Y), 1, 1)
      Select Case Fase
       Case Fases.MENU
        If ToqueRect.Intersects
         (Central.Botao_Jogar.Rectangulo) Then
        End If
        If ToqueRect.Intersects
         (Central.Botao_Recorde.Rectangulo) Then
                        TocarSFX("TOQUE")
                        ActualizarPontuacaoGlobal()
                        Fase = Fases.RECORDE
        End If
        If ToqueRect.Intersects
         (Central.Botao_Sair.Rectangulo) Then
                        TocarSFX("TOQUE")
                        Me.Exit()
        End If
       Case Fases.RECORDE
        If ToqueRect.Intersects
         (Central.Botao_Menu.Rectangulo) Then
         TocarSFX("TOQUE")
         Fase = Fases.MENU
        End If
       End Select
      End If
End Sub
```

É também nesta actualização que se detectam os toques do utilizador.
É necessária esta verificação sempre que existam elementos no ecrã que possam interagir com o toque. Como os menus têm botões, queremos que as escolhas sejam feitas com um toque no ecrã. Para obtermos os dados de interacção do ecrã, utilizamos uma referência a TouchPanel.GetState.

```
Dim TC As TouchCollection = TouchPanel.GetState
```

TC representa agora uma colecção de pontos de toque. Se TC estiver vazia, não existe nenhuma área do ecrã a ser tocada. Um item ou mais, significa que existem um ou mais focos de toque.

Por cada ponto de toque, é possível também determinar de que tipo de interacção se tratou: pode ser um toque novo, um movimento, ou um toque que deixou de existir. Isto é importante para podermos controlar o tipo de acção.

No caso dos botões, verificamos apenas no primeiro foco de toque que tenha sido recentemente levantado.

```
If TL.State = TouchLocationState.Released Then
```

Levantado para permitir que se possa tocar no botão sem levantar. Só consideramos uma opção quando levantam o dedo.
Quando ocorre, comparamos a coordenada exacta de onde aconteceu com a área dos rectângulos dos botões. Se existir intersecção, significa que o utilizador optou por aquele botão, e executamos as funções correspondentes.

Vamos dar um salto à classe parcial das inicializações para escrever o método TocarSFX, que é usado nos toques nas opções, para emitir um pequeno som ao seleccionar algo:

Ficheiro **Jogo.Inicializacoes.vb**

```
Public Sub TocarSFX(Tipo As String)
 Dim R As Random = Central.AleatorioGlobal
 Dim tmpEnum As IEnumerable(Of Som) = SFX.Where
                    (Function(x) x.Chave = Tipo)
 Dim tmpSom As SoundEffect = tmpEnum(R.Next(0,
                              tmpEnum.Count)).Media
        tmpSom.Play(1, 0, 0)
End Sub
```

O método TocarSFX pede um parâmetro que determina a chave do som que se pretende. A colecção dos efeitos de som contém vários itens do tipo Som, que são constituídos por uma chave e por o som. Esta chave pode ocorrer várias vezes, como já foi abordado.

O método TocarSFX vai primeiro recolher todos os sons que possuam a chave indicada, e vai tocar, apenas dessa recolha, um aleatório. Isto permite a adição de vários sons distintos para uma mesma acção.

Voltando para a classe principal Jogo, vamos escrever todos os métodos que são chamados do Draw:

Ficheiro **Jogo.vb**

```
    Protected Overrides Sub Draw(ByVal gameTime As
                                          GameTime)
        GraphicsDevice.Clear(Color.Black)
        Select Case Fase
            Case Fases.INTRO
                spriteBatch.Begin()
                DesenharIntro(spriteBatch)
                spriteBatch.End()

            Case Fases.MENU
                spriteBatch.Begin
                (SpriteSortMode.Deferred,
                             BlendState.AlphaBlend)
                DesenharOrganismosMenu(spriteBatch)
                spriteBatch.End()
                spriteBatch.Begin()
                DesenharMenu(spriteBatch)
                spriteBatch.End()

            Case Fases.RECORDE
                spriteBatch.Begin
                SpriteSortMode.Deferred,
                             BlendState.AlphaBlend)
                DesenharOrganismosMenu(spriteBatch)
                spriteBatch.End()
                spriteBatch.Begin()
                DesenharRecorde(spriteBatch)
                spriteBatch.End()

        End Select
        MyBase.Draw(gameTime)
    End Sub
```

Com o método Update pronto, está tudo a acontecer nos bastidores, mas as cortinas estão corridas até que colocquemos vida no spritebatch.

Há que destacar apenas a forma como se estão a desenhar os organismos. Tanto para o MENU como RECORDE, são enviados 2 spritebatches.

O dos organismos com blendstate definido para AlphaBlend, e outro para os restantes elementos, normal. O blendstate AlphaBlend faz com que a sobreposição de cor no buffer seja somada e não substituída por o objecto com a ordem de desenho superior.

Em termos de organismos, isto fará com que se evidenciem que os seus corpos são translúcidos e que estão de facto em intersecção. Como explicado num artigo anterior, este tipo de mistura é usado para dar brilho a explosões por exemplo.

Todos os métodos de desenho acontecem na classe parcial de desenho:

Ficheiro **Jogo.Desenho.vb**

```
Imports Organisms.Utilidades

Partial Public Class Jogo

    Private Sub DesenharIntro(SB As SpriteBatch)
        Central.EcraIntro_A.Desenha(SB)
        Central.EcraIntro_B.Desenha(SB)
```

```
        Central.Menu.Desenha(SB)
    End Sub

    Private Sub DesenharOrganismosMenu(SB As
                                       SpriteBatch)
     For Each B As Sprite In Central.ORGANISMOSMENU
            B.Desenha(SB)
     Next
    End Sub

    Private Sub DesenharMenu(SB As SpriteBatch)
        Central.Menu.Desenha(SB)
        Central.Botao_Jogar.Desenha(SB)
        Central.Botao_Recorde.Desenha(SB)
        Central.Botao_Sair.Desenha(SB)
    End Sub
```

Estes três métodos chamam simplesmente os métodos de desenho dos próprios objectos, passando a spritebatch.

```
Private Sub DesenharRecorde(SB As SpriteBatch)
  Central.Botao_Menu.Desenha(SB)
   Dim TmpLargura As Single =
        Central.Letras.MeasureString("GLOBAL MAX
                                      SCORE").X
        SB.DrawString(Central.Letras, "GLOBAL MAX
            SCORE", New Vector2(CSng(LARGURA / 2) -
            CSng(TmpLargura / 2), 30), Color.White)
        TmpLargura = Central.Letras.MeasureString
         (Central.PontuacaoMaximaGlobal.ToString).X
        SB.DrawString(Central.Letras,
            Central.PontuacaoMaximaGlobal.ToString,
               New Vector2(CSng(LARGURA / 2) - CSng
               (TmpLargura / 2), 100), Color.White)
        TmpLargura = Central.Letras.MeasureString
                                  ("MY MAX SCORE").X
        SB.DrawString(Central.Letras, "MY MAX
            SCORE", New Vector2(CSng(LARGURA / 2) -
               CSng(TmpLargura / 2), 170), Color.White)
        TmpLargura = Central.Letras.MeasureString
               (Central.PontuacaoMaxima.ToString).X
        SB.DrawString(Central.Letras,
        Central.PontuacaoMaxima.ToString, New
                  Vector2(CSng(LARGURA / 2) - CSng
                 (TmpLargura / 2), 240), Color.White)
    End Sub

End Class
```

Por fim, o desenho do recorde consiste essencialmente na apresentação de 4 linhas de texto que nos informam da pontuação global actual e da nossa pontuação pessoal máxima.

Testar, alterar, experimentar

Com isto concluímos a primeira parte, de duas. Podem correr o jogo e observar como se comportam os organismos, alterar valores e experimentar novas abordagens. Com o emulador e o simulador de gravidade é relativamente difícil conseguir mover o aparelho como o faríamos na realidade, mas é suficiente para testar.
O que temos já promete alguma coisa e será bastante interessante escrever convosco o ciclo de jogo. Na altura do lançamento do segundo artigo, o jogo que estamos a escrever estará no mercado do Windows Phone, que é também um dos focos da segunda parte: as mecânicas de certificação e colocação em mercado.

Até lá!

AUTOR

Escrito por Sérgio Ribeiro

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

Elege o melhor artigo desta edição

Revista PROGRAMAR

http://tiny.cc/ProgramarED37_V

ENIGMAS DO C# - ASYNC/AWAIT E THREADS

Resultado (continuação da página 27)

Mesmo dentro dos métodos assíncronos (métodos cuja declaração é antecedida da palavra-chave **async**), todo o código executado até à primeira palavra-chave **await** é executado no mesmo *thread* da chamada.

O resultado da execução de uma instrução precedida de **await** é, geralmente, numa tarefa (Task ou Task<T>) e apenas a execução dessa tarefa poderá provocar uma mudança de thread.

Todo o código que se segue a uma instrução precedida de **await** é transformado num novo método que é executado como tarefa na continuação da tarefa anterior.

No caso de ambientes de execução como as interfaces visuais (Windows Forms, WPF, Silverlight, etc.), se existir um contexto de sincronização corrente (SynchronizationContext.Current) a execução dessa tarefa será, normalmente, transferida assíncronamente para o contexto de sincronização através do método Post. No caso de ambientes de execução como as interfaces visuais, essa tarefa irá sempre ser executada no *thread de UI* (**UI Thread**).

Resultados

1

```
>>> 1
UI Thread
    DoWorkAsync()
Post
UI Thread
    DoWorkAsync()
Post
UI Thread
    DoWorkAsync()
Post
UI Thread
<<< 1
```

Explicação

Uma vez que a chamada tem origem na interface com o utilizador, a sua execução tem início no **UI Thread**.

Ao executar a tarefa correspondentemente ao método **DoWorkAsync** a execução é transferida para para um *work thread* da *thread pool* (ThreadPool) e, após a sua conclusão, é despachada a execução das restantes instruções do método para o contexto de execução.

2

```
>>> 2
UI Thread
>>> 2'
UI Thread
    DoWorkAsync()
Post
UI Thread
    DoWorkAsync()
Post
UI Thread
    DoWorkAsync()
Post
UI Thread
<<< 2'
UI Thread
<<< 2
```

Explicação

A diferença para o caso anterior é o facto de existir assincronismo em cascata.

Mas porque a mudança de *thread* só existirá ao iniciar-se a execução da primeira tarefa da sequência, essa execução iniciar-se-á sempre no *thread* de UI e, como tal, a execução continuará no *thread* de UI após a execução de cada tarefa.

3

```
>>> 2
UI Thread
>>> 2'
UI Thread
    DoWorkAsync()
Post
UI Thread
    DoWorkAsync()
Post
UI Thread
    DoWorkAsync()
Post
UI Thread
<<< 2'
UI Thread
<<< 2
```

Explicação

Este caso é, no essencial, idêntico ao anterior com uma pequena diferença: a execução da primeira tarefa é feita com a indicação de que a execução não deve ser executada no contexto de sincronização corrente, caso exista um.

Como já se constatou nos casos anteriores, sequências de chamadas assíncronas executadas na presença de um contexto de execução podem originar mudanças de contexto indesejáveis além da execução de código potencialmente moroso ou bloqueante.

Esta prática é recomendada em código que não manipule interfaces com o utilizador, em especial em bibliotecas a serem disponibilizadas. Seria muito prejudicial à aplicação que usasse uma biblioteca de leitura assíncrona de ficheiros se a execução retornasse à *thread* de UI a cada carácter lido.

4

```
>>> 4
UI Thread
>>> 4'
Worker Thread
    DoWorkAsync()
Worker Thread
    DoWorkAsync()
Worker Thread
    DoWorkAsync()
Worker Thread
<<< 4'
Post
UI Thread
<<< 4
```

Explicação

Neste caso, ao invés de se executar assincronamente uma tarefa resultante de uma chamada a um método assíncrono, é criada uma nova tarefa da qual é escondido o contexto de execução corrente, caso exista.

Esta prática é recomendada quando se quer garantir que o código a ser chamado, caso contenha instruções assíncronas, não será executado no contexto de execução corrente.

5

```
>>> 5
UI Thread
>>> 5'
Worker Thread
    DoWorkAsync()
Worker Thread
    DoWorkAsync()
Worker Thread
    DoWorkAsync()
Worker Thread
<<< 5'
Post
UI Thread
<<< 5
```

Explicação

Este caso é, no essencial, idêntico ao anterior. O método Task.Run foi introduzido na **Framework 4.5** com o objetivo de ser usado em conjunto com a palavra-chave **await**. Entre outras particularidades, tarefas criadas com o método **Task.Run** são criadas com TaskScheduler.Default, pelo que o código a ser executado dentro da tarefa será executado como se não exista um contexto de execução corrente.

Conclusão

A introdução da **Task Parallel Library (TPL)** na **Framework 4.0** veio facilitar a escrita, consumo e execução de código assíncrono.

Com a maturação da **TPL** na **Framework 4.**5 em conjugação com as palavras-chave **async** e **await** estas tarefas tornaram-se ainda mais facilitadas.

No entanto, na sua utilização é necessário ter sempre presente o seu funcionamento para que não se obtenham resultados inesperados e indesejados.

Ligações

- C# Reference
- Visual Studio 2012
- Framework .NET 4.5
- Task Parallel Library (TPL)
- Asynchronous Programming with Async and Await (C# and Visual Basic)
- An Async Premier
- Task.Run vs Task.Factory.StartNew
- Blogue do Eric Lippert
- Blogue do Lucian Wischik
- Blogue da equipa de Parallel Programming

Task-based Asynchronous Pattern

AUTOR

Escrito por Paulo Morgado
É licenciado em Engenharia Electrónica e Telecomunicações (Sistemas Digitais) pelo Instituto Superior de Engenharia de Lisboa e Licenciado em Engenharia Informática pela Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa. Pelo seu contributo para a comunidade de desenvolvimento em .NET em língua Portuguesa, a Microsoft premeia-o com o prémio MVP (C#) desde 2003. É ainda co-autor do livro "LINQ Com C#" da FCA.

Core Dump [8] - Fora de Horas

Estamos numa época do ano especial.

Estamos no inicio de mais um ano letivo, para uns é o inicio do seu derradeiro percurso académico, para outros o fim.

E outros ainda, começam agora a sua carreira profissional.

Seja qual for o estado em que se encontram, é importante que compreendam que na área das Tecnologias de Informação (TI) a aprendizagem é uma constante, como tantas vezes o tenho referido aqui e na comunidade P@P.

Para todos quantos encetam agora uma nova etapa da sua vida, é importante que tenham a consciência que, se se querem distinguir e ir mais além em relação aos seus pares, terão de se esforçar mais, um sacrifício que terá de ser feito fora de horas.

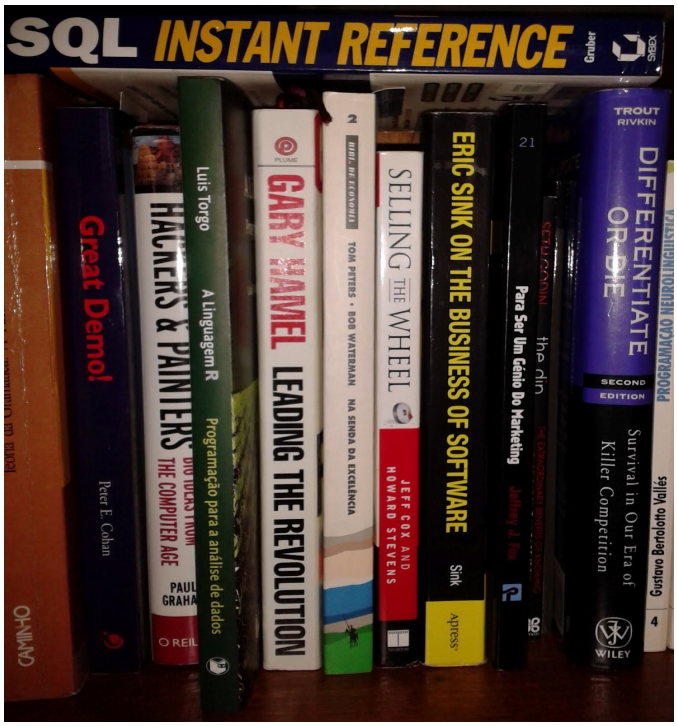
Não é durante as aulas, e muito menos durante o horário de trabalho, que vão conseguir aprender algo mais do que os vossos pares. Por isso, quem se procura destacar, requer uma força de vontade e um esforço adicional que vêm à custa do seu próprio tempo.

Costumo dizer que a minha segunda metade do dia começa depois de jantar. Para alguns parece cómico, para outros parece ridículo, e para quem procura ir mais além parece uma realidade da qual nunca se irão ver livres.

Se pensarem um pouco, verão que no ambiente em que estão envolvidos recebem a mesma atenção, os mesmos recursos e os mesmos estímulos que os vossos pares, pelo que evoluirão, mais ou menos, da mesma forma e ao mesmo tempo. É certo que numa carreira profissional alguns terão as suas especialidades, mas mesmo aí, sem um esforço adicional pessoal, serão sempre apenas mais um.

Para ultrapassar esta barreira, torna-se necessário recorrer a formação adicional ou ativar a capacidade autodidata de cada um e aprender novas tecnologias, paradigmas, modelos de gestão e tudo mais o que gira à volta do mundo das TIs. Se existe uma constante no mundo das TI essa constante é precisamente a mudança. Basta ver a evolução rapidíssima que tivemos desde o aparecimento dos primeiros computadores pessoais até ao aparecimento dos tablets. Passámos de um enorme computador em casa com uma capacidade perfeitamente ridícula para a atualidade, para dispositivos que transportamos na mão e que nos mantêm ligados ao mundo de tal forma que, em alguns casos, começam a substituir o computador tradicional.

Estamos numa época do ano especial, mas estamos também numa época que fervilha de inovação e que nos permite aprender, inovar e distinguir dos nossos pares de inúmeras maneiras. Desta forma, estaremos numa posição mais forte quando nos for apresentado um novo desafio profissional, mesmo que seja sobre algo que nos é desconhecido, temos a confiança necessária para saber que somos capazes de aprender e assim abraçar esse desafio sem medo.



AUTOR

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

Kernel Panic:

A importância da formação no ensino superior numa carreira dentro da área de segurança informática

Nos dias que correm, é comum ver-se na televisão múltiplas entrevistas e notícias de pessoal que acabou os seus cursos superiores e não consegue encontrar emprego.

Todas as opiniões expressas neste artigo são baseadas na minha opinião e percurso de vida/académico.

Ao longo deste artigo irei usar múltiplas vezes a expressão "Infosec", esta expressão é uma união das palavras Information Security e é uma maneira mais rápida de se dizer Segurança da Informação.

Este mês, no Kernel Panic , pretendo responder a múltiplas questões:

É importante ter um curso superior para se entrar na área de segurança da informação?

Quais as qualidades necessárias para se trabalhar nesta área?

Quais as melhores certificações a obter e porquê?

O que posso fazer para melhorar o meu currículo de maneira a conseguir diferenciar-me dos outros alunos que se vão licenciar no mesmo ano que eu?

Quais as principais empresas em Portugal para trabalhar na área de segurança?

Há pouco tempo foi realizado um estudo por um especialista de segurança - Robin Wood (@digininja) – que consistiu na criação de um questionário destinado a profissionais de segurança. O questionário era constituído pelas seguintes perguntas:

Há quantos anos trabalha em segurança?

Tipo de emprego?

É necessário saber programar ?

a. Se sim, qual a linguagem que recomenda?

Existe alguma utilidade para as certificações?

b. Se sim, quais?

Como é que iniciou a sua carreira em Infosec?

O que sabe actualmente que gostaria de saber quando começou a sua carreira em Infosec?

Que conselho daria a alguém que está a iniciar a sua carreira nesta área?

Quais os tópicos que considera que irão estar no topo dos problemas este ano ?

Existe algo que, na sua opinião, seja errado e que aconselha alguém a não fazer?

Acha que existe algum problema em praticar nos sites de empresas ou que não nos pertençam desde que não ocorra nenhum dano?

Vale a pena participar em conferências?

c. Quais as conferências que recomenda e porque?

Este foi, na minha opinião, um bom conjunto de perguntas, pois respondem às perguntas que são feitas normalmente por estudantes que se graduaram recentemente, e foram respondidas por pessoal com experiência na área de infosec. Podem consultar as conclusões deste estudo em [1] e [2].

Vamos então começar a tentar responder a algumas das questões que inicialmente foram apresentadas.

É importante ter um curso superior para se entrar na área de segurança da informação?

Na minha opinião, um curso superior é importante, não porque irá dar um elevado nível de conhecimento dentro da área de segurança, mas sim porque irá introduzir várias qualidades que são essenciais para um bom futuro dentro de infosec:

Metodologia: ao entrar para um curso superior, um aluno terá de aprender a ser metódico e a organizar-se. Dentro de infosec existem imensas áreas interessantes e uma pessoa pouco metódica poderá dispersar-se facilmente, acabando por não se conseguir focar apenas numa delas.

Curiosidade: esta é uma qualidade muito importante dentro de infosec. Uma pessoa curiosa irá sempre perguntar "porquê?" e irá levar um individuo a tentar fazer com que uma aplicação/ site/ objecto faça algo que normalmente não seria esperado, podendo desta maneira subverter os controlos de segurança existentes.

Investigação e filtração: esta é, por vezes, uma das qualidades mais complicadas de se obter. Dentro da área de infosec existe uma elevada quantidade de informação, alguma repetida, alguma falsa, alguma correcta e útil. Saber investigar e encontrar esta informação é importante, mas acima de tudo conseguir filtrar de modo a utilizar informação que foi obtida correctamente é algo bastante importante. Quando se faz trabalhos para a universidade, é exigido um nível de qualidade elevado, o que faz com que se procure dados e informação que vieram de fontes fidedignas.

Trabalho de equipa: esta é uma qualidade importante na medida em que, ao longo da nossa vida, por vezes teremos de trabalhar em equipa, ou até lado a lado com um cliente, e torna-se fundamental saber ouvir a opinião dos outros e saber explicar a nossa.

Todas estas qualidades são tão importantes como ter conhecimento técnico dentro da área de segurança. Não é muito útil ter conhecimento técnico se depois não o conseguirmos aplicar correctamente.

Quais as qualidades necessárias para se trabalhar nesta área?

Já mencionámos anteriormente algumas das qualidades não -técnicas, vamos agora falar das qualidades técnicas que são normalmente necessárias para se trabalhar dentro desta área.

Na opinião de alguns profissionais, saber programar não é uma necessidade. Como mencionei no início do artigo, todos os conselhos e sugestões que aqui dou são baseados na minha opinião, e esta é:

Sim! É importante saber programar! Não ao nível de Linus Torvalds, mas pelo menos conseguir escrever alguns scripts de modo a poder sempre testar-se algumas P.O.C. (Proof of concept –essencialmente um protótipo de um ataque/exploit/ algo técnico que queremos testar rapidamente). " Que linguagens de programação devo saber?" é uma pergunta que aparece regularmente. Hoje em dia, é importante saber javascript - normalmente quando se audita uma aplicação web, é necessário perceber a sintaxe desta linguagem de modo a conseguir explorar correctamente um XSS. Adicionalmente, é importante saber uma linguagem na qual nos sentimos à vontade para desenvolver pequenas POCs nos vários sistemas operativos existentes. Eu, pessoalmente, utilizo Python, Ruby e C# - Python quando preciso de um pequeno script para Linux ou OS X (por vezes também recorro a BASH) e C# para ferramentas e POC's em sistemas Windows. Para percebermos quais são as outras qualidades técnicas, é importante perceber que existem diversas áreas dentro de infosec. Se formos trabalhar como auditores de segurança (e em Portugal este é o tipo de emprego que se encontra 90% das vezes), é necessário tudo o que mencionámos anteriormente e saber:

Noção geral do funcionamento Interno de sistemas operativos;

Bases fortes no funcionamento de protocolos (TCP, IP, SSL, HTTP, BGP, SMTP, SNMP, etc…);

O que constitui uma infra-estrutura e quais as falhas comuns e pontos de entrada/vectores de ataque usados regularmente.;

Funcionamento de IDS/ Firewall / dispositivos de protecção de infra-estruturas;

Muito importante: Saber como auditar uma aplicação web, saber na "ponta da língua" as múltiplas vulnerabilidades que existem em aplicações web e como as encontrar (consultar [3] para uma lista com bastante informação);

Criptografia, perceber como funcionam os protocolos de cifra que são usados normalmente, criptografia simétrica e assimétrica, troca de chaves, como implementar correctamente estes protocolos e quais as versões correctas a usar;

Por fim, é importante saber usar algumas ferramentas essenciais, como nmap, metasploit e burp suite. No entanto, é importante entender que nunca se deve depender a 100% de ferramentas e que se deve entender como tudo funciona realmente e perceber as metodologias e falhas que ocorrem normalmente.

Existem outras áreas nas quais se pode trabalhar em infosec. No entanto, em Portugal é complicado arranjar emprego focado somente nestas áreas, podendo ser encontrados, por exemplo, os seguintes cargos:

Analista de malware

Administrador de IDS/Firewall

Analista de segurança aplicacional (análise de código)

Analista forense

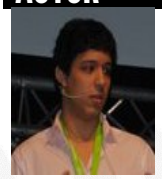
Com isto terminamos a primeira parte deste artigo. Para a edição iremos responder ao resto das perguntas na 2ª e ultima parte desde Kernel Panic .

[1] http://bit.ly/Kquqko

[2] http://bit.ly/JMJhaw

[3] http://bit.ly/bFax6s

AUTOR



Escrito por Tiago Henriques

@balgan, balgan@ptcoresec.eu - Team leader e founder da equipa de Segurança PTCoreSec, orador e organizador em multiplos eventos, nacionais e internacionais.

Para uma biografia mais detalhada www.balgan.eu <http://www.balgan.eu>

Media Partners da Revista PROGRAMAR

Análises

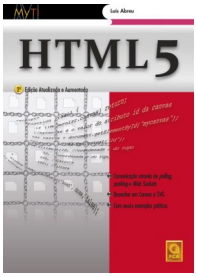
HTML 5 Segunda Edição revista e aumentada

Desenvolvimento em iOS iPhone, iPad e iPod Touch – Curso Completo

Sistemas Operativos (2. Edição Atualizada)

HTML 5 Segunda Edição revista e aumentada

Título: HTML 5 (2a Edição)



Autor: Luís Abreu

Editora: FCA - Editora de Informática, Lda.
Páginas: 336 (310 de conteúdo)
ISBN: 978-972-722-739-6

A primeira coisa que notei, mesmo antes de receber este livro, foi que não podia ser muito aprofundado: se cabia na caixa do correio, não era tão grosso quanto o habitual em livros técnicos. Fisicamente, é o normal para livros de capa mole, à excepção das páginas, que me pareceram de melhor qualidade do que o habitual. A tipografia é excelente para um livro que se proponha ser de referência: 40 linhas por página, 70 a 80 caracteres por linha, fonte com serifas leves.

Ainda antes de sair da introdução, uma das pequenas idiossincrasias deste livro já me estava a incomodar; eu percebo que o tamanho dos endereços bit.ly seja convidativo à poupança de caracteres, até entendo a racionalização que poderia ser feita quanto à quantidade de caracteres a copiar pelos leitores, mas, pessoalmente, apresentam dois defeitos inultrapassáveis: é impossível alguém decorar sequências de 6 caracteres aleatórios de relance, obrigando a copiar os endereços com uma mão no livro, outra no teclado, e impede que se reconheça um eventual endereço que já conheçamos. Várias vezes copiei laboriosamente os links para ir dar a páginas que já conhecia.

Depois de relegar esta irritação para o fundo do consciente, o resto do livro foi uma viagem relativamente suave. O estilo de escrita de Luís Abreu é extremamente acessível e percebe- se que, não só sabe perfeitamente do que está a falar, como, tirando uma ou outra excepção, também sabe como o explicar. As caixas de realce são agradavelmente relevantes e pertinentes. Os exemplos de código são de boa qualidade, se bem que preferiria um maior rigor nas boas práticas em Javascript.

O autor realça na introdução que o livro é dirigido a leitores com alguma experiência na área. Não obstante, desperdiça cerca de 20 páginas num primeiro capítulo dedicado inteiramente a fazer uma introdução ao HTML. Embora possa ser útil ao eventual novato que pegue no livro, eu preferiria que o autor se tivesse mantido fiel ao afirmado poucas páginas antes – essas 20 páginas fizeram falta mais adiante.

Outros capítulos com algum desperdício são os dedicados aos formulários (algumas explicações sobre coisas que deviam ser triviais, outras sobre coisas que não estão implementadas nem em metade dos browsers modernos) e ao SVG (que, na realidade, já cá anda há um ror de anos e só com algum esforço encontra espaço num livro sobre HTML 5 - sobretudo, a seguir ao capítulo sobre canvas).

O capítulo sobre canvas é, precisamente, onde todas estas páginas desperdiçadas deveriam ter vindo morar. O tema é tão vasto que quase merece um livro por si só e, infelizmente, a informação contida nas cerca de 45 páginas é claramente insuficiente. A não referência ao requestAnimationFrame, na secção sobre animação, foi especialmente desapontante. Salta à vista o esforço por abordar toda a temática, mas, em tão pouco espaço, ficou muito por explicar.

A segunda metade do livro tem as verdadeiras pérolas: os capítulos sobre web storage, aplicações offline, web sockets e web workers são dos melhores que já tive o prazer de ler, em português e em inglês.

Conclusão

No global, o livro é bom, sendo largamente superior ao que é normal ver-se em português, por portugueses. O preço, não sendo exactamente meigo, é uma ninharia por comparação ao que é habitual em livros técnicos.

Na minha opinião, não sendo exactamente um obrigatório, ficará muito bem na colecção de qualquer programador, curioso ou profissional.

AUTOR

Escrito por Marco Amado
Programador desde que um ZX Spectrum 48K lhe caiu nas mãos aos 7 anos. Frequentou o antigo curso tecnológico de Informática, na Escola Secundária José Estêvão, em Aveiro, e andou a ver as modas em Engenharia Informática no Instituto Politécnico da Guarda. Webdeveloper na Glups desde 2007. Escreve sobre programação, tecnologia e a vida em geral em DreamsInCode.

Curioso sobre tecnologia em geral. Trinta e poucos anos. Barbudo, a maior parte do tempo. Excesso de peso. Marido. Pai.

Desenvolvimento em iOS iPhone, iPad e iPod Touch – Curso Completo

Título: Desenvolvimento em iOS iPhone, iPad e iPod Touch – Curso Completo

Autor: Nuno Fonseca / Catarina Reis / Catarina Silva / Luis Marcelino / Vitor Carreira

Editora: FCA
Páginas: 432
ISBN: 978-972-722-740-2

Sabe bem acordar com o carteiro a tocar à campainha, a dizer que tem uma encomenda registada para entregar. Ainda melhor sabe, abrir o envelope almofadado e retirar dele, nem mais nem menos que o livro "Desenvolvimento em iOS para iPhone, iPad e iPod Touch, curso completo".

Como é a minha primeira review, fiquei logo em pulgas para ver se o seu conteúdo correspondia às minhas expectativas. Já tinha passado uma vista de olhos pelos assuntos, uma vez que consultei as informações disponíveis online. Mas realmente fantástico foi ver que o livro é simplesmente brutal .

Fazendo uma análise geral pude constatar que o livro está pensado de uma forma muito cuidada. Prevê leitores de diferentes contextos e background de conhecimentos. Está elaborado de forma a possibilitar que o leitor faça tanto uma leitura seguida, como uma consulta mais específica e aleatória.

O livro inicia com uma contextualização histórica de modo a situar o leitor no conteúdo abordado. Cada capítulo tem no seu início um pequeno sumário do que o autor pretende tratar e no final, um pequeno resumo contendo os objectivos que o leitor deverá ter atingido no final. Estes detalhes constituem, a meu ver, uma mais-valia para quem pretende uma leitura selectiva, o que permite ao leitor uma optimização do tempo, tirando todo o proveito do livro.

Podemos ainda contar com um índice detalhado, bem como um índice remissivo no final, onde podemos seguir termos e conceitos específicos e localizá-los no livro.

Este manual possui ainda a vantagem de ter todos os termos em Português, com toda a coerência do ambiente em Inglês. Todo o código está devidamente comentado e encontramos ao longo de cada capítulo, explicação de conceitos e termos-chave, bem como contextualização de cada assunto abordado, para uma melhor compreensão do leitor.

Todo o conteúdo abordado é enriquecido com imagens exemplificativas e exercícios práticos, os quais guiam o leitor, passo a passo, na criação da aplicação "O Meu Diário", através de uma exploração clara, perceptível e intuitiva. Esta aplicação permite ao leitor experimentar todas as potencialidades apresentadas em cada capítulo, aplicando-as na construção da aplicação proposta. Esta servirá apenas de modelo para que, a partir dela, o leitor dê largas à sua imaginação.

O desenvolvimento de aplicações para iOS é desta forma apresentado começando nos primeiros passos e chegando ao nível avançado. Desde a criação e desenvolvimento da aplicação, linguagem Objective-C, desenho e animação, gestos e toques multitouch, detalhes de som e imagem, georreferenciação, passando pela sua ligação às redes socias, disponibilização na rede e comercialização na AppStore, através de uma abordagem leve e fácil de explorar, o autor guia cuidadosamente qualquer ávido leitor de simples aluno a *appillionário*.

Apesar de estar a dar os meus primeiros passos na programação para aplicações móveis, penso que são manuais como este, com qualidade de conteúdo e tão completos, que incentivam qualquer leitor a uma aprendizagem interessada e leve, deixando aquele gostinho de "quero mais".

Bom proveito!

AUTOR

Escrito por Sara Santos
Professora do Ensino Básico e de Educação Musical. Deu os seus primeiros passos na programação no tempo do DOS e do dBaseIII plus. Autodidacta nas novas tecnologias, iniciou a aprendizagem em VB.Net v2005 estando actualmente a aprofundar conhecimentos através de formação certificada na área da programação web e plataformas moveis. Membro do Staff da Revista Programar desde a 30ª edição.

Sistemas Operativos (2. Edição Atualizada)

Título: Sistemas Operativos (2.ª Edição Atualizada)

Autor: José Alves Marques / Paulo Ferreira / Carlos Ribeiro / Luís Veiga / Rodrigo Rodrigues

Editora: FCA
Páginas: 752
ISBN: 978-972-722-756-3

Quando se fala em livros de sistemas operativos, o primeiro nome que nos vem à cabeça é o muito popular "Modern Operating Systems" de Tanenbaun, uma obra já adotada por muitas instituições de ensino superior nacionais e internacionais para dar apoio à unidade curricular homónima. No entanto, com a edição "Sistemas Operativos" editada pela FCA, tudo poderá mudar no mercado nacional, que até aqui, não possuía uma obra tão completa do género.

O livro Sistemas Operativos da FCA, escrito por quatro professores de prestigiadas universidades portuguesas já bem experientes no assunto, vem desafiar o império criado pela anterior obra, criando um manual bastante completo e interessante deste grande mundo dos sistemas operativos. Apesar de no dia a dia, apenas usarmos a interface, existe muito para além disso, pois este tipo de software tem como principais objetivos fazer a gestão dos recursos do computador e disponibilizar ao utilizador uma interface que permita que este tire partido da melhor forma possível dos mesmos, por isso não é difícil imaginar o grande mundo que se passa por baixo do capot.

Em termos de conteúdo, para além de abordar os conceitos essenciais da temática dos sistemas operativos de uma forma bastante acessível e concisa, como a Gestão de Processos onde para além dos termos teóricos, possui também exemplos práticos da sua implementação nos sistemas operativos baseados em Unix como o Linux e o Mac OS X, bem como do Windows, a Gestão de Memória onde são tratados de assuntos como os vários tipos de endereçamentos, segmentações e paginações, a programação concorrente onde são abordados temas como sincronizações e interblocagens, sendo também apresentados diversos exemplos e os sistemas de ficheiros, onde para além de ser dada uma excelente explicação acerca do seu funcionamento e estruturas são também dados exemplos práticos dos existentes em Windows e Linux, entre outros temas; é também dedicado um capítulo final à temática da segurança, onde são explicados alguns dos termos essenciais e como funcionam os sistemas de segurança do Windows e do Linux, abordando o funcionamento da autenticação do mesmo, criação dos *logs* de sistema e a implementação dos privilégios dos utilizadores.

Os exercícios presentes no final de cada capítulo também são de louvar, e a sua variedade na abordagem dos diferentes temas abordados em cada um dos capítulos faz com que sejam certamente um elemento indispensável na compreensão dos mesmos, no entanto penso que poderiam também ser dadas as soluções pelo menos de alguns deles como os mais práticos, pode-se também destacar os resumos existentes no final de cada capítulo que de uma forma breve e sucinta, descrevem as temáticas do seu capítulo.

Concluindo, após analisar este livro, podemos finalmente dizer que há uma obra de Sistemas Operativos em português digna de fazer frente ao peso pesado já referido.

Está na hora das universidades e cursos nacionais adotarem e apoiarem os autores de manuais técnicos portugueses, e esta é uma excelente opção para a unidade curricular de sistemas operativos.

Quanto aos utilizadores mais curiosos que querem compreender melhor o que se passa por detrás da interface em que estes interagem com o sistema como autodidactas, este livro é a escolha perfeita para eles, uma vez que, a linguagem simples e bastante explícita simplifica em muito a aprendizagem por parte destes, sendo que os resumos finais e os exercícios são um dos complementos ideias.

AUTOR

Escrito por Fábio Domingos
Finalista na licenciatura em Gestão de Sistemas de Informação na Escola Superior de Ciências Empresariais do Instituto Politécnico de Setúbal. É moderador global do fórum Portugal-a-Programar, membro do staff desta revista e por vezes contribui com soluções para a comunidade Experts-Exchange.

COMUNIDADES

PtCoreSec - Introdução-Auditoria-Passwords

NetPonto - BizTalk360 uma ferramenta de suporte e monitorização para a plataforma BizTalk Server

Introdução à auditoria de Passwords

Introdução:

A segurança das contas de utilizadores é um tema que tem preocupado cada vez mais empresas e organizações. Com os crescentes casos de ataques informáticos, o método e local de armazenamento de passwords têm-se tornado requisitos fundamentais para a proteção dos utilizadores. Casos como o do Linkedin, Yahoo, eHarmony e Blizzard, são situações que todos queremos evitar, pelo que a utilização de cifra nas passwords se torna fundamental.

Uma solução para o incremento da segurança dos utilizadores passa pela utilização de métodos de autenticação por hardware criptográfico e SSL client-side, normalmente assente em PKI, como é o caso do Cartão de Cidadão. Contudo, como esta prática ainda não é muito comum, a utilização de passwords continua a ser imperativa, como por exemplo, ao realizar login no sistema operativo, ainda antes da utilização de um certificado digital. Cabe aos responsáveis pela segurança de utilizadores, sejam eles clientes de uma plataforma web ou colaboradores de uma empresa, garantir que a probabilidade de utilização ilícita de contas seja reduzida ao máximo, nomeadamente com o recurso a cifras. Porém, é necessário ter em linha de conta que apesar dos métodos tradicionais de funções de hash terem sido eficazes no passado, estão bastante vulneráveis face ao poder computacional atual, uma vez que hoje em dia é possível adquirir hardware com capacidade de processamento elevado a um custo acessível a qualquer carteira.

Partindo deste pressuposto, este artigo procura fornecer conceitos, ferramentas e informação adicional de como auditar a força e probabilidade de comprometimento de passwords recorrendo à capacidade de processamento distribuído em chipsets gráficos. Procurámos apoiar os responsáveis pela segurança dos dados de utilizadores a precaverem-se contra os ataques atuais, sensibilizando-os para a importância de uma auditoria, gestão e educação dos utilizadores, pelo que este artigo não é vocacionado para script kiddies.

De forma a demonstrar as diferenças entre GPU e CPU, iremos realizar vários testes a uma determinada password com diferentes cifras. A verdadeira vantagem de utilização do GPU para cracking de passwords, prende-se com a facilidade de distribuição de centenas de pequenas operações pelos cores da gráfica. No CPU, pelo contrário, o limite de cores é um fator determinante na performance, pelo que é mais indicado para as situações em que a carga computacional não é relevante, ou seja, quando se tratam de pequenas operações. A título de exemplo, num processador quad-core podemos ter um processamento paralelo de 4 operações, enquanto que numa gráfica podemos ter centenas de operações em paralelo.

Após a explicação de alguns conceitos utilizados neste artigo, vamos apresentar uma prova de conceito com a identificação dos pressupostos e dos resultados obtidos.

Conceitos

Hash

Em criptografia, uma hash é o resultado de uma função de hash, que visa a criação de um sumário não reversível, de tamanho fixo, de um valor ou ficheiro. Uma vez que, se ocorrer qualquer alteração dos dados, o valor da hash será alterado, as hashes são maioritariamente utilizadas como garantia de integridade de ficheiros, de assinaturas digitais e de passwords.

Existem diversos tipos de funções de hash, diferenciados pela relação entre o processamento e a força da hash produzida, como os algoritmos MD5 (Message-Digest Algorithm) e SHA (Secure Hash Algorithm). A força da função de hash é definida tanto pelo processamento necessário como pela probabilidade de ocorrência de duas ou mais hashes iguais para diferentes valores ou ficheiros, chamada de Colisão de Hashes. Estas colisões simplificam o processo de encontrar a uma password correcta, porque afinal de contas nem sempre há real interesse em saber qual a password original, mas sim em que a sua utilização conceda a permissão desejada.

Salt

O salt é utilizado por um sistema para introduzir complexidade a uma hash e/ou a um determinado valor, como por exemplo, adicionando um caractere a uma password onde seguidamente será aplicada uma função de hash. Desta forma, e com a adição de apenas um caractere alfa-numérico, tornamos

62 vezes mais complexa a tentativa de obtenção da password original.

Para uma melhor compreensão da utilização de salt, tomemos a título de exemplo a criação de uma password de 6 caracteres alfanuméricos. Do ponto de vista da obtenção da password, temos como caracteres possíveis A-Z, a-z, 0-9, ou seja, 62 combinações possíveis para cada caractere utilizado.

A nossa password terá então: 62*62*62*62*62*62 = 62^6 caracteres possíveis, portanto, aproximadamente 57 mil milhões de combinações.

No caso de adição de mais um caractere como salt, implicaria 62^7, logo 62 vezes mais complexa a sua obtenção do que no exemplo anterior.

Técnicas de Auditoria

Tal como já dito anteriormente, é cada vez mais importante garantir a integridade e segurança dos dados dos seus utilizadores. Para tal, é importante estar ciente dos ataques mais conhecidos passíveis de serem utilizados, como o Ataque de Força Bruta e o Ataque de Dicionário. Um ataque é composto pelos métodos utilizados por um auditor para obter/recuperar, por exemplo, uma password, após interceptar dados cifrados. Um desses métodos passa por lançar um Ataque de Força-Bruta, no qual o analista tenta cada combinação possível até conseguir decifrar alguma parte da mensagem. Se a chave secreta é a hash de uma palavra, é possível lançar um ataque de dicionário e tentar somente palavras comuns de um idioma.

Ataque de Força Bruta (Brute-force Attack)

O Ataque de Força-Bruta, também conhecido como Brute-Force Attack, implica o teste de todas as combinações possíveis sobre uma determinada cifra até obter garantia de sucesso. Na prática, é um processo lento em que as combinações possíveis podem atingir valores na casa dos milhões ou mais. Com estes valores, por vezes torna-se impossível de atingir um resultado satisfatório em tempo útil.

Ataque de Dicionário (Dictionary Attack)

O Ataque de Dicionário é uma das técnicas usadas com o objectivo de quebrar a protecção de um sistema, como é o caso da autenticação através de password. Para iniciar esta técnica, o auditor deve criar um dicionário, vulgarmente chamado wordlist, que contém palavras encontradas em dicionários, como no da língua portuguesa, e/ou passwords típicas de serem utilizadas. Este dicionário permite sistemáticas tentativas, palavra a palavra, com o objectivo de mais facilmente quebrar a autenticação do sistema. Deste modo, é demonstrado que o uso de passwords normalmente utilizadas, denominadas fracas, são um mecanismo vulnerável ao tipo de ataque anteriormente descrito. Em comparação com o Ataque de Força-Bruta, este método é mais rápido, contudo, não garante o sucesso da obtenção da password visto estar limitado ao dicionário utilizado.

Prova de conceito

De forma a demonstrar os tópicos referidos neste artigo serão executados diversos testes e benchmarks.

Para garantir resultados em tempo útil, iremos recorrer a testes comparativos com CPU. Foi utilizada por omissão uma password de 6 caracteres alfanuméricos (apenas minúsculas). Deste modo, obtemos um universo de 36 combinações (a-z + 0-9) para cada um dos 6 caracteres, implicando 36^6 ou seja, aproximadamente 2 mil milhões de combinações possíveis.

Foram utilizadas as funções de hash mais conhecidas e utilizadas, nomeadamente as de MD5, SHA-1, SHA-256, SHA-512 e MD5 com aplicação de salt na password.

Apresentamos de seguida todas as componentes utilizadas para a realização dos testes, como o hardware, sistema operativo e software utilizado.

Hardware

Máquina 1

Processador: Intel Core Quad i5 3550 3.30Ghz Skt1155 6MB

Memoria: 2x 4GB (8GB) DDR3 GSkill PC3-14900 (1866MHz) CL9 Sniper

Motherboard: Asus Sabertooth iZ77

Disco: Western Digital SATAIII 500GB 7200RPM 32MB Black 6GB/s

Gráfica: Asus GeForce GTX560 Ti DC2 1GB GDDR5

Máquina 2

Processador: AMD FX-8120 Eight-Core

Memoria: 2x4GB (8GB) DDR3 Kingston Hyperx (1600Mhz) CL9

Motherboard: Asus Sabertooth 990FX

Disco: Samsung HM160HI SATAII 160GiB 5200RPM 8MB

Gráfica: ATI Radeon HD 5770 1GB DDR5

Sistema Operativo

Para a realização dos testes optámos pelo Ubuntu 12.04 LTS (Precise Pangolin) 64bit, tendo esta escolha recaído exclusivamente no facto desta distribuição ser das mais amplamente utilizadas e, como tal, abrangendo um maior número de leitores deste artigo.

Este sistema operativo detecta e instala automaticamente as drivers proprietárias da gráfica, essenciais para a ferramenta apresentada. Os drivers podem também ser encontrados no site oficial das gráficas em questão.

Hashcat

Actualmente intitulado como o password cracker mais rápido, o hashcat foi novamente o vencedor do desafio "Crack Me If You Can" na Defcon 2012, competição que visa o aumento da performance, métodos e técnicas de password cracking. O hashcat permite o processamento em várias threads utilizando CPUs, mas o seu verdadeiro poder advém da utilização do processamento paralelo facultado pelos GPUs. Outra vantagem indiscutível desta ferramenta prende-se com a sua extrema simplicidade de configuração e utilização, facilmente perceptível no decorrer do artigo. Pelos testes efectuados, os resultados são verdadeiramente impressionantes e a sua utilização torna-se obrigatória para quem é responsável pela segurança de utilizadores, onde as passwords continuam a ser um recurso obrigatório.

Existem várias opções dentro da família hashcat:

- Hascat
  - Orientado para a utilização de processadores
  - Possui a maior variedade de funções de hash
- oclhashcat_lite
  - Orientado para a utilização de gráficas
  - Optimizado para lidar apenas com uma hash
  - Possui a menor variedade de funções de hash
  - É o mais rápido devido à sua especificidade
- oclhashcat_plus
  - Orientado para a utilização de gráficas
  - Optimizado para lidar com listas de hashes
  - Maior variedade de funções de hash relativamente ao oclhashcat_lite

No decorrer dos testes foram utilizadas as diversas variantes hashcat. Esta situação prendeu-se com o facto de existir a necessidade de avaliar a performance de CPU como de GPU, sendo que nem todas as funções de hash são suportadas por todas as ferramentas.

No caso dos testes de processador, apenas foi utilizado o hashcat. rRelativamente à componente gráfica, o oclhashcat_lite suportou todas as funções de hash com a excepção de MD5 com salt, tendo sido utilizado neste caso o oclhashcat_plus.

Instalação

```
~$ sudo apt-get install p7zip

~$ wget http://hashcat.net/files/hashcat-0.40.7z

~$ wget http://hashcat.net/files/oclHashcat-lite-
                                              0.10.7z
~$ wget http://hashcat.net/files/oclHashcat-plus-
                                          0.081-64.7z
~$ p7zip -d hashcat-0.40.7z

~$ p7zip -d oclHashcat-lite-0.10.7z

~$ p7zip -d oclHashcat-plus-0.081-64.7z
```

Utilização

```
# hashcat
~$ cd ; hashcat-0.40/
~$ ./hashcat-cli64.bin --help
# hashcat-lite
~$ cd ; cd oclHashcat-lite-0.10/
~$  ./cudaHashcat-lite64.bin --help
# hashcat-plus
~$ cd ; cd oclHashcat-plus-0.08/
~$ ./cudaHashcat-plus64.bin --help
```

Nota: Os comandos anteriores estão adequados para processadores gráficos Nvidia, no caso de utilização de GPUs ATI o termo "cuda" deverá ser alterado para "ocl"

Procedimento de Testes

Para a execução dos testes foram acordadas as seguintes etapas:

- Os testes seriam realizados numa instalação base do sistema operativo (sem updates/upgrades)
- A password a utilizar seria "pa55wd" (sem aspas)
- Sobre a password escolhida seriam aplicadas as funções de hash MD5, salted MD5, SHA-1, SHA-256 e SHA-512
- Após a hash gerada é aplicada a instrução hashcat correspondente
- Este processo é repetido para cada tipo de hash

Criação das Hash

```
# md5
~$ echo -n pa55wd|md5sum|awk {'print $1'} > file
2801c4706f6f1f8c42ab607dd362ffc1

# sha1
~$ echo -n pa55wd|sha1sum|awk {'print $1'} > file
1dea63c73f8f3d4b46e0dcb82c5673fa17f52a2d

# sha256
~$ echo -n pa55wd|sha256sum|awk {'print $1'} >
file
57846ba7f6f13c0454ebec6c9b31c7f52fe72afb606a8ba336
170168ca0c408f

# sha512
~$ echo -n pa55wd|sha512sum|awk {'print $1'} >
file
7300c57b73488820013ed81d8f37c264e5f08515dcc06f4ce8
25c65b17e8e1b500d1a4d8c7d539add52bd009a4e7fe0c33b4
a799bf622f1d772b0fffe2223602

# unix salted md5
~$ mkpasswd -m md5 -S zAUw4nfB pa55wd > file
$1$zAUw4nfB$q6JVL3xTN6MRPvfK361y1/
```

Instruções Utilizadas para GPU

```
# md5
~$ time ./cudaHashcat-lite64.bin -m 0 file -1 ?l?
d ?1?1?1?1?1?1

# sha1
~$ time ./cudaHashcat-lite64.bin -m 100 file -1 ?
l?d ?1?1?1?1?1?1

# sha256
~$ time ./cudaHashcat-lite64.bin -m 1400 file -1 ?
l?d ?1?1?1?1?1?1

# sha512
~$ time ./cudaHashcat-lite64.bin -m 1700 file -1 ?
l?d ?1?1?1?1?1?1

# unix salted md5
~$  time ./cudaHashcat-plus64.bin -a 3 -m 500 file
-1 ?l?d ?1?1?1?1?1?1
```

Legenda

-a = Tipo de ataque a utilizar, 3 representa Brute Force

-m = Tipo de hash, por exemplo 1700 representa sha-512

file = Ficheiro resultante da criação da hash, contendo apenas uma linha com a hash a processar

-1 = Definição de quais os tipos de caracteres a utilizar

?l = a-z (gama de letras minúsculas, 26 alternativas)

?d = 0-9 (gama de dígitos, 10 alternativas)

?1 = Representa um caractere composto pela gama definida anteriormente com a flag "-1"

Nota: Os comandos anteriores estão adequados para processadores gráficos Nvidia, no caso de utilização de GPUs ATI o termo "cuda" deverá ser alterado para "ocl"

Instruções Utilizadas para CPU

```
# md5
~$ time ./hashcat-cli64.bin -n 4 -a 3 -m 0 file -
1 ?l?d ?1?1?1?1?1?1

# sha1
~$ time ./hashcat-cli64.bin -n 4 -a 3 -m 100 file
-1 ?l?d ?1?1?1?1?1?1

# sha256
~$ time ./hashcat-cli64.bin -n 4 -a 3 -m 1400 file
-1 ?l?d ?1?1?1?1?1?1

# sha512
~$ time ./hashcat-cli64.bin -n 4 -a 3 -m 1700 file
-1 ?l?d ?1?1?1?1?1?1
```

```
# unix salted md5
~$ time ./hashcat-cli64.bin -n 4 -a 3 -m 500 file
-1 ?l?d ?1?1?1?1?1?1
```

Legenda

-n = Número de threads a utilizar, no caso do processador AMD foram utilizadas 8

-a = Tipo de ataque a utilizar, 3 representa Brute Force

-m = Tipo de hash, por exemplo 1700 representa sha-512

file = Ficheiro resultante da criação da hash, contendo apenas uma linha com a hash a processar

-1 = Definição de quais os tipos de caracteres a utilizar

?l = a-z (gama de letras minúsculas, 26 alternativas)

?d = 0-9 (gama de dígitos, 10 alternativas)

?1 = Representa um caractere composto pela gama definida anteriormente com a flag "-1"

Benchmarks

Para existir uma percepção do tempo necessário para cada processo de auditoria, procedeu-se ao benchmark das máquinas de teste relativamente às funções de hash utilizadas. Para o efeito foi criada uma hash aleatória compatível com cada tipo de função e executada a ferramenta durante um período de trinta segundos para cada uma.

O quadro seguinte fornece a informação recolhida, os dados foram arredondados para facilitar a interpretação dos resultados:

Tipo de Hash	Máquina 1 via GPU	Máquina 1 via CPU	Máquina 2 via GPU	Máquina 2 via CPU
md5	1624 M/s	50 M/s	2692 M/s	56 M/s
sha1	420 M/s	40 M/s	943 M/s	46 M/s
sha256	148 M/s	14 M/s	283 M/s	23 M/s
sha512	61 M/s	9 M/s	40 M/s	10 M/s
salted md5	320 k/s	60 k/s	1 M/s	67 k/s

Legenda:

M/s = Milhões de tentativas por segundo

k/s = Milhares de tentativas por segundo

Cálculo para o Tempo Previsto

A tabela seguinte apresenta o cálculo relativo ao número de combinações possíveis para a password, tendo em conta o tempo obtido nos benchmarks anteriores. É de realçar que estamos a utilizar todo o universo de combinações possíveis, sendo que a colisão desejada provavelmente acontecerá antes do término das combinações possíveis. Concluímos assim que os tempos obtidos serão o máximo do tempo necessário para percorrer um universo completo de combinações.

Tipo de Hash	Máquina 1 via GPU	Máquina 1 via CPU	Máquina 2 via GPU	Máquina 2 via CPU
md5	$36^6/(1624*10^6) \simeq 1s$	$36^6/(50*10^6) \simeq 44s$	$36^6/(2692*10^6) \simeq 1s$	$36^6/(56*10^6) \simeq 39s$
sha1	$36^6/(420*10^6) \simeq 5s$	$36^6/(40*10^6) \simeq 54s$	$36^6/(943*10^6) \simeq 3s$	$36^6/(46*10^6) \simeq 47s$
sha256	$36^6/(148*10^6) \simeq 15s$	$36^6/(14*10^6) \simeq 3min$	$36^6/(283*10^6) \simeq 8s$	$36^6/(23*10^6) \simeq 2min$
sha512	$36^6/(61*10^6) \simeq 36s$	$36^6/(9*10^6) \simeq 4min$	$36^6/(40*10^6) \simeq 54s$	$36^6/(10*10^6) \simeq 4min$
salted md5 (UNIX)	$36^6/(320*10^3) \simeq 2h$	$36^6/(60*10^3) \simeq 10h$	$36^6/(1*10^6) \simeq 36min$	$36^6/(67*10^3) \simeq 9h$

Legenda:

s = Segundos

min = Minutos

h = Horas

Resultados dos Testes

Após a realização de todos os testes pretendidos, foi construída a tabela que se segue. Aqui é possível constatar o tempo de processamento efectivo para obtenção da password utilizada.

Legenda: s = Segundos, min = Minutos, h = Horas

Tipo de Hash	Máquina 1 via GPU	Máquina 1 via CPU	Máquina 2 via GPU	Máquina 2 via CPU
md5	1s	22s	1s	33s
sha1	3s	27s	3s	50s
sha256	10s	45s	5s	1min
sha512	16s	2min	28s	3min
salted md5 (UNIX)	11min	4h	3min	8h

Conclusões

A grande conclusão com que nos deparámos é que uma boa password é uma password longa. A utilização cada vez mais comum do termo passphrase faz todo o sentido e, num futuro próximo, com a evolução crescente das capacidades de processamento paralelo, apenas este modelo será relevante quando nos referimos a esta forma de autenticação. A implicação de cada caractere adicional, que resulta num aumento exponencial de tempo necessário para reverter uma hash para o valor inicial, é sem dúvida um factor determinante para a segurança da password utilizada.

Se um utilizador com equipamento "doméstico" consegue velocidades assustadoras de combinações por segundo para os métodos de hashing mais utilizados, tal como demonstrado neste artigo, imaginem Governos ou mesmo as clouds de GPUs à distância de uns quantos euros.

Outra conclusão pertinente prende-se com a utilização de salt nas passwords. Como pudemos comprovar nos testes, quando utilizamos uma função de hash MD5, garantimos uma protecção quase irrelevante da password. Contudo, ao adicionar salt à função e mesmo com o conhecimento do salt utilizado, deparámo-nos com um acréscimo brutal na segurança disponibilizada.

Ao olharmos para o recente caso do Linkedin, as hashes obtidas do dump da base de dados de utilizadores estavam em formato SHA-1, sem salt ou qualquer outra protecção adicional. Tal como pudemos comprovar, é trivial a obtenção de colisões de hashes para este tipo de função.

Existem hoje em dia outras formas de aumentar a segurança dos utilizadores, nomeadamente, através de uma segunda componente na autenticação - two-factor authentication. Uma forma acessível de utilização deste método passa, por exemplo, pelo recurso a One Time Passwords, disponíveis através de yubikeys e/ou Google Authenticator. O two-factor authentication torna-se, portanto, altamente recomendado na autenticação dos utilizadores.

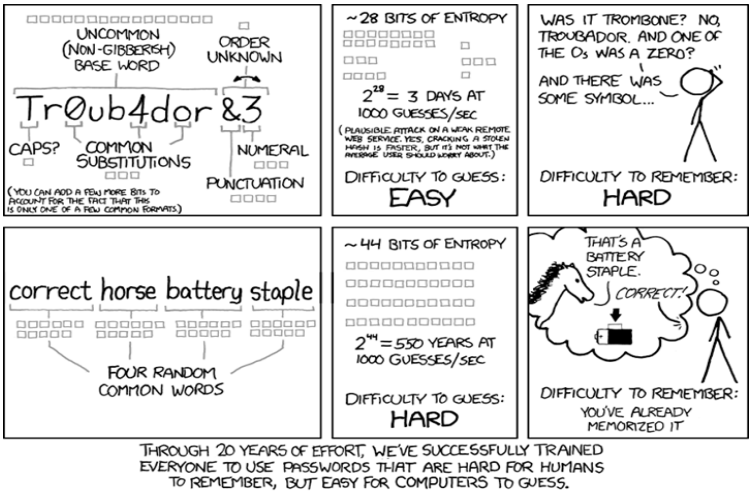
Esperamos que este artigo tenha fornecido um panorama atual no que diz respeito à auditoria de passwords e que tenha proporcionado alguma sensibilização para a importância da implementação de políticas que protejam tanto os utilizadores, como as organizações.

Por fim, não conseguimos resistir a incluir um dos web comics mais épicos da xkcd, totalmente enquadrado nos assuntos abordados:

http://xkcd.com/936/

http://hashcat.net/

https://contest-2012.korelogic.com/



AUTOR

Escrito Por Joel Bastos (@kintoandar, kintoandar@ptcoresec.eu) **e Tomás Lima** (@synchroack, synchroack@ptcoresec.eu)

Equipa PTCoreSec: http://www.ptcoresec.eu

BizTalk360

Na maioria das organizações o BizTalk Server processa dados críticos o que faz dele uma parte importante nas suas infra-estruturas, isto significa que a perda de dados ou tempo de inactividade da plataforma BizTalk pode ter um impacto significativo sobre o negócio que ele suporta. Como qualquer componente ou serviço crítico, ele deve ser administrado e monitorizado por técnicos que possuem o conhecimento e a experiência necessária. No entanto, a maioria dos administradores de sistema não está familiarizada com esta plataforma, o que pode dificultar o correcto funcionamento da plataforma e, como resultado, da própria organização.

A infra-estrutura da plataforma BizTalk Server pode variar de organização para organização, desde a utilização de um único servidor (estação standalone) ou plataformas robustas com a utilização de clusters de servidores BizTalk Server e SQL Server.

A figura abaixo mostra uma representação gráfica das diferentes camadas da infra-estrutura do BizTalk Server que deveram ser monitorizadas de forma activa por forma a permitir que a equipa de administração do BizTalk Server possa garantir a máxima disponibilidade da plataforma:

Falhas ou eventos de erro podem ocorrer em vários servidores ou até mesmo em vários locais, por exemplo: *Event viewer*, logs das aplicações, …

Devido à complexidade da infra-estrutura por detrás de um plataforma BizTalk Server, não é muito eficiente e eficaz quando um administrador tem de verificar manualmente cada servidor ou aplicação por eventos que possam ter ocorrido. Idealmente, a equipa de administração deverá fazer uso de todas as ferramentas de monitorização ao seu dispor, quer sejam elas incluídas no produto como a consola administrativa do BizTalk, *Event Viewer*, HAT ou BAM, outras ferramentas de monitorização da Microsoft como a Microsoft *System Center Operation Manager* (SCOM) ou soluções de monitorização de terceiros como o BizTalk360, por forma a facilmente monitorizar todos estes eventos e desta forma prevenir a ocorrência de falhas tomando medidas preventivas, diagnosticar problemas ou mesmo recuperar de falhas.

Estas ferramentas deverão ter a capacidade de ler eventos de todas as camadas da infra-estrutura e ajudar a equipa de administração a tomar mediadas preventivas, notificando-a quando um determinado incidente estiver para acontecer - por exemplo, quando um disco estiver com espaço livre abaixo dos 10%. Além disso, elas deverão permitir a automatização de operações quando um determinado evento ocorre, por exemplo, reiniciar um serviço quando a quantidade de memória usada por ele for superior a 200MB, prevenindo assim incidentes ou falhas, sem que seja necessário a intervenção humana.

Neste artigo pretendo demonstrar com a ferramenta BizTalk360 pode ajudar as equipas de administração a dar suporte, monitorizar e automatizar operações prevenindo assim a ocorrência de falhas na plataforma BizTalk Server.

O que é o BizTalk360

BizTalk360 (http://www.biztalk360.com/) é uma aplicação Web (RIA), concebida principalmente para efectuar monitorização e suporte aos ambientes de BizTalk Server (Produção, Testes, Desenvolvimento).

BizTalk360 vai na sua quinta versão, o que demonstra um constante crescimento ao longo destes 3 anos. É um produto desenvolvido pela empresa Kovai Limited, com sede em Londres no Reino Unido, mas todos a conhecem através do seu CTO Saravana Kumar, Microsoft BizTalk Server MVP desde 2007 e um reconhecido membro da comunidade BizTalk.

Em que se difere das outras ferramentas?

A consola administrativa do BizTalk, que é a única ferramenta incluída no produto, é muito útil com muitas funcionalidades, o que permite que esta seja utilizada para efectuar praticamente tudo na plataforma:

- Instalar, configurar e monitorizar aplicações ou adaptadores;
- Configurar *Host* e *Host Instances*;
- Configurações dos grupos de BizTalk
- Criar e configurar *Parties,* …

Podemos e devemos usá-la para diagnosticar e resolver problemas associado a aplicações de BizTalk assim como monitorizar parcialmente as camadas:

- Aplicação e Serviços
- E Plataforma BizTalk Server

Mas, como o nome indica, é uma ferramenta com foco na administração e não na monitorização, não sendo assim ideal utilizar esta ferramenta para a monitorização da plataforma como um todo, existindo algumas lacunas na ferramenta como por exemplo:

1. A falta de definição de restrições de acesso a determinados recursos ou operações, exemplo:
    a. Acesso somente de leitura a um determinado grupo ou restringir determinado grupo a visualizar e/ou gerir certas aplicações, restringir que determinado grupo visualize mensagens confidenciais ou até mesmo não permitir o acesso ou visualização das configurações da infra-estrutura (Host, Host instances, adaptadores, …) ao grupo de suporte.
    b. A falta de processos de auditoria, exemplo: Quem fez o quê.
    c. Visualização da topologia dos seus ambientes em tempo real.
2. Outra importante limitação da ferramenta é que não tem interface Web, ou seja, é necessário acesso desktop remoto às máquinas de produção para podermos usar a ferramenta, o que nem todas as organizações permitem.

Podemos também monitorizar a plataforma BizTalk Server como um todo utilizando ferramentas de monitorização robustas como o SCOM.

No entanto, usar apenas o SCOM para monitorizar a plataforma BizTalk poderá tornar-se extremamente caro para as médias empresas ou até mesmo para algumas grandes empresas. Uma vez que a curva de aprendizagem para usar esta ferramenta é grande, ela é demasiado completa e complexa, esta requer pessoas altamente qualificadas. Para além da necessidade de recursos especializados, também requer um investimento grande a nível de hardware e software, uma vez que o SCOM pode monitorizar qualquer produto da Microsoft, Azure ou mesmo produtos de terceiros. Seria um desperdício de recursos usar apenas o SCOM para monitorizar uma plataforma de BizTalk Server.

Saravana Kumar utilizou a experiencia adquirida ao longo de muitos anos como consultor de BizTalk em diversos clientes para criar a ferramenta BizTalk360 (http://www.biztalk360.com) que aborda as questões acima e muitas outras mais, preenchendo quase todas as lacunas deixadas Microsoft na área de suporte e monitorização do BizTalk Server de um forma incrivelmente simples e intuitiva, respondendo assim às necessidades mais comuns dos clientes e ao mesmo tempo que não existisse uma curva de aprendizagem íngreme para utilizar esta ferramenta comparadas com ferramentas como o SCOM.

- São várias as funcionalidades que esta ferramenta nos apresenta, no entanto podemos destacar as seguintes como senda as principais:
- É uma aplicação Web (RIA - Rich Internet Application), o que ao contrário da consola administrativa do BizTalk, que necessita de acesso aos ambientes de BizTalk, nos permite aceder de qualquer lugar.
- Capacidades avançadas de gestão de permissões de acesso aos mais variados recursos do BizTalk;
- Possibilidade de realizar auditoria em todos os níveis, mensagens e eventos;
- Disponibilização de Dashboards de monitorização muito completos e detalhados que cobrem as necessidades mais frequentes
- Capacidades de monitorização dos mais variados recursos e notificação proactivas.
- Integração com o BAM Portal, Message Box Viewer e com outras ferramentas como HP Operation manager Integration
- Suporte multi-ambiente e capacidade de configurar todos os ambientes BizTalk através de um local único.
- Repositório da base de conhecimento
- Visualizador avançado de eventos (Event Viewer)

Capacidades avançadas de gestão de permissões

É imperativo que as organizações tenham a possibilidade de gerir permissões de acessos aos mais variados recursos nas plataforma de Microsoft BizTalk Server. BizTalk360 torna este pressuposto possível, trazendo incluída no produto a capacidade de gestão de permissões totalmente personalizáveis que permitem às organizações definir os seus próprios requisitos de autorização. Estas

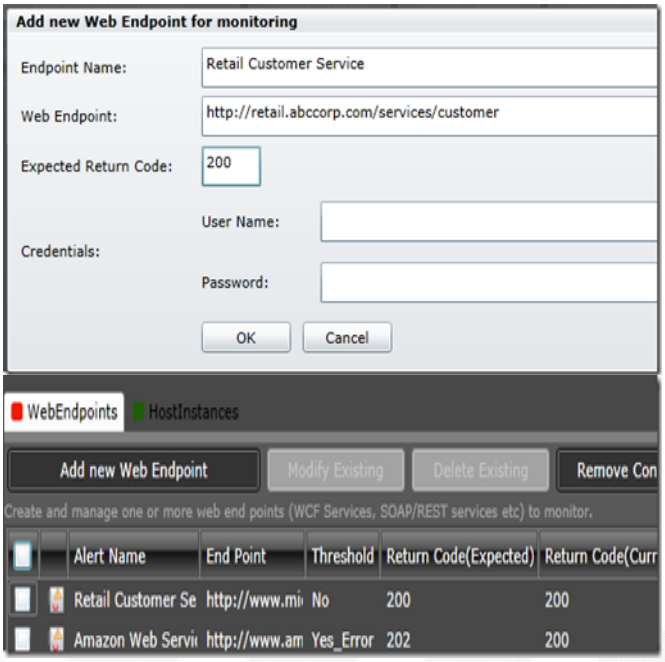
funcionalidades ajudam a equipa de administração do BizTalk a configurar devidamente os acessos ao ambiente para os vários departamentos ou equipas de projectos, sem o medo que estas interferiram com outros recursos.

Capacidade de Monitorização e Notificação de alertas

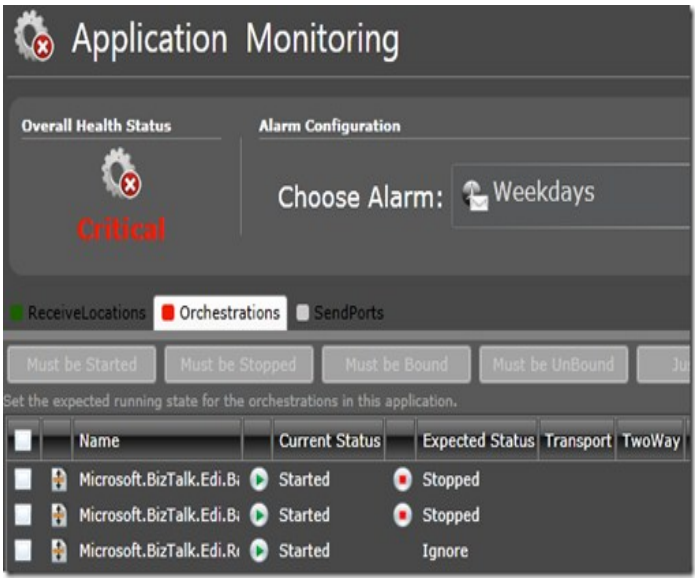
BizTalk360 permite-nos monitorizar os mais diversos recursos associados às diferentes camadas da infra-estrutura do BizTalk Server, nos quais podemos destacar:

Aplicações e Serviços

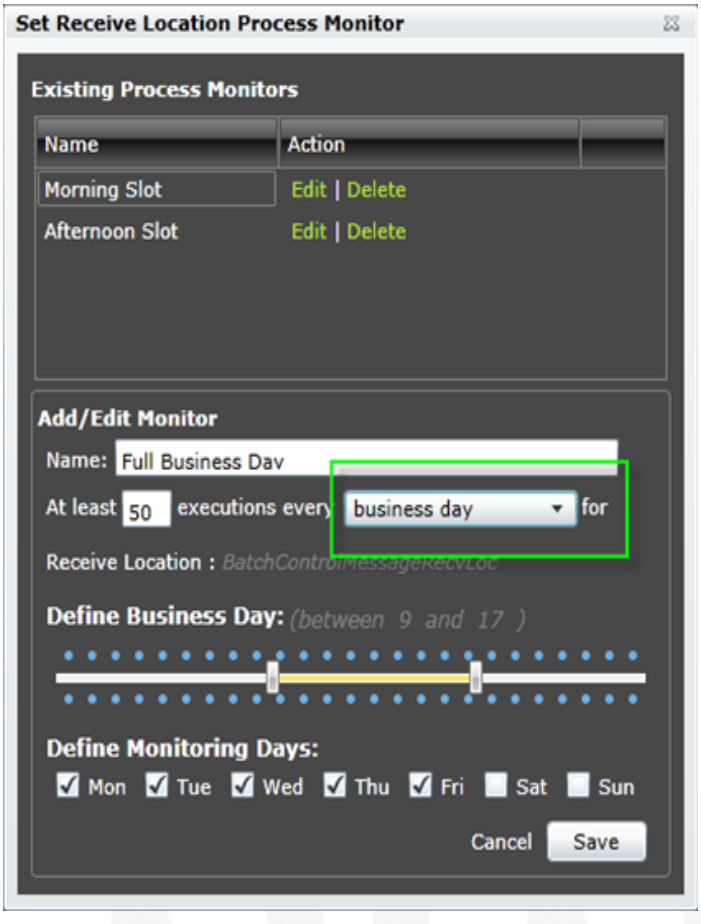
Permite a monitoração de endereços HTTP ou serviços Web externos para códigos de erros conhecidos (ex: 200, 202, etc.).



Monitorização dos estados (Started, Stopped, Enlisted, Un-enlisted, Enabled ou Disabled) das orquestrações, portas de envio (send ports) ou locais de recepção (Receive locations)

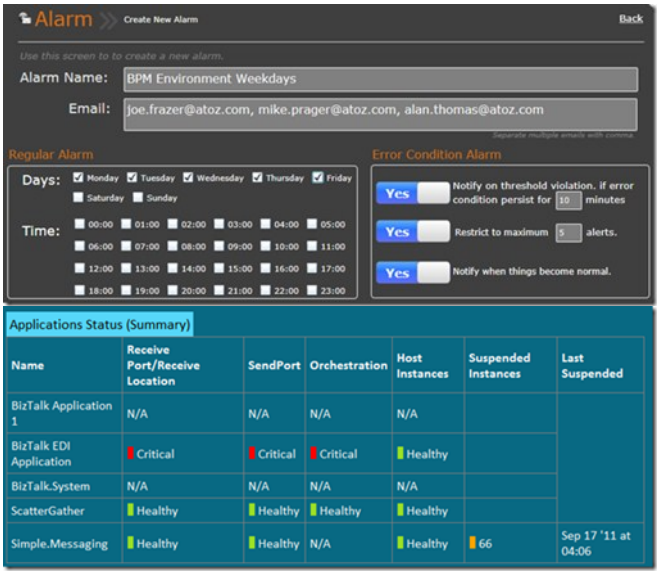


Monitorização de processos, como por exemplo estarmos à espera de, receber ou enviar, uma certa quantidade de mensagens de ou para um determinado ambiente. Se esta regra não for cumprida, então deverá existir uma implicação no negócio e alguém precisa ser notificado.



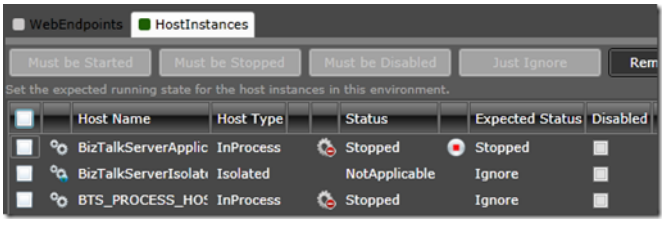
BIZTALK360

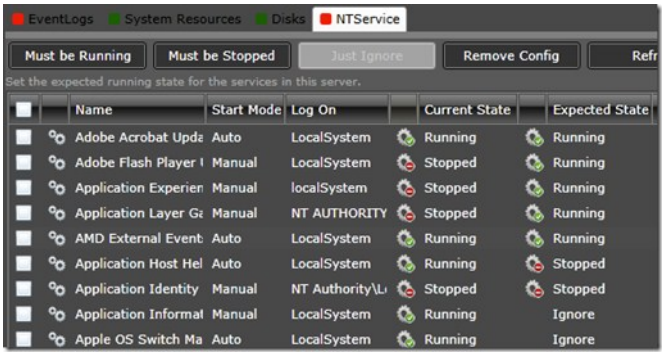
Monitorização automática de falhas ou violações, controlando a forma como estas nos são notificadas: se é um problema intermitente, aguarde 10 minutos antes de enviar alerta; enviar apenas 5 alertas em vez de 100 alertas nas próximas 4 horas; se alguém corrigir, podemos também configurar o sistema para nos notificar. Podendo definir o envio destas notificações por correio electrónico ou SMS.



Plataforma BizTalk Server

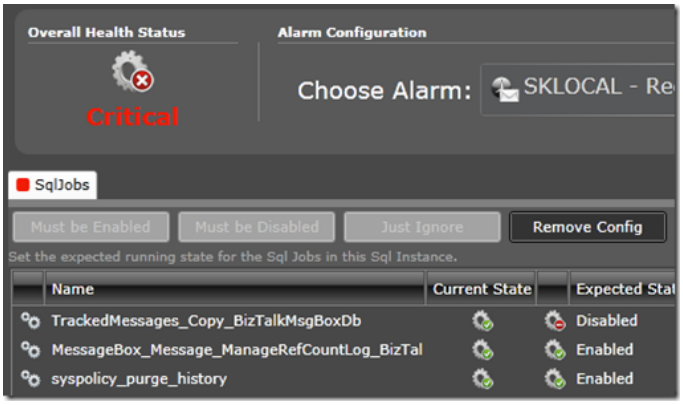
Monitorização do estados das Host Instances, SQL Jobs ou NT Services. Para cada um deles também podemos definir o estado espectável, este ponto é extremamente útil por exemplo para a monitorização dos SQL jobs que vem com o BizTalk, uma vez que não é suposto estarem todos activos (MessageBox_Message_Cleanup_BizTalkMsgBoxDb deverá estar desactivo e ser activado pelo job MessageBox_Message_ManageRefCountLog_BizTalkMsgBoxDb)



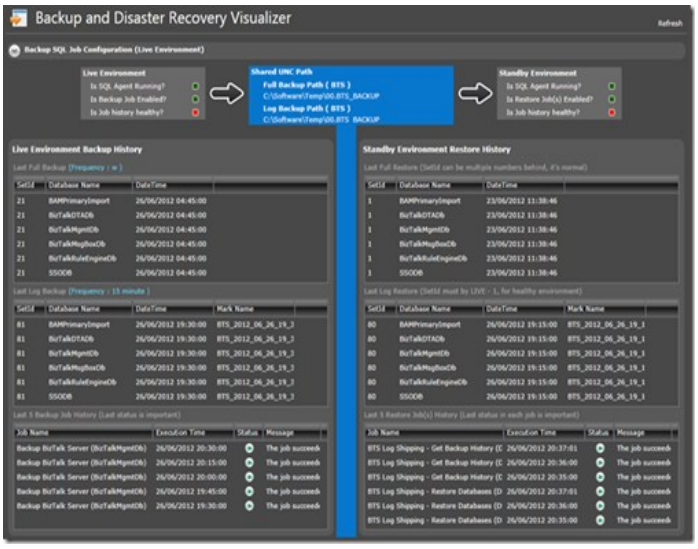


O SQL Server e em especial a base de dados "*MessageBox*" são o coração da plataforma BizTalk Server. E é por essa razão que a Microsoft normalmente trata o SQL Server como uma caixa preta quando se trata de BizTalk Server, é algo que nós sabemos que existe, mas que não deveríamos mexer! A plataforma traz incluída no produto uma série de *SQL jobs*: *backups*, limpeza, … que garantem o perfeito funcionamento da plataforma. Na verdade, e ao contrário do que muitos DBA utilizam, a única forma de garantir um *backup* integral e que possibilite o seu restauro fidedigno é utilizando estes SQL Jobs e não outros processos normalmente criados para essa funcionalidade.

A fim de garantir que todas estas actividades, em especial os backups, estão a funcionar correctamente, a equipa de administração de BizTalk ou DBA precisa de monitorizar periodicamente estes Jobs. BizTalk360 ajuda-nos a resolver todos esses desafios fornecendo mecanismos de controlo e notificações:



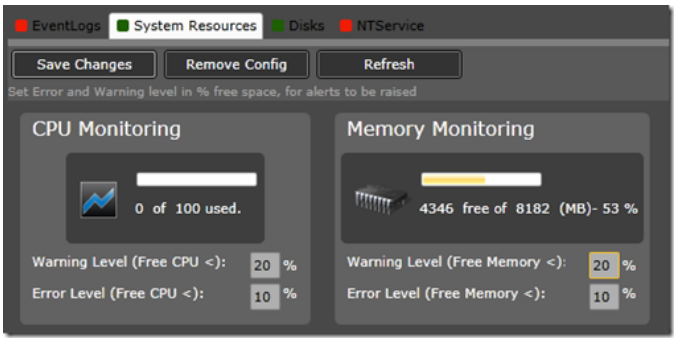
Assim como um painel de visualização de "BizTalk Backup and Disaster Recovery":



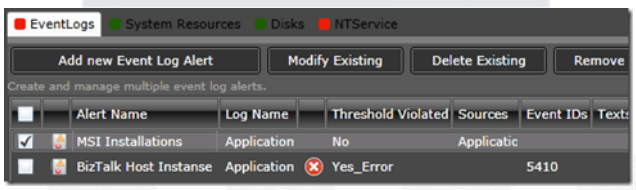
Capacidade de monitorizar o estado das instâncias (Service Instances), permitindo por exemplo definir diferentes níveis de alerta e/ou erro para um determinado número de serviços para os diferentes estado (Suspended, Active, Scheduled etc), notificação caso os mesmos sejam atingidos.

Sistema Operativo

Capacidade de monitorizar a utilização dos CPU's ou consumo de memória, permitindo configurar níveis de alertas de erro e/ou de aviso relativo à quantidade de memória livre ou ao consumo do CPU. Por exemplo, enviar uma notificação se a memória livre ficar abaixo dos valores definidos durante 30 minutos ou se o consumo do CPU se mantiver acima dos valores definidos durante um período de tempo.

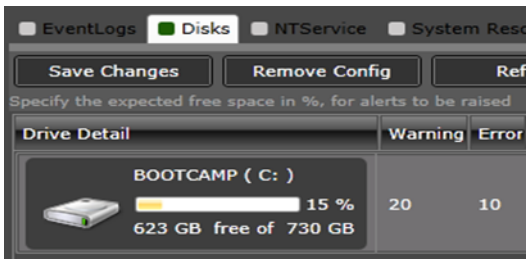


Monitorização de eventos de logs, permitindo a parametrização de diferentes notificações para diferentes consultas dos logs, por exemplo quero receber um alerta caso o número de erros associados ao BizTalk seja superior a 10 nos últimos 30 minutos.



Hardware do Servidor

Capacidade de monitorizar os discos rígidos, permitindo uma vez mais o envio de notificações caso o espaço livre nos discos atinja os valores definidos.



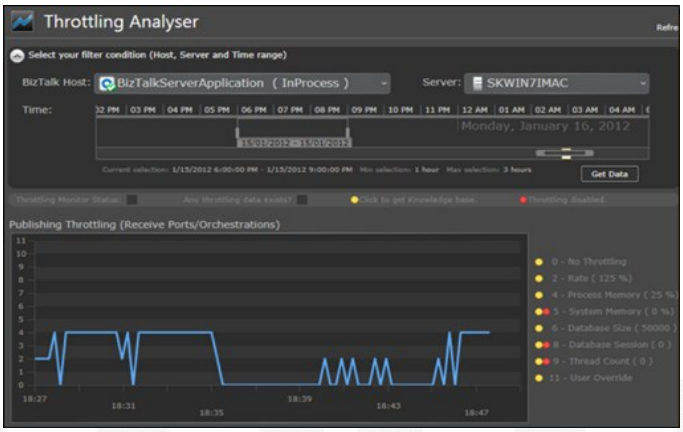
Abstracção da complexidade de certas tarefas

Devido a várias razões, quer seja por não ter trabalho suficiente para manter os recursos ocupados, não justificando assim ter uma equipa dedicada ou simplesmente por razões financeiras, poucas empresas têm uma equipa dedicada na administração da plataforma BizTalk Server. Desta forma, as organizações tendem a agrupar estas tarefas de administração numa equipa com funcionalidades alargadas a diversos sistemas ou plataformas, como administradores de sistemas, DBA's ou mesmo BizTalk.

Um dos objectivos da ferramenta BizTalk360 é abstrair a complexidade de alguns dos principais conceitos ou funcionalidade do BizTalk Server, possibilitando assim que os ambientes de BizTalk Server sejam geridos ou suportados por pessoas com conhecimentos básicos sobre BizTalk, mas sem ter de ser obrigatoriamente um especialista.

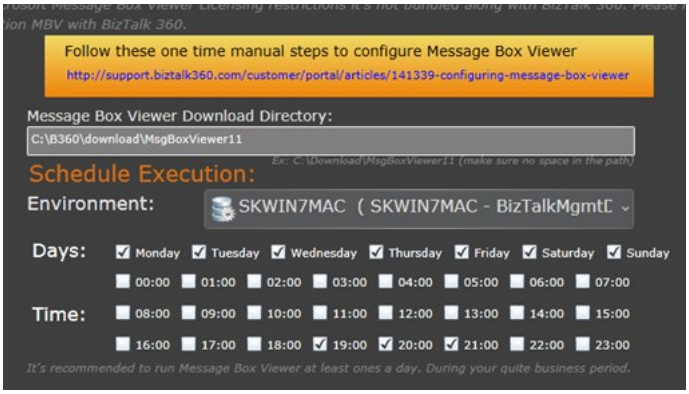
Throttling Analyser

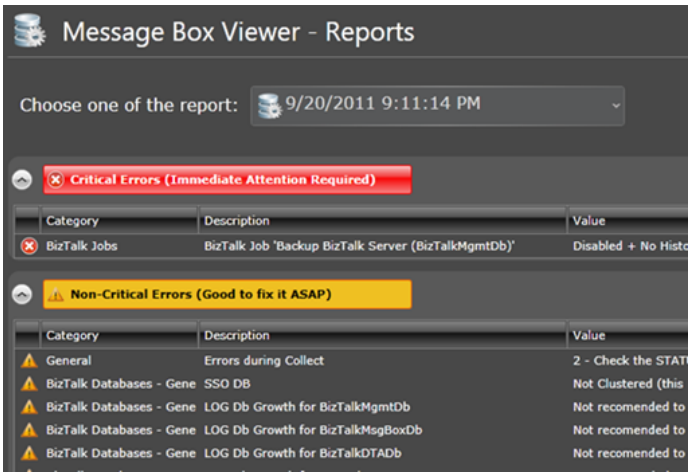
Outra das características interessantes dssta ferramenta é o Throttling Analyser. Com esta funcionalidade, BizTalk360 abstrai-nos da complexidade que é analisar estes comportamentos na plataforma, permitindo assim, de uma forma fácil e quase em tempo real, monitorizar as condições do ambiente, analisando onde estão a acontecer os constrangimentos e optimizando estas configurações sem termos um conhecimento profundo sobre BizTalk Throttling, planeando assim as suas capacidades e SLA's. Por defeito os dados de throttling dos últimos 7 dias são mantidos em histórico.



Integração com Message Box Viewer (MVB)

Message Box Viewer ou MBV é uma das principais ferramentas de suporte da Microsoft para analisar o seu ambiente de BizTalk Server e assim produzir alguns relatórios sobre o estado do seu ambiente. Ele capta todos os detalhes do sistema e produz um relatório de fácil leitura destacando erros críticos e não críticos ou potenciais problemas no seu ambiente.





Tracking manager

Sendo BizTalk Server um middleware, este inclui funcionalidades de rastreamento dos mais diversos artefactos. BizTalk360 ajuda-nos a visualizar as configurações de rastreamento ao nível das aplicações a partir de um único local.

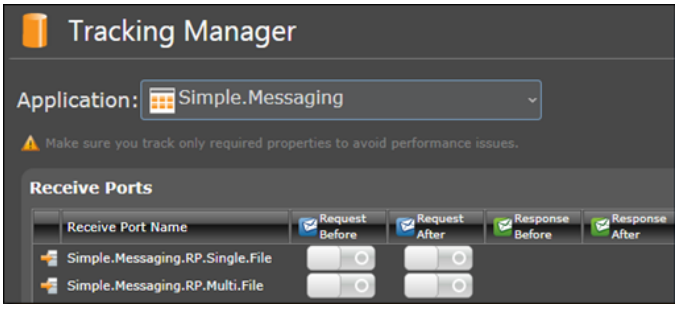


Diagrama dinâmico da topologia do seu ambiente

A maioria dos ambientes de BizTalk Server é composta por multi-servidores, tipicamente o mínimo aconselhável será ter dois servidores de BizTalk Server e dois servidores de SQL Server para suportar "alta disponibilidade". Desta forma, é importante às equipas de administração conhecerem a topologia da sua plataforma e terem acesso fácil a alterações que ocorram. BizTalk360 fornece a capacidade de apresentar uma visão gráfica da sua topologia gerada dinamicamente a partir de seu ambiente.

Ferramentas de Produtividade

Knowledge Base integrada

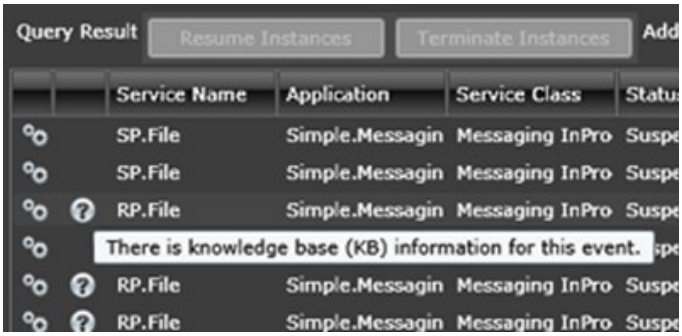
Assim que os problemas ou falhas são resolvidas, normalmente poderão acontecer duas coisas:

1. A equipa de administração documenta o problema num repositório interno (Wiki, SharePoint) da organização, ou num cenário mais pessimista nos seus blocos de notas pessoais;
2. A equipa de administração após resolvido o incidente, não efectua qualquer tipo de documentação

Muitos de vocês deverão estar familiarizados com o passo 2… O problema com estas duas abordagens é que a informação, ou não existe ou não está disponível prontamente aos utilizadores quando estes incidentes ocorrerem novamente.

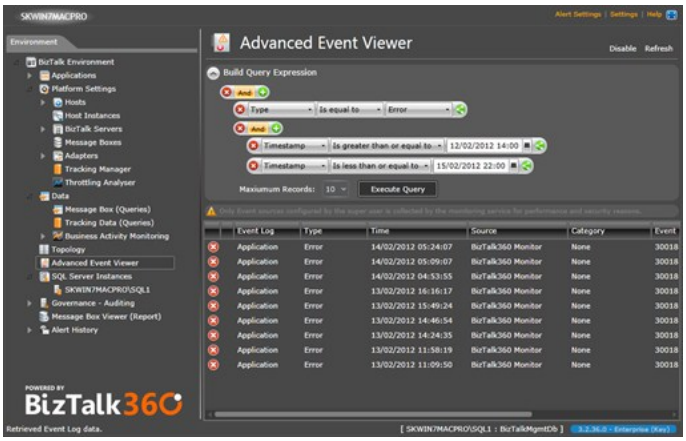
BizTalk360 aborda esta questão fornecendo um repositório *Knowledge Base* integrado, que possibilita aos utilizadores adicionar nos diferentes eventos (adicionar KB associados aos ids dos eventos do *Event Viewer* ou aos códigos de erros das *Services Instances*) informações sobre os processos

necessários para corrigir problemas. A próxima vez que o mesmo erro acontecer, o BizTalk360 irá informa-lo que existe uma base de conhecimento disponível para o problema, acelerando assim a resolução da questão.



Visualizador avançado de eventos

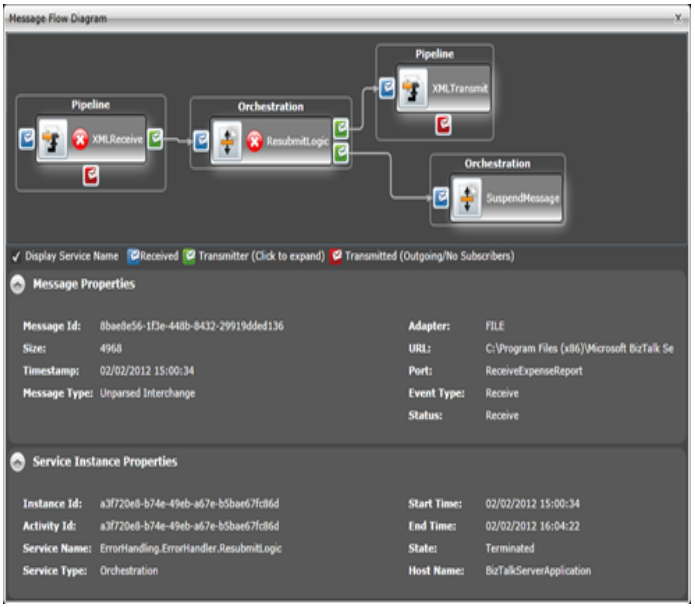
Um dos problemas comuns que as equipas de administração enfrentam quando estão a tentar monitorizar ou a diagnosticar problemas nas plataforma BizTalk Server, em especial em plataformas multi-servidores, é que os erros podem ocorrer em diferentes servidores o que nos obriga desta forma a percorre-los todos, além desta tarefa ser demorosa poderá trazer riscos de segurança, uma vez que temos de permitir que determinadas equipas tenham acesso aos servidores (o que nem sempre é permitido nas empresas). BizTalk360 aborda esta questão, fornecendo uma funcionalidade de um Event Viewer centralizado, agregando todas as informações dos diferentes logs de eventos e apresenta-a num lugar único.

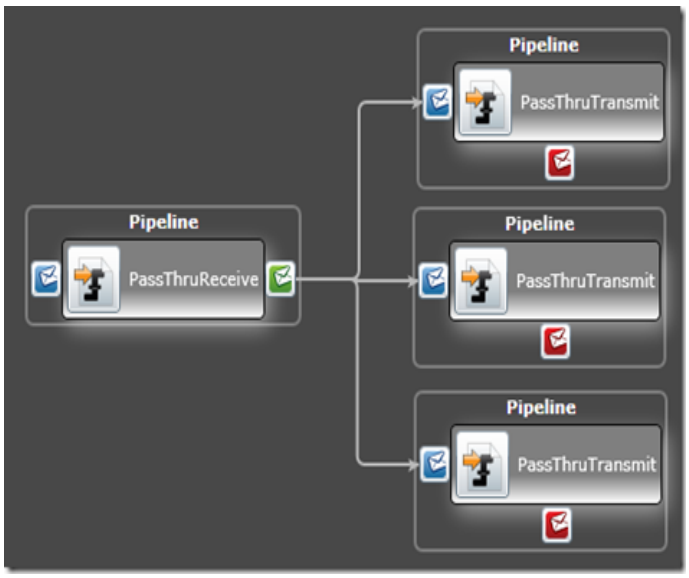


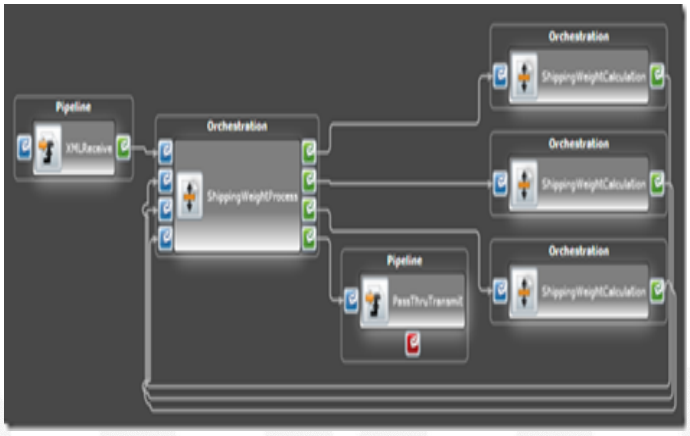
Ferramenta gráfica Message Flow Viewer

Para as pessoas encarregues de diagnosticar incidentes com limitado conhecimento sobre BizTalk, normalmente equipas de suporte, um dos maiores desafios é entender o caminho que as mensagens percorrem dentro da plataforma. De realçar que o núcleo duro da arquitectura da plataforma BizTalk é baseada em *publish* / *subscribe*. Desta forma, uma mensagem enviada para o sistema será subscrita por uma ou várias portas ou orquestrações e poderá circular internamente por diversas fases. Normalmente, este tipo de equipa de suporte para efectuarem diagnósticos a este nível deverão ter um conhecimento profundo sobre BizTalk e como usar as ferramentas de fluxo de mensagens disponível com o produto.

Mais uma vez o BizTalk360 elimina a complexidade deste problema, fornecendo um visualizador gráfico do fluxo de mensagens com base na informação de tracking do BizTalk (BizTalk tracking data).







Capacidade de Auditoria (Governance/Auditing)

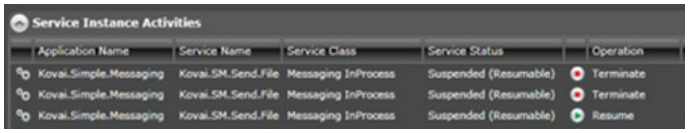
Deixo para o final, o que na minha opinião talvez seja uma das principais funcionalidade desta ferramenta e que mais nenhuma outra no mercado possibilita: a capacidade de trazer funcionalidades de auditoria à plataforma BizTalk.

Da mesma forma que é imperativo para às organizações ter a possibilidade de gerir permissões de acessos aos mais variados recursos nas plataforma de Microsoft BizTalk Server, também estas têm de ter a capacidade, muitas das vezes por requisitos regulamentares ou simplesmente por boas práticas, de saber quem anda a fazer o quê na plataforma, acompanhando assim as actividades das pessoas de suporte ou de administração nos seus ambientes produtivos.

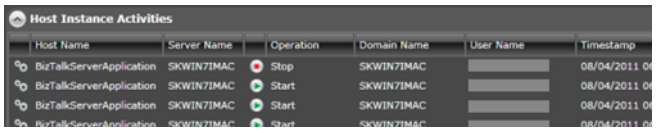
Já pensou que por exemplo alguém acidentalmente ou propositadamente terminar várias mensagens críticas de negócios (mensagens de encomendas). Como irá descobrir quem efectuou esta operação?

BizTalk360 disponibiliza capacidades de auditoria completas, registando as normais actividades operacionais para as seguintes áreas:

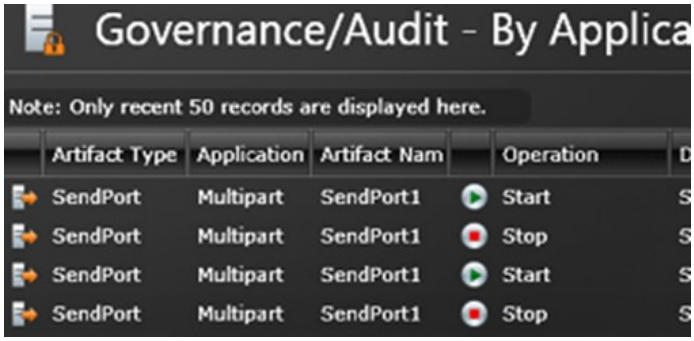
- **Operações sobre Service Instances:** registar de todas as actividades sobre as Service instances como: suspender, retomar ou terminar as instâncias.



- **Operações sobre Host Instances:** registar de todas as actividades sobre quem pára as host instances e qual o motivo ou quem as inicializa.



Operações sobre as aplicações: registar de todas as actividades ao nível das aplicações, como iniciar/parar portas de envio, orquestrações ou activar/desactivar locais de recepção (receive locations)



Conclusão

Existem várias produtos no mercado que permitem efectuar a monitorização sobre os ambientes BizTalk Server: System Center Operation Manager, HP OpenView, Minotaur, Moesion, BizTalk Processing Monitor, entre outros. Todos eles têm as suas vantagens e desvantagens, mas nenhum consegue efectuar uma monitorização completa de acordo com as necessidades reais dos clientes, o que nos obriga a termos de utilizar conjugações de ferramentas para atingir os nossos objectivos.

BizTalk360 é uma das ferramentas mais completas para efectuarmos a monitorização e suporte dos ambientes de BizTalk Server e uma das únicas que focalizou os seus esforços para preencher todas as lacunas deixadas Microsoft na área de suporte e monitorização do BizTalk Server.

No entanto, existem muitas mais funcionalidades nesta ferramenta que não foram alvo de detalhe neste artigo e que poderão descobrir utilizando o trial da versão beta aqui: http://www.biztalk360.com/Content/beta. Poderão também encontrar informação dos preços da ferramenta aqui: http://www.biztalk360.com/pricing/enterprise.

AUTOR

Escrito Por Sandro Pereira [MVP & MCTS BizTalk Server 2010]

Actualmente *Senior Software Developer* na empresa DevScope. É *Microsoft Most Valuable Professional* (MVP) em Microsoft BizTalk desde Janeiro de 2011. O seu principal foco de interesse são as tecnologias e plataformas de Integração (EAI): BizTalk e SOAP / XML / XSLT e Net, que utiliza desde 2002 e Windows Azure Service Bus. É um participante bastante activo nos fóruns da Microsoft (*MSDN BizTalk Server Forums*), contribuidor no *MSDN Code Gallery*, *Microsoft TechNet Gallery* e no CodePlex, autor no *Microsoft TechNet Wiki*, autor do **Blog**: http://sandroaspbiztalkblog.wordpress.com,

No Code

Análise: O que faz de nós bons programadores?

Falácias da Computação na Nuvem

Entrevista a João Barreto

Projecto em Destaque na Comunidade Portugal -a- Programar : NotíciasPT

Análise: O que faz de nós bons programadores?

Este artigo nasceu de uma recente discussão de que fui espectadora, e creio que a discussão se iniciou devido ao editorial do número anterior da nossa revista. E tomando a liberdade de citar António Santos:

"Nos dias de hoje, mais do que nunca existe tendência a escrever "spaghetti-code" sem grande qualidade, consumidor de enormes recursos de hardware, muitas vezes "usando e abusando" de código gerado automaticamente pelas IDEs, etc…"

A discussão, como o leitor poderá facilmente perceber, foi acerca da arte de bem programar e até que ponto somos bons e maus programadores. No fundo acho que todos os intervenientes da dita discussão estavam correctos. E foi neste contexto, como leitora assídua que sou da nossa revista, que resolvi dedicar-me a esta análise. Na tentativa de perceber, tal como o leitor, em que ponto do movimento estamos nós, em que ponto da evolução nos encontramos.

Após uma pequena pesquisa que qualquer um de nós poderia fazer, deparei-me com mais de meia centena de linguagens de programação, que podem ser encontradas de A a Z, todas diferentes entre si mas iguais no seu principal objectivo: o de ajudar o mundo a evoluir. Porque no fundo é isso que nós, programadores, tentamos fazer: construímos programas não só pela necessidade de trabalhar em troca de um salário, mas pomos um pouco de nós em cada programa que fazemos, tentando sempre ajudar o nosso utilizador final a atingir um objectivo. A nossa função é simplificar esse processo e fazemos isto em cada módulo que desenvolvemos. Se nos recordarmos da essência do que aprendemos, todos os programadores sabem que devem "dividir para conquistar". Apesar de ser um mote das linguagens de programação orientadas a objectos (como Java), todos nós aprendemos que para criarmos bons programas, devemos dividi-los em partes lógicas. E vamos resolvendo os problemas um a um, módulo a módulo, de forma a conquistarmos um todo, o objectivo que queremos, a simplicidade de uma acção.

A programação tem sempre uma vertente modular. Recordo as palavras de um professor meu, que nos ensinou que "um bom programador, é um bom preguiçoso", porque tenta, através de funções, optimizar o seu código para que seja genérico e possa ser facilmente usado em várias etapas de um projecto sem precisar de alterações. Usamos apontadores e abusamos dos objectos e das estruturas porque sabemos que nos facilitam a vida.

Mas hoje em dia os computadores a que temos acesso são cada vez mais potentes, mais rápidos e, às vezes, o típico código "esparguete" ou o "código pastilha" podem não demonstrar ao utilizador comum que estão a alocar recursos a mais. Se o programa fizer o que o utilizador espera, o utilizador nunca irá pensar no que está por trás.

Mas além da rapidez, a evolução trouxe-nos também a autonomia. E hoje em dia precisamos de programar tendo em conta o dispositivo no qual a nossa aplicação será mais utilizada (o processamento e a memória de um tablet ou de um smartphone ainda não se assemelham à memória e ao processamento de um computador convencional). Nestas circunstâncias temos sempre que nos lembrar de que não podemos alocar mais recursos do que o realmente necessário, senão corremos o risco de ninguém querer usar a nossa aplicação.

Mas voltando ao ponto fulcral do nosso artigo… com um pouco de boa vontade poderíamos definir a nossa informática como um ciclo.

Podemos falar da Máquina de Turing, com a qual Alan Turing deu inicio a uma nova descoberta criando o primeiro "computador" teórico, em que a máquina fazia qualquer cálculo que lhe fosse pedido. E passar pela Tese de Church, o cálculo Lambda, sem esquecer as funções recursivas de Kleene. Se juntarmos tudo, estamos perante o nascimento da noção de algoritmo que hoje conhecemos. Com as funções recursivas de Kleene é possível programar o cálculo Lambda e este, por sua vez, permite-nos programar e construir a máquina de Turing correspondente a qualquer problema que queiramos resolver.

No entanto, quando programamos, não é com modelos teóricos com que nos preocupamos mas sim com a causa/efeito que queremos. E procuramos ser especialistas numa determinada "família" de linguagens de programação. Escolhemos entre os diversos paradigmas (estrutural, imperativo, entre outros), passamos para a programação orientada a objectos com Java, PHP, aprofundamos o conhecimento no paradigma funcional com Ocaml, Haskell, damos uns toques nas linguagens de *markup* e de hipertexto como o HTML e CSS, e olhamos com curiosidade para o paradigma lógico com o Prolog, por exemplo. E com um pouco de pesquisa descobrimos um paradigma novo, o esotérico, que deu origem às *esolangs*. Sendo estas últimas linguagens de programação projectadas para testar os limites dos projectos de linguagem de computadores, como uma aplicação de uma teoria embora não seja possível utiliza-las

na prática. Algumas destas linguagens procuram ser "Turing completas" ou seja, equipotentes à máquina de Turing.

Mas esta forma de caminharmos é apenas um exemplo, e tal como eu, o leitor pode escolher sempre o próximo passo a dar, pois são diversos os paradigmas que podemos escolher. E nem anos de estudo ou de prática fazem com que nos possamos auto-proclamar especialistas. A informática é um mundo em constante evolução, e temos os últimos anos para nos provar isso. Deixámos de programar com cartões furados… e passámos a programar com linhas de código. Com GUIs e IDEs, e hoje em dia já temos quem programe visualmente, arrastando componentes para desenhar a sua aplicação (como acontece no NetBeans, por exemplo).

O facto de tantas pessoas contribuírem para esta nossa revista, a Programar, mostra que em Portugal são muitos os que se preocupam em evoluir, em descobrir, e atingir novos conhecimentos. Não esquecemos o passado para projectarmos o futuro.

Um informático ou programador hoje em dia tem que saber de design, tem que ser um bom gestor, tem que ter muitas vertentes. Porque não vamos desenvolver um programa que não seja agradável à vista do nosso utilizador alvo, não vamos colocar todas as opções umas em cima das outras, temos que gerir o espaço que temos disponível. Desenvolver uma ideia, um projecto e só depois implementá-lo.

Na minha opinião, um bom programador não depende apenas da linguagem que utiliza, uma vez que todos nós temos opiniões e gostos diferentes. Apesar de que programar numa linguagem de programação bem nossa conhecida é praticamente "meio caminho" andado. Um ponto importante na nossa formação é a forma como desenvolvemos um algoritmo, os passos que damos para atingir um objectivo, se programamos ou não de forma recursiva. Isso é talvez o ponto que acho que distingue os bons e os maus programadores.

E isto, caro leitor, traz-nos de volta a um novo ponto do ciclo inicial. A linguagem de programação. E o que é uma linguagem de programação?

Não adianta escrever sobre um assunto se evitamos dar uma definição do mesmo. *"Uma linguagem de programação é um método padronizado para comunicar instruções para um computador. Permite que um programador especifique precisamente sobre quais dados um computador vai actuar, como estes dados serão armazenados ou transmitidos e quais acções devem ser tomadas sob várias circunstâncias."*

Ora, se o leitor considerar uma linguagem de programação um alfabeto, sendo este um conjunto constituído por letras, que originam palavras que por sua vez originam frases que servem para comunicar e trocar informação podemos afirmar que um programa é um conjunto de regras sintácticas e semânticas usadas para definir uma acção pretendida.

O que nos leva a percorrer mais um espaço no ciclo. Sendo o nosso programa uma sequência de caracteres precisamos de um compilador que leia e interprete os nossos comandos.

Para ser um bom informático, um bom programador, há que conhecer a linguagem de programação utilizada. E qual é a melhor forma de conhecer essa linguagem? Conhecer o seu compilador. E a melhor forma de o conhecer é escrevê-lo.

A construção dum compilador envolve a utilização de vários métodos e ferramentas de análise léxica e sintáctica. A última fase da compilação é a geração de código que é realizada em várias etapas que correspondem a tradução para várias linguagens intermédias antes de se concluir pela produção de código executável. E dando a conhecer um pouco de forma rápida as principais fases da compilação:

Análise léxica é a primeira fase do compilador. A função do analisador léxico, é ler o código fonte, caractere a caractere, separar e identificar os elementos do programa fonte, normalmente chamados símbolos léxicos ou tokens. O objectivo dos geradores automáticos é limitar o esforço de programação de um analisador léxico especificando-se apenas os tokens a ser reconhecidos. Ou seja, o objectivo da análise léxica é reconhecer palavras e neste contexto, um alfabeto são os caracteres do ficheiro fonte, uma linguagem é o conjunto de unidades léxicas e o output é uma sequência de unidades léxicas reconhecidas.

A **análise sintáctica**, também conhecida como análise gramatical é o processo de se determinar se uma lista de unidades léxicas pode ser gerada, ou não por uma determinada gramática. O analisador sintáctico é responsável por verificar se os símbolos contidos no programa fonte formam um programa válido ou não. Os objectivos desta fase são:

- Reconhecer a estrutura do programa
- Ver se o programa respeita a gramática prevista
- Construir a árvore de sintaxe abstracta
- Construir a tabela de símbolos

Nesta fase, o alfabeto são as unidades léxicas, a linguagem é o conjunto das sequências das unidades léxicas que respeitam a gramática, o input a sequência de tokens e o output é a árvore de sintaxe abstracta por preencher com a tabela de símbolos correspondente.

A **análise semântica**, apesar de ocorrer em último lugar, não deixa de ter menos importante do que as duas fases anteriores. Isto porque as análises léxica e sintáctica não estão preocupadas com o significado ou semântica dos programas que processam. O papel do analisador semântico é prover métodos pelos quais as estruturas construídas pelo analisador sintáctico possam ser avaliadas ou executadas. Assim, esta análise não se reduz a um problema de

reconhecimento da linguagem, trata-se é de reconhecer a validade das frases construídas. É nesta fase que se "decora/preenche" a árvore de sintaxe abstracta, que nos permite ter uma representação estruturada e independente da sintaxe dos programas analisados.

Um bom programador, se conhecer o seu compilador alvo, está melhor preparado para evitar erros de código, de forma a que possa evoluir mais facilmente na construção do seu programa. Quando "ajudamos" o compilador a fazer o seu trabalho, podemos ter a certeza que o código gerado será sempre o mais optimizado. E por isto mesmo, a não ser que o leitor enverede pela engenharia da compilação, todos nós temos o nosso trabalho facilitado quando implementamos um programa. Porque temos a certeza que já houve um outro programador que pensou numa forma de reconhecer os erros e de nos ajudar, dizendo-nos qual o erro de compilação que estamos a cometer, ou se temos código que não estamos a utilizar, os chamados "dead blocks".

Contudo caso queiramos mais e queiramos uma linguagem mais especifica, devemos sempre recordar que a qualquer momento podemos ser nos a escrever a nossa própria linguagem de programação e para isso basta que aceitemos o desafio de escrever o nosso próprio compilador. E escrever um compilador é olhar para trás e ver alguns dos passos que Turing, Church e Kleene trilharam há muitos anos atrás. Apesar de o terem feito de forma diferente, qualquer problema dos nossos dias poderia ser resolvido com o modelo de computação destes génios de outrora. E assim, no fim deste artigo, voltamos ao inicio, "encerrando o ciclo" a que nos propusemos.

Em suma, uma das principais metas das linguagens de programação é permitir que programadores tenham uma maior produtividade, programando acções entendidas mais facilmente do que quando comparado com a linguagem que um computador entende nativamente (código de máquina).

E no fim, todos os compiladores, todas as linguagens rumam para o mesmo objectivo… À nossa vontade, enquanto programadores, de simplificar a vida do utilizador. E de certa forma de ajudarmos a melhorar o mundo. Porque melhorar é simplificar, e simplificar é tornar tudo mais claro. E é termos vontade sempre de melhorar, todos os dias, independentemente de usarmos C, Java ou qualquer outra linguagem.

Porque enquanto bons programadores, voltaremos ao inicio… o "Hello World"… o que apesar de ser num sentido figurado, este pequeno programa que todos nós conhecemos, é a base de todos nós. Porque sendo uma parte de nós… cada programa que escrevemos… é uma nova forma de dizermos… "Olá Mundo".

AUTOR

Escrito por Rita Peres

Natural de Castelo Branco e estudante de Engenharia Informática da Universidade da Beira Interior. Membro do P@P desde Janeiro de 2010 (https://www.facebook.com/rita.aperes)

Falácias da Computação na Nuvem

O conceito de computação como um serviço não é original mas a interpretação moderna desse conceito é muito diferente do que se assistiu na década de 70 com os sistemas de time-sharing. Nas suas diferentes formas, a nuvem tem hoje em dia o propósito de: democratizar o acesso a recursos computacionais (Infrastructure as a Service, IaaS); concentrar as equipas de TI no desenvolvimento de aplicações minimizando tarefas de manutenção e administração de infraestrutura (Platform as a Service, PaaS); e disponibilizar soluções Web com o mesmo tipo de funcionalidades que seriam expectáveis numa aplicação desktop (Software as a Service, SaaS).

As várias valências da nuvem são utilizadas por empresas das mais variadas áreas de negócio que exploram algumas das características únicas que este novo modelo veio proporcionar. Duas das características que servem de bandeira ao serviço são a elasticidade e o pagamento dos recursos mediante a utilização.

Elasticidade

A elasticidade refere-se à capacidade dos recursos se dimensionarem automaticamente mediante a carga. Um serviço elástico é aquele em que os recursos disponibilizados correspondem aos recursos exigidos ao longo do tempo. Um serviço não elástico é, por sua vez, um serviço em que os recursos não se adaptam consoante a variação da procura.

Serviço não-elástico

Num serviço não elástico os recursos disponibilizados são fixos, podendo, mediante o dimensionamento feito, estar acima ou abaixo dos recursos exigidos.

A Figura 1 mostra a procura de um serviço ao longo do tempo. O serviço apresenta o seu pico de utilização à tarde, sendo que a linha horizontal corresponde aos recursos disponibilizados/provisionados, a linha preta corresponde aos recursos exigidos no momento e a área quadriculada corresponde ao excesso entre os recursos disponibilizados e os recursos exigidos.

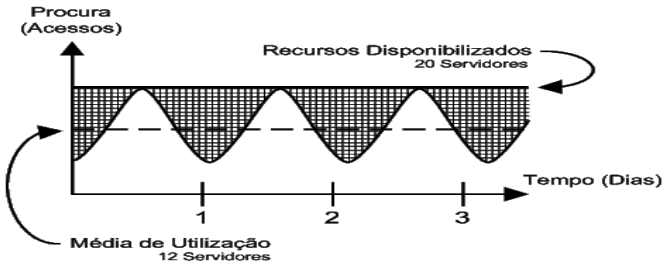


Figura 1 - Serviço não-elástico

Embora o dimensionamento de 20 servidores esteja correcto para o período de pico, estes mesmos 20 servidores são excessivos no resto do dia. A empresa é penalizada financeiramente pois tem custos como electricidade, ar condicionado e manutenção associados a equipamentos subaproveitados.

Se, pelo contrário, o número de servidores for definido por baixo, o problema é ainda maior uma vez que os clientes podem abandonar a aplicação como reacção a um aumento do tempo de resposta ou ao descarte de pedidos.

Serviço elástico

Num serviço elástico os recursos disponibilizados correspondem aos recursos exigidos no momento, ou seja, a empresa nunca paga pelo excesso ou pela escassez de recursos.

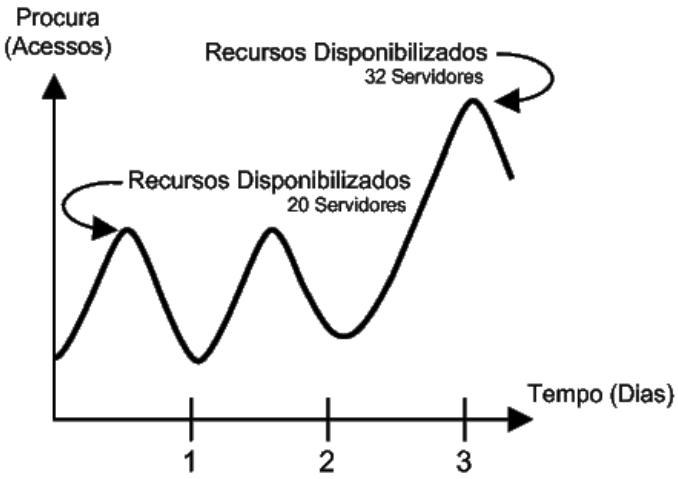


Figura 2 - Serviço Elástico

Assim, a elasticidade que caracteriza a nuvem é aliciante para serviços com picos de tráfego sazonal ou pontual e para serviços para os quais é impossível prever a procura.

Na Figura 2 é mostrado que o número de servidores é flexível e se adapta à carga exigida em determinado instante.

Pagamento dos recursos mediante a utilização

Uma das premissas da nuvem é a de que os recursos sejam pagos mediante a sua utilização num modelo que se assemelha ao encontrado na rede de electricidade, água ou telefone. Por exemplo, se o recurso for processamento o utilizador paga um valor que resulta da multiplicação de uma tarifa pelo número de horas utilizadas mensalmente. Essa tarifa, que indica o preço por hora, varia de acordo com o poder computacional da instância escolhida.

FALÁCIAS DA COMPUTAÇÃO NA NUVEM

Fazendo uma analogia com a rede eléctrica, o utilizador paga um valor que resulta da multiplicação de uma tarifa pelo número de horas utilizadas mensalmente. Essa tarifa, que indica o valor do kWh, varia de acordo com a potência eléctrica contratada.

Como se pode observar, o modelo empregue na nuvem é o mesmo aplicado aos serviços públicos. O modelo aplicado é justo porque o utilizador paga somente o que consome. Um acréscimo no número de horas de computação mensais significa que houve um acréscimo na procura ao serviço ou seja, um eventual aumento do valor pago mensalmente é compensado por um acréscimo no número de potenciais clientes e na receita gerada pela aplicação.

Tendo em conta as características da nuvem existem autores que sugerem a seguinte definição de *cloud-computing*:

"De uma forma geral podemos definir *cloud-computing* como uma infraestrutura altamente escalável desenhada para hospedar aplicações e paga mediante a utilização"

Foram efectuados testes práticos que confrontam a definição apresentada com a versão 1.5 da plataforma de *cloud-computing* da Microsoft, o Windows Azure. A análise crítica resultante destes testes serve de mote ao artigo "Falácias da Computação na Nuvem", o qual está dividido em dois pontos que correspondem à elasticidade e ao pagamento mediante a utilização.

Característica I - Elasticidade

É verdade que a nuvem coloca ao dispor do utilizador quer escalabilidade horizontal quer escalabilidade vertical mas, o facto de a aplicação ser migrada para a nuvem não garante por si só que esta seja elástica. Os benefícios da elasticidade só são realmente alcançados através de um desenho cuidado da aplicação que alcance a independência entre os componentes da mesma, que evite tarefas bloqueantes e preveja mecanismos de prevenção e resposta à saturação do acesso a dados.

A adequação do número de instâncias face a picos de utilização necessita de intervenção manual e por isso a nuvem deve ser vista como um potenciador da elasticidade e não como uma solução.

```
<?XML VERSION="1.0" ENCODING="UTF-8"?>
<!-- FICHEIRO SERVICECONFIGURATION.CSCFG -->
<SERVICECONFIGURATION>
  <ROLE NAME="CLOUDSHOP">
   <!-- ALTERAÇÃO DO NUM DE INSTÂNCIAS DA
APLICAÇÃO -->
    <INSTANCES COUNT="10" />
```

Listagem 1 - Configuração do número de instâncias

No Windows Azure, quando um servidor está a chegar aos limites, o utilizador pode lançar um servidor adicional sem interromper o serviço através da alteração do número de instâncias no ServiceConfiguration.cscfg (escalabilidade horizontal) como mostra a Listagem 1.

```
<?XML VERSION="1.0" ENCODING="UTF-8"?>
<!-- FICHEIRO SERVICEDEFINITION.CSDEF -->
<SERVICEDEFINITION>
 <!-- ALTERAÇÃO DO TAMANHO DA INSTÂNCIA DA
APLICAÇÃO -->
  <WEBROLE NAME="CLOUDSHOP" VMSIZE="LARGE">
```

Listagem 2 - Configuração do tamanho da instância

Do mesmo modo, o utilizador pode optar por escolher uma instância de tamanho superior no ficheiro ServiceDefinition.csdef (escalabilidade vertical) se a sua aplicação não estiver preparada para ser distribuída (Listagem 2).

Estas são duas maneiras de aumentar os recursos da aplicação, mas quer uma quer outra requerem intervenção manual por parte do utilizador, o que pode não ser viável. O facto da escalabilidade necessitar de intervenção manual coloca em causa outro benefício vulgarmente associado à nuvem, o da redução do esforço/custo em tarefas de manutenção.

Embora a escalabilidade automática não esteja implementada de raiz no Windows Azure, é passível de ser alcançada. O primeiro passo para a escalabilidade automática é recolher um conjunto de métricas das instâncias que depois de tratadas nos vão dizer se a aplicação está sob carga. Para recolhermos essas métricas usa-se o agente Windows Azure Diagnostics Monitor. Esse agente permite a recolha de informação de diagnóstico, tais como *logs* do IIS ou eventos do Windows.

O agente *MonAgentHost.exe* foi a solução encontrada pela Microsoft para que o processo de diagnóstico de uma aplicação na nuvem fosse mais parecido ao processo de diagnosticar uma aplicação local. Para isso o agente copia os *logs* guardados no armazenamento local das máquinas e centraliza-os numa conta de armazenamento do Windows Azure providenciada pelo utilizador. Neste caso o agente foi utilizado para recolher métricas que indicam entre outras coisas, a saúde do servidor (ocupação de memória e ocorrências de paginação) e a carga exigida ao mesmo.

A execução do comando *typeperf.exe /q* retorna uma listagem das métricas que podem ser recolhidas e o sítio http://goo.gl/KLcjY pode ser utilizado em complemento para descrever cada uma das métricas e indicar como interpretá-las.

```
/* Instância do Diagnostic Monitor com a con-
figuração por omissão */
DiagnosticMonitorConfiguration dmc = Diagnos-
ticMonitor.GetDefaultInitialConfiguration();
(...)
/* Define a recolha de uma métrica "\Processor
(_Total)\% Processor Time" */
PerformanceCounterConfiguration pcc = new Per-
formanceCounterConfiguration();
pcc.CounterSpecifier = @"\Processor(_Total)\%
Processor Time";
pcc.SampleRate = System.TimeSpan.FromSeconds
(5);
dmc.PerformanceCounters.DataSources.Add(pcc);
dmc.PerformanceCounters.ScheduledTransferPerio
d = TimeSpan.FromSeconds(15);
```

Listagem 3 - Configuração da recolha de métricas

Para se activar a recolha de métricas é necessário adicionar o trecho de código (Listagem 3) ao método *OnStart* do tipo de instância usado, neste caso *Web Role*. O código cria uma instância da configuração por omissão do Windows Azure Diagnostics Monitor, depois define a métrica a recolher e a frequência de amostragem. É ainda definida a periodicidade do envio dos dados recolhidos para a conta. Ao adicionar o código da Listagem 3, as métricas escolhidas são exportadas para a tabela *WADPerformanceCountersTable* da conta de armazenamento, na cadência definida pela propriedade *ScheduledTransferPeriod*.

O segundo passo para a escalabilidade automática é analisar as métricas armazenadas e desencadear acções consoante essa mesma análise. Para adequar os recursos à carga exigida, o utilizador deve usar a *API* de Gestão do Windows Azure. Contudo, o esforço necessário para criar uma aplicação deste tipo é muito grande, pois requer o desenvolvimento de um conjunto de funcionalidades. As funcionalidades a implementar incluem, entre outras: definição das métricas a recolher; monitorização da conta de armazenamento para onde as métricas são exportadas; agregação e análise dos dados recolhidos; alteração dos recursos disponibilizados à aplicação de acordo com a análise feita ou mediante agendamento.

O utilizador pode optar por utilizar ferramentas já desenvolvidas como o *WASABi (Autoscaling Application Block)* da Microsoft ou ferramentas de terceiros no modelo *SaaS* como o RightScale, Opstera ou o Paraleap AzureWatch. Qualquer uma destas ferramentas necessita de estar em execução para que detecte picos de carga e altere os recursos disponibilizados. Deste modo, não são desencadeadas acções caso a ferramenta esteja indisponível porque, apesar das métricas continuarem a ser exportadas para a conta de armazenamento, não são alvo de nenhuma análise.

Para este teste foi escolhida a ferramenta da Paraleap. O AzureWatch tem o preço de 0,33 USD/hora por instância monitorizada tendo como vantagem ser a própria Paraleap a hospedar a ferramenta. A integração do AzureWatch revelou-se simples e intuitiva podendo o utilizador recorrer às instruções disponíveis na página oficial http://goo.gl/rRQ1X.

Finalizado o processo de integração do AzureWatch, não é necessária mais nenhuma alteração ao código da aplicação monitorizada, uma vez que a definição da recolha de novas métricas está presente na aplicação. A configuração da recolha de uma métrica em antecedência não é obrigatória servindo apenas para verificar a criação da tabela *WADPerformanceCountersTable* na conta de armazenamento.

No exemplo foi configurada a métrica *"\Processor (_Total)\% Processor Time"* que não tem grande relevância se for recolhida isoladamente. Um pico de *CPU* momentâneo não é alarmante mas um histórico que indique uma média de utilização acima dos 80% nos últimos 10 minutos já o é. O AzureWatch possibilita que os valores sejam agregados ao longo do tempo e que novas instâncias sejam lançadas caso se detecte um pico de tráfego. Deste modo, a elasticidade automática é alcançada e os utilizadores da aplicação na nuvem não devem sequer notar que os recursos estiveram perto da exaustão. O administrador da aplicação é por sua vez notificado por email cada vez que o número de instâncias é alterado.

Importa referir que a elasticidade não é instantânea. De facto, uma nova instância do tipo Web requer em média 6 minutos e 32 segundos a estar pronta, enquanto uma instância parada (no estado *Stopped*) requer 2 minutos e 40 segundos a entrar ao serviço. Estes tempos fazem com que seja necessário configurar um valor de *trigger* abaixo do que seria expectável para que as instâncias estejam prontas com antecedência. Por exemplo, lançar uma nova instância da aplicação quando a média de carga do *CPU* nos últimos 5 minutos for superior a 90% pode não ser adequado uma vez que os 10% que faltam para a exaustão podem não ser suficientes para suportar o tempo de arranque de uma instância adicional.

Característica II - Pagamento dos recursos mediante a utilização

A maior parte dos serviços do Windows Azure como é o caso dos *BLOBs*, Tabelas ou *Queues* são pagos

mediante a utilização (quantidade de dados armazenados). O mesmo não acontece contudo, com as instâncias de computação em que o utilizador paga o mesmo quer o servidor virtual esteja parada (no estado *Stopped*) quer esteja perto dos limites de utilização.

Na realidade, as instâncias de computação são pagas por estarem reservadas e não consoante o processamento. Este modelo pode fazer com que os utilizadores fiquem reticentes a tirar partido de cenários que tenham instâncias que não estão a processar.

Um destes cenários são os *static upgrades* em que é utilizado um ambiente de testes para fazer actualizações à aplicação. Neste cenário, as aplicações são actualizadas num ambiente de *staging* no qual, é executado uma bateria de testes. Se a aplicação passar nos testes é colocada em produção. O envio da aplicação do ambiente de testes para o ambiente de produção é feito através da comutação de um *IP* virtual (Figura 3); deste modo, a *pool* de instâncias de *staging* passa a ser a *pool* de produção e vice-versa.

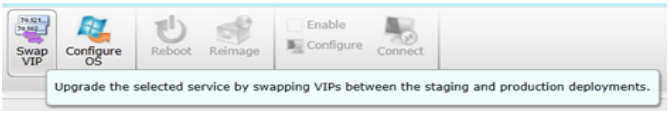


Figura 3 - Comutação do IP virtual no portal de administração

A opção mais sensata a tomar quando se executa um *static upgrade* é a de manter a antiga *pool* de produção como salvaguarda. Se algo de imprevisto ocorrer devido à actualização basta comutar o *IP* para recuperar o serviço. Consoante a política da empresa a salvaguarda pode ser mantida por uma semana ou mais, sendo que neste período o valor pago por uma instância da salvaguarda é igual ao valor pago por uma instância de produção. Parar a instância através do portal de administração também não surte qualquer efeito porque o servidor virtual está reservado na mesma.

Conclusão

Foi verificado que algumas das características associadas com a computação na nuvem não podem ser tomadas como garantidas.

Ao confrontar a definição de *cloud-computing* com a plataforma da Microsoft denota-se que o Windows Azure disponibiliza, de facto, recursos que podem ser solicitados a qualquer momento. Contudo, essa solicitação necessita de intervenção manual ou recorre a código externo que monitoriza as instâncias e que actua em conformidade. A escalabilidade automática é alcançável mas necessita de um esforço adicional que não era expectável. Foi ainda verificado que nem todos os recursos são pagos mediante a utilização. Se alguns recursos são realmente pagos consoante a utilização como é o caso dos *BLOB*s, outros como as instâncias de computação são pagos por estarem reservados, estando ou não a ser utilizados.

As divergências encontradas entre o que é publicitado e o que é realmente oferecido pelo fornecedor de serviço devem ser tidas em conta, antes que uma instituição opte por introduzir este novo paradigma na sua estratégia de negócio.

Referências

Above the Clouds: A Berkeley View of Cloud Computing - Armbrust, Michael et all, University of California, Berkeley, 2009

AzureWatch - Paraleap Technologies

http://goo.gl/rRQ1X

Develop, Enabling Diagnostics in Windows Azure - Microsoft http://goo.gl/zKRDV

Is Cloud Computing Ready For The Enterprise - Staten, James, Forrester Research, 2008

Manning Azure In Action - Hay, Chris; Prince, Brian H., Manning, 2011

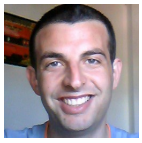
Microsoft Patterns & Practices, Autoscaling and Windows Azure - Microsoft

http://goo.gl/PfNwB

Microsoft Patterns & Practices, Chapter 15 - Measuring .NET Application Performance - Microsoft

http://goo.gl/KLcjY

AUTOR



Escrito por Edgar Santos

Encontra-se actualmente a concluir o Mestrado em Engenharia Informática - Computação Móvel do Instituto Politécnico de Leiria, sob orientação dos professores Patrício Domingues e Sílvio Mendes, no âmbito do qual este artigo está inserido. Profissionalmente, o seu interesse pelas redes de computadores levou-o a ingressar numa multinacional do sector das teleco-municações.
Prefere linguagens interpretadas e a sua linguagem de eleição é o C#.

As reais ameaças de segurança não são os APT

As ameaças na web evoluem rapidamente e por vezes é difícil acompanhar o ritmo dos utilizadores maliciosos, bastante motivados por desafios, dinheiro e poder.

Alguns países estão a criar grupos/exércitos especializados em desenvolver *malware* especificamente criado para roubo de informação confidencial, principalmente na área da espionagem. Estes grupos tentam encontrar novas falhas e novas maneiras de penetrar nos sistemas que muitas pessoas pensavam ser invulneráveis.

Com esta evolução surgiu o termo APT (*Advanced Persistent Threats*), termo esse que serve também, para distinguir ataques comuns dos ataques mais sofisticados.

Temos visto ultimamente nas notícias da especialidade que os APT são as grandes ameaças da nova geração. Para quem não sabe o que são os APT, são ataques persistentes e cíclicos a um alvo especifico, envolvendo técnicas avançadas e levadas a cabo por utilizadores bastante capazes e experientes. O termo surgiu no início do ano e veio referir ataques *zero day*, especialmente as vulnerabilidades nos PDFs e nos documentos que tentavam permanecer indetectáveis enquanto se propagavam em redes governamentais ou corporativas, na maioria dos casos utilizando *spear phishing*.

Tomando a liberdade de pensar que todos sabem o que é o *phishing*, o *spear phishing* é algo mais direccionado a um alvo. O objectivo é, a partir desse alvo, entrar numa rede corporativa de uma forma mais fácil do que, por exemplo, encontrar uma falha no sistema informático da empresa. Utilizando técnicas de engenharia social* e angariando informações nos resultados dos motores de busca e nas redes sociais, o utilizador malicioso tem a possibilidade de construir informação credível permitindo desta forma ganhar confiança e alguma credibilidade no alvo. Um email bem construído (Figura 1), com linguagem preparada e com um link ou anexo malicioso (*FUD - Fully Undetectable*), é uma porta de acesso ao objectivo final do atacante.

Caro(a) Cliente ,

Em anexo enviamos-lhe a sua Factura Electrónica **edp**5D nº00072362356 emitida em 2012/03/24 relativa ao local de consumo C22349082.

A sua factura está também disponível para consulta na sua área de cliente **edp**Online, acessível em www.edp.pt. Poderá ainda consultar o seu histórico de consumos, comunicar leituras e gerir alterações contratuais.

Disponibilizamo-nos para os esclarecimentos adicionais que considerar necessários através do nº 808 53 53 53, todos os dias úteis das 8h às 20h. Poderá ainda contactar-nos por e-mail em edpcliente@edp5d.pt ou visitar-nos no site www.edp5d.pt.

Com os melhores cumprimentos,

Serviço ao Cliente

Figura 1: Exemplo de phishing utilizando a EDP como isco.

Algumas técnicas que tenho conhecimento são bastante simples de executar.

O utilizador malicioso compromete um servidor web, ou apenas uma conta deste, e aloja um iframe em algum dos sites. Envia para a vítima esse site e este não suspeita de nada. Ao mesmo tempo o malware é carregado e compromete a vítima.

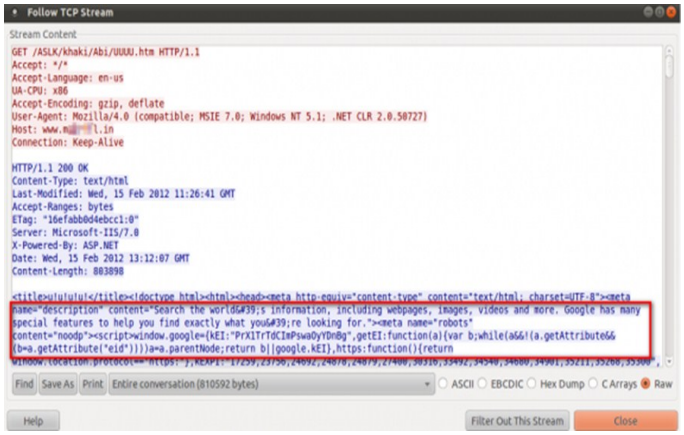


Figura 2: Conteúdo malicioso numa págiona da web.

As falhas mais exploradas recentemente são vulnerabilidades de execução remota no Adobe Flash Player (*CVE-2012-0779* e *CVE-2012-1535*), Internet Explorer (*CVE-2012-1875*) e Microsoft XML Core Services (*CVE-2012-1889*).

Existem algumas regras básicas, mas importantes, para prevenir ataques de *spear phishing* (e outros ataques) das quais destaco:

Apenas disponibilizar na web aquilo que está disposto a perder.

- Não clicar em links suspeitos. Se tem dúvidas, questione alguém que tenha esse conhecimento.
- Manter o software actualizado (sistema operativo, antivirus, browser, etc.).
- Consultar regularmente as últimas notícias sobre segurança informática.

Presentemente, também utilizam o termo APT para falhas não divulgadas aos *vendors* e geralmente são ataques ligados a governos ou a espionagem industrial.

Pessoalmente penso que se trata de apenas um nome *pomposo* para divulgar aos média o que provavelmente sempre existiu.

AS REAIS AMEAÇAS DE SEGURANÇA NÃO SÃO OS APT

De facto, muitos especialistas de segurança concordam que os últimos APT têm pouco de A (*Advanced*) e mais de P (*Persistent*).

Também existem casos, já públicos, em que uma grande empresa ou governo sofre um ataque informático e desconhece a razão (ou não quer *má* reputação por parte dos media) e refere que sofreu um APT. Depois pode-se dar o caso de descobrir que afinal o APT foi um ataque via plugin desactualizado de algum CMS (*Content Management System*). Depois foi só escalar os privilégios. Os utilizadores maliciosos têm acesso à conta do utilizador e depois com um *exploit* local aumentam os privilégios para acesso de administrador.

Claro que com este exemplo não estou a mencionar que os APT não existem ou devem ser desvalorizados; apenas acho que devemos ter mais em conta outras prioridades.

Num exemplo em que uma empresa é afectada por um APT, é importante que a equipa de segurança de resposta rápida seja capaz de procurar em zonas que estão aparentemente *diferentes*. Tudo o que possa parecer estranho, deve ser considerado suspeito. Como este tipo de ataques são persistentes, por vezes é possível criar uma assinatura para cada tipo de actividade.

> ***"Afinal a grande ameaça somos nós."***

É irónico, mas uma equipa de segurança de resposta rápida pode melhorar a capacidade de detectar APTs sendo vítima de um ataque APT. Rob Lee, da Sans, afirmou há uns tempos que - *É necessário primeiro ser vítima de APT para que desta forma ajude a não ser vítima novamente no futuro*.

É importante ter em conta que:

- Ataques de phishing são dos mais lucrativos para os criminosos online.
- Engenharia social é aplicado diariamente para roubar informação confidencial.
- Falhas de programação web, simples de corrigir (RFI, XSS, SQL), ainda são responsáveis pela maior parte dos ataques aos sites actuais.
- Malware, muitas vezes já detectado pelos antivirus, é responsável por infectar milhares de utilizadores diariamente.

Prevenção, educação e formação. Estes são os principais tópicos a focar na segurança.

Os ataques informáticos poderiam ser reduzidos drasticamente se os utilizadores (desde o comum utilizador até ao administrador dos sistemas informáticos) acompanhassem as últimas actualizações e tivessem acompanhamento ou formação por parte de responsáveis de segurança de informação.

Aplicar certas regras de conduta, como por exemplo da SANS – http://www.sans.org/critical-security-controls/ – pode prevenir ou dificultar os ataques comuns e até os famosos APT.

Afinal a grande ameaça somos nós.

* **Engenharia Social**

De forma sucinta, é uma técnica/arte/ciência, que consiste em manipular os indivíduos a tomar certas decisões numa determinada altura das suas vidas. A engenharia social não é um termo recente, é utilizado por exemplo, pelas entidades policiais para obter informações sobre os criminosos, pelas entidades políticas quando transmitem medidas rigorosas para a população (sabemos bem como funciona em Portugal...), pelos advogados quando interrogam os arguidos, e até mesmo pelas crianças quando são pequenas na manipulação da decisão dos seus progenitores... Esta técnica, bem utilizada, pode ter um impacto bastante elevado.

AUTOR

Escrito por David Sopas

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

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

ENTREVISTA A JOÃO BARRETO

João Barreto, um dos fundadores da SysValue e apaixonado pela Segurança da Informação desde os 19 anos de idade que também foi um dos principais responsáveis pelo aparecimento da AP2SI (Associação Portuguesa para a Promoção da Segurança da Informação), fala-nos da falsa segurança em que as nossas informações são partilhadas e até que ponto cada um de nós está consciente de todas as implicações na segurança da informação. Partilha também a sua opinião sobre grupos como "anonymous" e "lulzsec".

Revista PROGRAMAR (RP): Fale-me um pouco de si e como decidiu seguir a área da segurança de informação.

João Barreto (JB): O gosto pela segurança da informação surgiu quando ainda esta mal estava definida, tal designação não fazia parte do léxico diário – muito menos em Portugal - numa altura em que com um pouco de iniciativa e curiosidade conseguia-se investigar e descobrir os segredos que faziam os computadores funcionarem. Estes não eram, nem de perto nem de longe, protegidos (no concreto ou potencialmente) como hoje em dia. Não o eram tecnicamente, do ponto de vista de controlos disponíveis, nem o eram administrativamente pois administradores de sistemas e staff associado não estavam sensibilizados para o efeito. Tinha 19 anos, estudava Informática na Faculdade de Ciências da Universidade de Lisboa e a rede universitária era o ambiente natural e ideal para jovens curiosos testarem os limites do perímetro com que outros condicionavam a sua intervenção. Esta curiosidade sobre como tudo funcionava era alimentada por escritores como William Gibson, numa vertente romanceada e idealista, e magazines como a 2600 que proporcionavam uma visão técnica da coisa.

O meu primeiro emprego foi nos laboratórios de investigação da HP em Bristol, UK, a investigar e desenvolver técnicas de elevar a tolerência a faltas de sistemas e o bichinho ficou. A partir dessa altura andei sempre mais ou menos próximo do tema (inteligência artificial, desenvolvimento de sistemas de testes automáticos, redes de comunicações) até que participei na criação de uma empresa dedicada ao tema que ainda hoje persiste, a SysValue.

Presentemente, sou um dos responsáveis desta empresa onde tenta-se desenvolver e manter valências end-to-end no domínio da segurança da informação. As áreas técnicas onde tenho maior envolvimento são auditorias de processos (testes de intrusão e avaliações técnicas são responsabilidade de outra área na SysValue), gestão de segurança da informação, continuidade de negócio e recuperação de desastres, sensibilização à segurança e, por improvável que soe, o desenvolvimento de sistemas (seguros, pois claro) pois programar é uma disciplina que gosto particularmente.

Paralelamente, sou ainda docente convidado da Faculdade de Engenharia da Universidade Católica onde lecciono módulos na Pós-graduação em Segurança em Sistemas da Informação.

> **"...acredito que muitos não acreditam ainda o quão simples é serem prejudicados (...) por não terem cuidado com a utilização que fazem dos seus sistemas e, principalmente, com a sua informação."**

RP: Diga-nos de forma breve quem é a AP2SI, qual a sua missão e objectivos?

JB: A AP2SI é uma associação sem fins lucrativos que tem como missão a promoção da segurança da informação junto de todas as audiências que identificarmos como necessitadas

de tal. É uma missão que se escreve numa linha mas que implica um esforço hercúleo, permanente e complexo. Tal será atingido, na nossa percepção, pelo desenvolvimento de um conjunto vasto de iniciativas que vão desde a sensibilização do cidadão comum sobre as ameaças a que está sujeito no seu dia-a-dia (quando navega na Internet, quando transporta dados pessoais no seu telefone ou portátil, etc.) até ao reforço de competências de profissionais a trabalhar na área da informática para que, sistematicamente e com qualidade, produzam sistemas e soluções intrinsecamente seguros, não dependentes de remendos e *quick fixes*, resilientes e dotados dos demais atributos que associamos à segurança da informação.

A AP2SI pretende crescer, naturalmente, atraindo para as suas fileiras todo e qualquer indivíduo com skills e competências alinhados com o seu ADN e propósito – independência, rigor e qualidade. Todas estas valências são necessárias para que possamos produzir materiais de sensibilização (na forma de revistas, newsletters, sites, etc.), organizar eventos, orientar profissionais na procura de competências qualificadas, apoiar esforços de regulação profissional que eventualmente surjam e, porque não, participar em esforços governamentais na área.

A AP2SI é uma iniciativa ainda recente, promovida por 13 indivíduos mas que é, e pretende sê-lo cada vez mais, virada para a comunidade e nunca para si própria. Consequentemente, procuramos outros que se queiram associar e participar nas várias actividades que estamos a desenvolver.

RP: Considera que em Portugal o publico está sensível para as questões de segurança na Sociedade da Informação ?

JB: Penso que está hoje mais sensível que há, por exemplo, dois anos atrás. Porém, acredito que muitos não acreditam ainda o quão simples é serem prejudicados pessoal ou profissionalmente, em termos financeiros ou reputacionais, por não terem cuidado com a utilização que fazem dos seus sistemas e, principalmente, com a sua informação. A comunicação social tem tido a sua quota-parte desta melhoria ao produzir peças que remetem para incidentes como os gerados por grupos de hacktivistas como os Anonymous ou os Lulzsec ou, num cenário mais patético, miúdos que acedem e partilham fotografias íntimas de figuras públicas. A comunidade sabe que situações destas são a ponta do icebergue mas o público ainda não tem tal percepção.

RP: Numa altura em que se vê cada vez mais noticias na comunicação social, ligadas a grupos como "anonymous" e "lulzsec", o que nos pode dizer sobre ambos os grupos ?

JB: Tudo o que possa ser dito sobre os grupos que menciona, e todos os demais da mesma natureza, são opiniões baseadas no efeito das suas acções e nas suas declarações dado que pouco se sabe sobre a constituição destes grupos e da sua verdadeira agenda. Pessoalmente, considero que na globalidade tratam-se de jovens idealistas que, apoiando-se em skills técnicos, tentam atrair holofotes para situações ou contextos onde consideram estarem a ser realizados abusos. Nem sequer discutindo o facto de, na vasta maioria dos casos e países, estarem a realizar acções condenáveis à luz da lei, um problema de base é que a noção de abuso é muito relativa pois poderá ir desde a existência de um regime absolutista até à privação de liberdade de alguém como Julian Assange. Se a opinião pública poderia até descomprometê-los em certas situações, noutras nem por isso. O meu maior receio é que estes grupos tenham efectivamente uma agenda e objectivos que não são do conhecimento dos seus operacionais ou, pior, que sejam distintos dos que estes operacionais pensam que são. Eventualmente, serão o bode expiatório de crimes económicos ou, porque não, geopolíticos devido à sua ingenuidade e vontade em acreditar num algo que imaginam maior que eles.

> **“Se a curiosidade, a procura pelo conhecimento, a experimentação e a descoberta forem realizados com cautela, em ambientes controlados, sem dolo, sem prejuízo de terceiros e sem ir contra a lei seremos hackers sem nunca sermos criminosos.”**

RP: Constantemente vemos o termo "hacker" conotado com "crime", o que pensa sobre isto ?

JB: É uma pena pois o termo “hacker” tem, historicamente, um significado que não poderia estar mais longe disso. Porém, como a percepção tem hoje mais impacto que a

verdade, vivemos essa realidade. Dado que combater tal confusão é um esforço desproporcional (além de provavelmente inglório) ao benefício que daí adviria, penso que as pessoas deveriam menos preocupar-se com o que lhe chamam e mais com o que fazem. Se a curiosidade, a procura pelo conhecimento, a experimentação e a descoberta forem realizados com cautela, em ambientes controlados, sem dolo, sem prejuízo de terceiros e sem ir contra a lei seremos hackers sem nunca sermos criminosos.

RP: A sociedade de uma forma "genérica" parece sentir-se segura e sentir que a informação que disponibiliza on-line é segura. Concorda com esta ideia ?

JB: Concordo. É gritante a exposição que muitas pessoas fazem da sua vida e a displicência com que tratam a sua informação. Algures entre a ignorância e a confiança em terceiros, penso que na maioria das vezes os abusos de que as pessoas se sentem alvos deveram-se à sua incapacidade de se protegerem e não a terem sido vítimas de ataques técnicos sofisticadíssimos. Desde colocarem tudo online e não configurarem os parâmetros de privacidade de sistemas como o Facebook até levarem o PC para arranjar a uma loja de centro-comercial com todas as suas fotos íntimas, declarações de IRS digitalizadas e até, porque não, ficheiros com passwords de acesso a sistemas de homebanking, estou certo que acontece de tudo. E como nem toda a gente é séria, na informática como em tudo na vida, os problemas acontecem.

> **“Aos que eventualmente pensem aderir a movimentos (…) que optem antes por colocar os seus conhecimentos e competências a apoiar iniciativas transparentes, públicas, orientadas ao cidadão, sérias.”**

RP: E as redes sociais, têm algum papel importante na segurança da informação ?

JB: Têm-no duplamente. Por um lado, ao não serem os primeiros a promoverem o acesso controlado à informação de uns utilizadores pelos outros pois dependem precisamente dessa interacção e conectividade para prosperarem, contribuem para o estado em que vivemos. Tal tem provocado que controlos e opções de privacidade surjam apenas quando a sociedade civil ou governos mais rigorosos e atentos assim obrigam.

Por outro lado, dada a sua penetração na sociedade (o Facebook, por exemplo, tem 900 milhões de utilizadores registados), deveriam ser um canal privilegiado de promoção e sensibilização da segurança da informação ou, pelo menos, da privacidade. Não o serão, pelo primeiro facto apontado, até que sejam obrigados. E apenas o serão quando todos nós o forçarmos, nomeadamente a sociedade civil que é o seu mercado e alvo principal. O “como” é o que temos que descobrir e, no que a tal respeita, instituições como a AP2SI têm algum trabalho a desenvolver.

RP: Considera importante que se aposte mais na segurança ao nivel do desenvolvimento (produção de código seguro) ?

JB: Não apenas importante mas absolutamente fundamental. Se no passado as infraestruturas eram a fraqueza principal das TIs das organizações, hoje temos efectivamente a situação inversa. Dois factores contribuíram simultaneamente para tal estado de coisas. Por um lado, os fabricantes de soluções infraestruturais produzem produtos muito mais robustos e configurados, out-of-the-box, de forma mais segura. Por outro, o desenvolvimento de código é hoje em dia realizado por muitas mais pessoas, menos sensibilizadas, cujo resultado executa em ambientes que protegem menos – nativamente, entenda-se – o código produzido. 80% das vulnerabilidades presentes em sistemas hoje em dia são aplicacionais. Tal número tem de ser dramaticamente reduzido. O único modo – pelo menos, o mais fácil – é ensinar quem desenvolve código a fazê-lo intrinsecamente mais seguro.

RP: Para terminar, existe alguma mensagem que queira deixar aos nossos leitores ?

JB: Duas. Aos que eventualmente pensem aderir a movimentos como os Anonymous, Lulzsec e afins, que optem antes por colocar os seus conhecimentos e competências a apoiar iniciativas transparentes, públicas, orientadas ao cidadão, sérias. Elas existem. Se não encontrarem nenhuma que lhes agrade, que nos contactem na AP2SI que temos ideias e falta de pessoas. Ao cidadão em geral, que se proteja, protegendo a sua informação. Que se informe sobre como a sua privacidade é garantida em sistemas que utilize e que armazene e transporte a sua informação de forma segura e controlada, não a partilhando facilmente.

PROJECTO EM DESTAQUE NA COMUNIDADE P@P: NOTÍCIASPT

O projecto ao qual escolhemos dar destaque nesta edição tem como criador o Sérgio Ribeiro (ribeiro55).

A NotíciasPT é uma aplicação destinada a clientes Windows Phone, que tem como principal objectivo tornar-se a ferramenta indispensável para quem necessita tirar o melhor partido do tempo e manter-se sempre actualizado.

A aplicação concentra em si as principais fontes de notícias portuguesas, através de RSS, utilizando como fonte o site http://www.jornaiserevistas.com. Além de dar acesso às publicações nele disponíveis, apresenta também as respectivas capas, o que possibilita uma selecção mais célere para os leitores mais apressados. Existe ainda a possibilidade de serem adicionadas mais fontes, a pedido.

Entre as diversas funcionalidades da aplicação, podemos realçar a apresentação dos títulos em destaque, a apresentação das últimas notícias, a possibilidade de adicionar comentários à notícia, entre outras. Uma das funcionalidades que enriquece a aplicação é a existência de uma Shoutbox, que permite a partilha de opinião com outros leitores, acerca dos temas desejados.

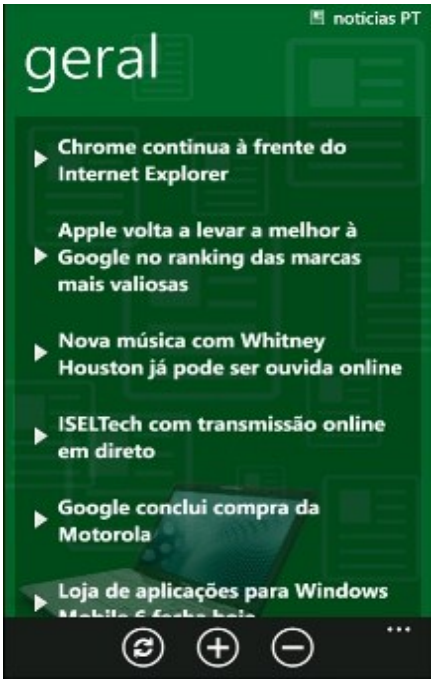
De acordo com o criador da NotíciasPT, a versão 2000 contará com a possibilidade de adição de fontes remotas.

De momento, a versão 2.2 (1800) é a última disponível, ocupando apenas 7MB de espaço de armazenamento. Para que possa funcionar plenamente, tem como requisitos mínimos, a instalação do Windows Phone 7.5 ou superior.

A aplicação foi bem recebida pela comunidade Portugal-a-Programar, tendo sido alvo de inúmeras críticas positivas, bem como vários downloads.

Esta aplicação é gratuita e pode ser descarregada directamente no site oficial do Mercado Windows Phone, ou através do endereço abaixo.

http://bit.ly/QK7jW4



JÁ DISPONÍVEL A APLICAÇÃO DA REVISTA PROGRAMAR, PARA WINDOWS 8

A revista PROGRAMAR já está disponível no Windows 8, com todas as edições e detalhes sobre cada edição.

A aplicação tem todas as edições já publicadas da revista desde o número 1 em 2006. Com esta aplicação pode navegar entre as edições e ler todos os artigos já publicados e pesquisar por artigos através de palavras-chave. Foi desenvolvida por um membro desta comunidade e líder da comunidade NetPonto - Caio Proiete (obrigado pelo seu contributo!).

Algumas imagens da aplicação:

Veja também as edições anteriores da Revista PROGRAMAR

36 Edição - Agosto 2012

354 Edição - Junho 2012

34 Edição - Abril 2012

33ª Edição - Fevereiro 2012

32ª Edição - Dezembro 2011

31ª Edição - Outubro 2011

e muito mais em ...

www.revista-programar.info

DUVIDAS?

IDEIAS?

AJUDAS?

PROJECTOS?

portugal-a-programar.org