PROGRAMAR

A REVISTA PORTUGUESA DE PROGRAMAÇÃO

Revista nº19 - Abril de 2009

www.portugal-a-programar.org

Google WEB TOOLKIT

Desenvolvimento de aplicações web com AJAX mais simples e rápido

AJAX
Asynchronous Javascript And XML

LINUX GRAPHICS STACK

História da evolução do sistema de gráficos de Linux, desde o X até aos mais recentes, DRI2 e Gallium3D.

Processamento de Texto AWK

Manipulação e processamento de dados baseados em texto.

e ainda...

INTRODUÇÃO PRÁTICA À POWERSHELL

Breve iniciação à Windows Powershell e alguns scripts úteis

índice

equipa PROGRAMAR

contacto
revistaprogramar
@portugal-a-programar.org

website
www.revista-programar.info

issn
1647-0710

Novos Horizontes

Esta é a nossa primeira edição como coordenadores, depois da saída do Miguel Pais anunciada na última revista. Parece-nos estranho, mas o diferente cargo que agora ocupamos não tornou o ciclo de preparação desta edição diferente de qualquer outro em que já estivemos envolvidos. Mas, no final de contas, é bastante natural que não sintamos grande diferença - coordenar, articular, preparar cada edição da revista apenas a partir da vossa colaboração voluntária foi o que sempre fizemos, e é o que pretendemos continuar a fazer agora, da mesma forma, embora agora com uma diferente responsabilidade.

Por outro lado, é claro que o novo cargo não é exactamente igual àqueles que desempenhámos durante todo este tempo. Acresce-nos agora a responsabilidade de darmos a cara pelo projecto, de conseguirmos mantê-lo sem o suporte do Miguel Pais da maneira a que nos habituámos e é nessa frente que apostamos para tornar a mudança na equipa o menos brusca possível. Desta forma, e no seguimento de algumas sugestões do antigo coordenador, tencionamos mudar vários aspectos relativos à equipa.

Em primeiro lugar, e uma vez que nenhum de nós tem muita experiência na gestão de uma equipa e de um projecto destas dimensões, vamos procurar expôr todos os assuntos internos da revista a toda a equipa, envolvendo mais os restantes colaboradores na gestão desta. Assim, vamos também recolher opiniões da equipa antes de tomar quaisquer decisões, porque vinte cabeças pensam sempre melhor que duas.

Em segundo lugar, também numa tentativa de procurar suporte adicional à nossa coordenação e de colmatar algumas falhas que nos têm sido apontadas, criámos dois grupos responsáveis, respectivamente, pela revisão linguística dos artigos e pela divulgação da revista. Esperamos que a diferença se faça notar já a partir desta edição, com menos erros linguísticos e um discurso mais apropriado a uma publicação como a nossa, se possível culminando num maior número de downloads.

Finalmente, fruto dos resultados do recente inquérito do Portugal-a-Programar, confirmámos que a mudança da imagem, quer da própria revista, quer da sua presença na web, é uma das nossas maiores prioridades. Este é um dos aspectos em que nos vamos concentrar ao longo das próximas edições, embora os resultados visíveis ainda possam tardar.

Destaque ainda para o novo editor, António Silva, colaborador frequente do projecto há já algum tempo. Resta-nos apenas esperar que a nova equipa consiga manter o seu trabalho ao nível do que tem sido este projecto, cada vez mais uma referência para os programadores em Portugal e além fronteiras.

Joel Ramos e Pedro Abreu

Lançamento do Debian 5.0

http://www.debian.org/News/2009/20090214

Lançamento do Internet Explorer 8

http://www.microsoft.com/windows/Internet-explorer/

Mozilla lança o Bespin

http://labs.mozilla.com/2009/02/introducing-bespin/

Lançamento do GNOME 2.6

http://library.gnome.org/misc/release-notes/2.26/

Lançamento do LLVM 2.5

http://llvm.org/releases/2.5/docs/ReleaseNotes.html

Lançamento do Parrot 1.0

http://www.parrot.org/news/2009/Parrot-1.0.0

Lançamento do Google Code Labs

http://google-codeupdates.blogspot.com/2009/03/introducing-labs-for-google-code.html

Próximos eventos

2ª Edição do Concurso de Projectos de Programação P@P

No âmbito do Concurso de Projectos de Programação P@P, cujo objectivo é a dinamização e divulgação das diversas linguagens de programação, assim como reconhecer os programadores pelos seus trabalhos, está a decorrer a sua segunda edição.

Esta segunda edição decorre na categoria "Software Proprietário", na qual podem participar todos os projectos que não possuam uma licença de software livre. A entrega do código fonte não é obrigatória e não será avaliada.

O calendário desta edição é o seguinte:

Início do Concurso:	01-03-2009
Prazo de Entrega:	17-04-2009
Anúncio de Finalistas e Votação do Público:	24-04-2009
Fim de Votação e Anúncio do Vencedor:	01-05-2009

Para mais informações

http://www.portugal-a-programar.org/forum/index.php/topic,32312.0.html

Olimpíadas Nacionais de Informática 2009

http://www.dcc.fc.up.pt/oni/2009/

Take Off 2009

- 25 de Abril, no Departamento de Engenharia Informática da Universidade de Coimbra

http://takeoff.ideias3.com/2009/

CPAS - Concurso de Programação para Alunos do Secundário

- 29 de Abril, no Colégio Internato dos Carvalhos
- Inscrições até ao dia 24 de Abril

http://e-escola.cic.pt/cpas/

Google Web Toolkit

Introdução

Certamente muitos de vocês já tiveram necessidade de desenvolver algum tipo de aplicação para a Web e, como tal, devem ter investigado sobre plataformas que já dessem algum suporte ao desenvolvimento deste tipo de aplicações. Com uma rápida pesquisa no Google encontrámos um grande número de plataformas Web sobre as mais diversas tecnologias e, a quantidade e diversidade é tal, que se torna difícil saber qual plataforma escolher. Mas, este artigo não irá focar as plataformas Web existentes, mas sim sobre uma plataforma em particular, o Google Web Toolkit. O Google Web Toolkit foi desenvolvida pela Google (tal como o nome sugere), inicialmente com o objectivo de dar suporte aos seus engenheiros no desenvolvimento de aplicações Web. Posteriormente a Google teve a amabilidade e a brilhante ideia de disponibilizar a plataforma para a comunidade em geral...

A vida Pré-GWT... Javascript vs Browser

Javascript, é uma linguagem muito utilizada na Web, pela tecnologia AJAX (Asynchronous JavaScript and XML) criada para a construção de páginas dinâmicas sendo directamente interpretada pelo browser. O nome Javascript chega a ser enganador pois, numa primeira abordagem, pensamos logo ser uma derivação do Java o que não é de todo verdade, pois esta linguagem possui este nome simplesmente por questões de marketing, já que na data em que o nome lhe foi atribuído o Java era linguagem sensação. Um dos problemas do JavaScript é o facto de esta possuir uma relação estreita com os diferentes browsers, na medida em que cada um tem a sua própria interpretação do código, ou seja, não existe um padrão que permita a correcta interpretação da linguagem, em todos os browsers. Trocando por miúdos significa que, ao programar em JavaScript, corre-se risco de as funções criadas não terem comportamento semelhante em todos os browsers. Imagine-se o programador, confortável na sua secretária, a desenvolver uma aplicação Web, fazendo umas coisas engraçadas com JavaScript e, na altura de testar, todo entusiasmado, repara que a aplicação não funciona correctamente em todos os browsers. E porquê??? Simplesmente porque o JavaScript precisa de "ser refinado" de forma a ser bem interpretado por cada um deles... É claro que qualquer bom programador quer que sua aplicação seja executada de uma forma correcta e completa, independentemente do browser, e como tal começa a procurar os potenciais problemas, investigar sobre incompatibilidades e entra no terrível ciclo de procurar, modificar, testar, procurar, modificar, testar....

É realmente muito incómodo andar a programar módulos para que a mesma aplicação corra correctamente em todos, ou pelo menos, na maioria dos browsers. Isso implica que o programador tenha conhecimentos dos problemas de incompatibilidades e tempo para que possa corrigi-los (e tempo no mundo do desenvolvimento de software é preciosíssimo). Foi então que os engenheiros da Google, pessoas experientes no desenvolvimento de aplicações com tecnologia AJAX como o GMail, Google Calendar, Google Maps, cansados de incompatibilidades, correcções e testes, sentiam necessidade de usufruir de uma plataforma que desse suporte à correcção de tais incompatibilidades, e que torna-se o processo de desenvolvimento das aplicações mais produtivo, mais flexível e consequentemente mais rápido.

Como resultado surgiu o Google Web Toolkit.

Google Web Toolkit – A "caixinha mágica"

Esta plataforma revelou-se bastante poderosa na medida em que quem a utiliza pode desenvolver as suas aplicações Web puramente em Java... É verdade, Java. O GWT, como já foi dito, é uma plataforma Web, e como tal consiste num conjunto de API's e componentes visuais que facilitam o processo de desenvolvimento das aplicações, sendo que este é feito utilizando a linguagem Java, uma linguagem madura, robusta, orientada a objectos, utilizada por muitos programadores e cuja documentação na Web é vasta. Para além disso, o GWT permite a escolha de um IDE Java onde será desenvolvida a aplicação. Assim sendo, para utilizar a plataforma basta apenas fazer o download da mesma, escolher um IDE Java e começar a trabalhar, tão simples quanto isto. Mais adiante, no tópico "GWT – Primeiros Passos", este processo inicial será explicado com mais pormenor. Mas o que torna o GWT mesmo interessante é o facto de que, depois de desenvolvida a aplicação, o que o

este faz é compilar todo o código Java para código Javascript, para que possa então ser correctamente interpretado pelo browser, corrigindo automaticamente todas as incompatibilidades.

JAVA[script]?

Java para Javascript dizem vocês... Vamos, como se costuma dizer, "take a look inside the Box". O que acontece é que o GWT possui um compilador Java-to-JavaScript responsável por traduzir todo o vosso código Java para o respectivo código JavaScript tendo em conta factores importantes como optimização de espaço, ou seja, o código JavaScript gerado, é percorrido e analisado de forma a determinar se todo o código é utilizado, pelo que se houver alguma secção que nunca é utilizada, esta é simplesmente descartada. Por outro lado, são apenas incluídas as bibliotecas que são realmente utilizadas.
Espaço e optimização foram factores considerados fundamentais pela equipa que desenvolveu o GWT.

Java-to-JavaScript – Mas...

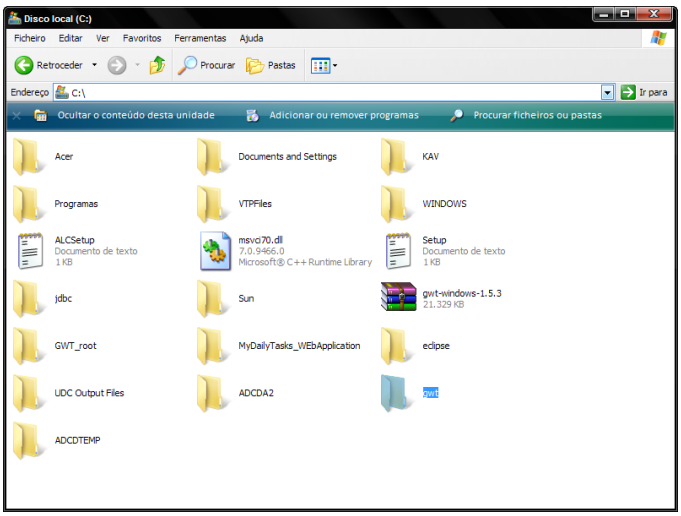
No meio de todo este processo engraçado existe um "Mas", ao qual precisamos de ter muita atenção. É que, como muitos sabem a linguagem Java é mais madura e mais evoluída que a linguagem JavaScript e, como tal, existem classes e bibliotecas da primeira que não são suportadas por esta última. Falando concretamente no GWT 1.5 (última versão disponibilizada para a comunidade) que já é compatível com Java 1.5, e onde por exemplo, genéricos já são suportados. Mas para uma visão geral e completa, o melhor mesmo é consultar a documentação disponibilizada pela Google (http://code.google.com/docreader/#p=google-web-toolkit-doc-1-5&s=google-web-toolkit-doc-1-5&t=RefJreEmulation).

GWT – Primeiros Passos

Agora que já têm noção do que é e para que serve a plataforma GWT vamos meter mãos na massa... De seguida, serão explicados os passos de demonstração de como começar a utilizar o GWT. Irá ser criada a estrutura da aplicação e do projecto para ser importado no Eclipse. Note que neste caso é utilizado o Eclipse mas, como já foi referido anteriormente, as aplicações GWT podem ser desenvolvidas utilizando qualquer IDE Java da sua preferência.

Passo 1 – Descarregando o Google Web Toolkit para o nosso computador

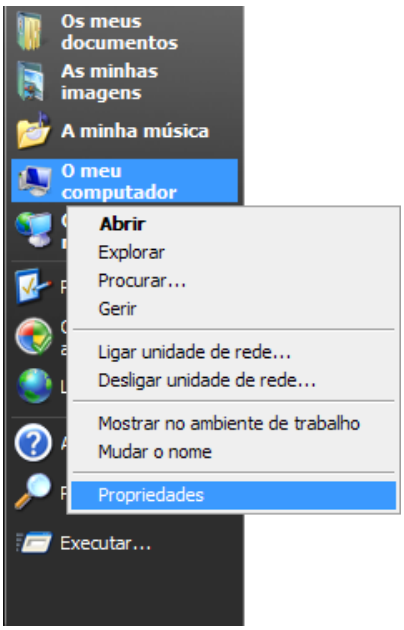
Antes de mais é necessário descarregar o GWT. Para tal, vamos à http://code.google.com/intl/pt-PT/webtoolkit/download.html e fazemos o download da última versão da plataforma (GWT 1.5).
Depois de descarregado o GWT, extraia a pasta, copie-a para a localização C:\ e renomeie-a com o nome "gwt".



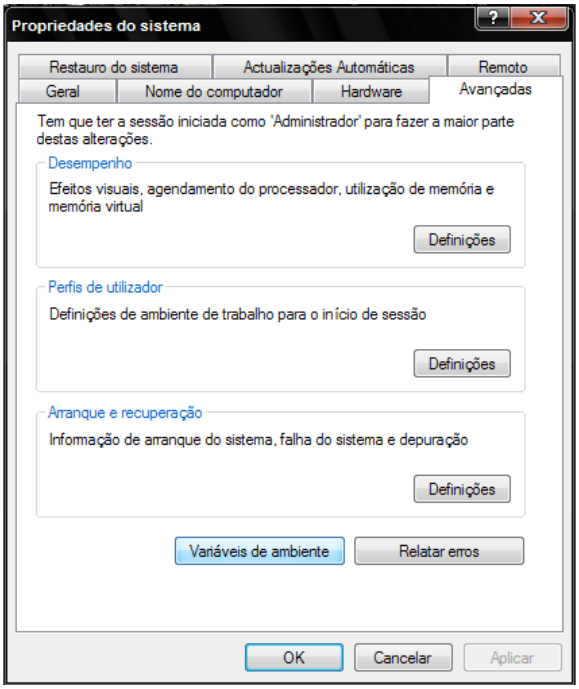
Ao abrir a respectiva pasta podemos reparar em dois aplicativos: o applicationCreator e o projectCreator, que já permitem, através da linha de comandos, criar a estrutura da aplicação e do projecto respectivamente, para que este possa ser então importado no Eclipse.

Passo 2: Variáveis de Ambiente - Adicionar o GWT ao PATH

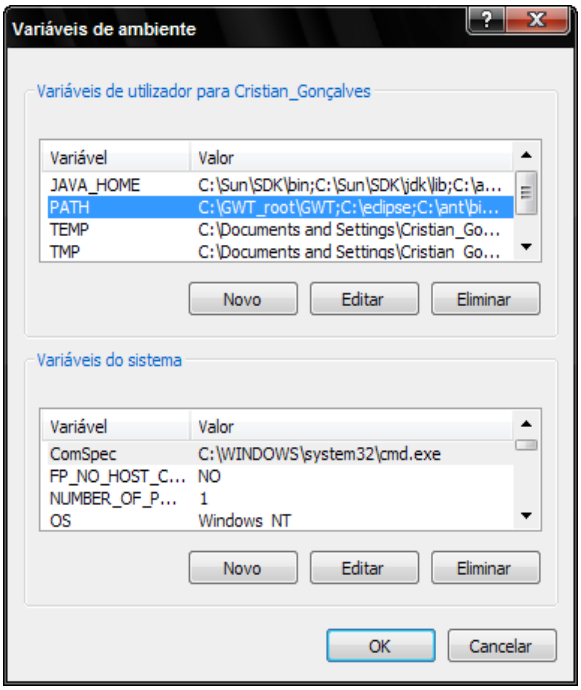
Uma vez que os aplicativos anteriores são invocados por linha de comandos e para evitar colocar todo o endereço sempre que estes são invocados, o melhor a fazer é adicionar a aplicação GWT ao PATH do sistema, ou seja, às Variáveis de Ambiente. Para isso vá até ao menu "Iniciar", clique com o botão direito do rato sobre "O meu computador" e escolha "Propriedades".



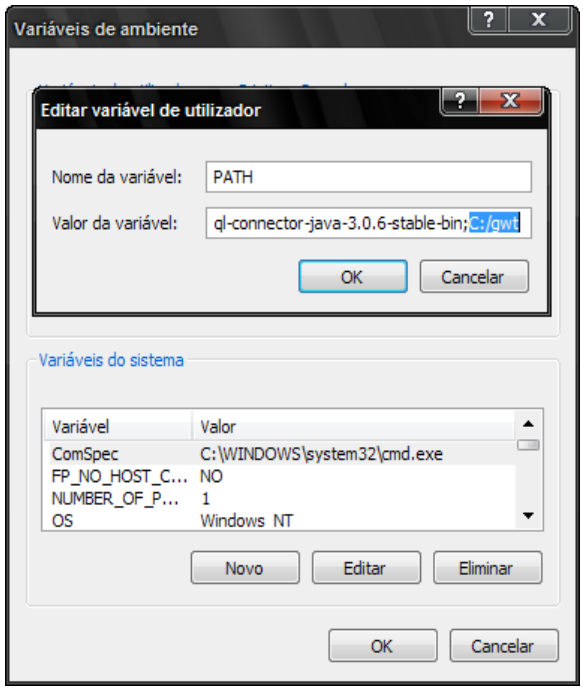
De seguida irá surgir uma nova janela, como mostra a figura seguinte, que dá acesso as Variáveis de Ambiente. Selecione o separador avançadas e posteriormente: "Variáveis de Ambiente".



Finalmente surge a janela das Variáveis de Ambiente. Clique sobre a variável PATH e de seguida clique em "Editar".



Já "dentro" da variável PATH, digite "C:\gwt" (note que tem de separar todos os endereços por ponto e vírgula, tal como mostra a figura).
Depois é só clicar em OK e está feito... O GWT está adicionado a variável PATH.



Passo 3: Gerar a estrutura do Projecto

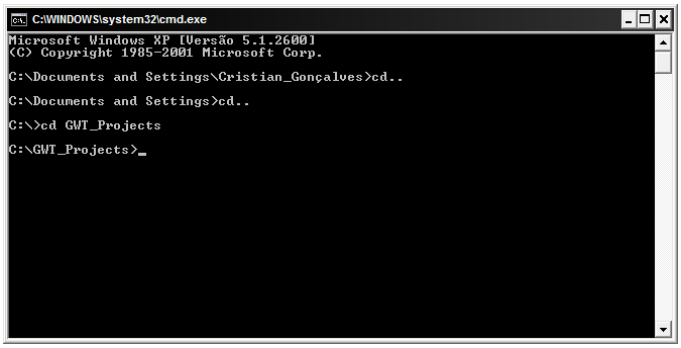
Uma vez que o GWT já foi adicionado ao PATH do sistema é agora mais fácil invocar os aplicativos applicationCreator e projectCreator na linha de comandos. As linhas executadas para os aplicativos têm a seguinte sintaxe, para o applicationCreator:

```
applicationCreator -eclipse
NOMEDOPROJECTO -out LOCALIZAÇÃO
gwt.myfirst.sample.NOMEDOPROJECTO.clien
t.NOMEDOPROJECTO
```

Como podem ver, existem três secções importantes que definem respectivamente, o nome e localização do projecto e (a terceira linha) o local onde os ficheiros java serão guardados (sempre dentro da LOCALIZAÇÃO). Pode especificar esta terceira linha como quiser, no entanto, convém que seja do tipo *.client.* ou *.server.* apelando assim à organização do código (código do lado cliente, código do lado servidor).
Para o projectCreator:

```
projectCreator -eclipse NOMEDOPROJECTO
-out LOCALIZAÇÃO
```

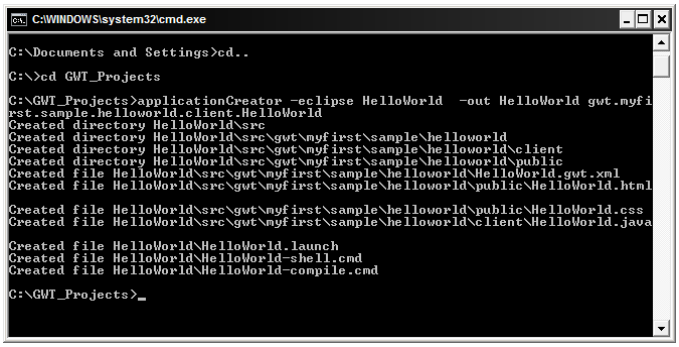
Para gerar a estrutura do Projecto basta especificar o nome do Projecto e a sua localização. Este gera um conjunto de pastas e ficheiros que irão consistir na estrutura do projecto a importar no Eclipse.
Voltando ao trabalho, crie uma nova pasta em "C:\" com o nome "GWT_Projects". Esta será a pasta onde ficará guardada a sua primeira aplicação GWT.
Agora abra a linha de comandos e vá para a pasta "GWT_Projects".



Para criar a estrutura da sua primeira aplicação GWT, designada "HelloWorld" digite a seguinte linha:

```
applicationCreator -eclipse HelloWorld
-out HelloWorld
gwt.myfirst.sample.helloworld.client.Hel
loWorld
```

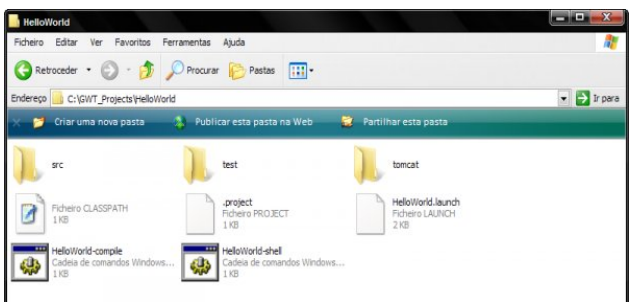
Como pode ser observado, foram criados um conjunto de pastas e ficheiros que constituem a estrutura da aplicação.



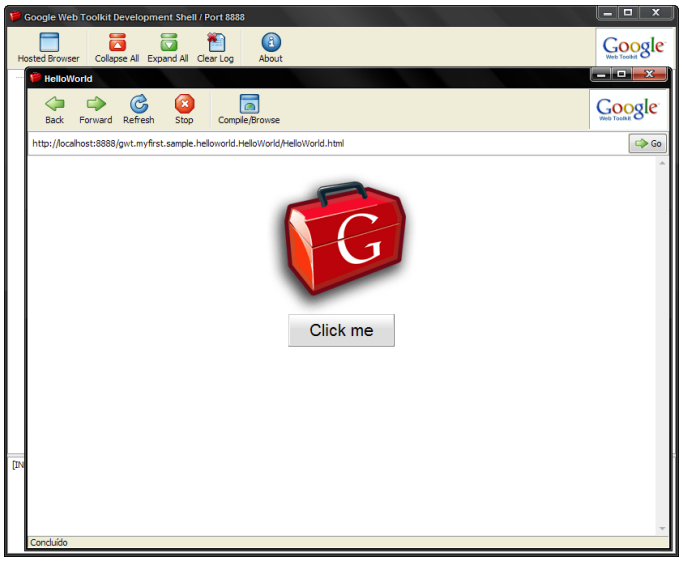
Criada a estrutura da aplicação, agora será criada a estrutura do projecto. Para tal, digite agora a seguinte linha:

```
projectCreator -eclipse HelloWorld -out
HelloWorld
```

Mais uma vez, foram gerados um conjunto de ficheiros e pastas que irão constituir a estrutura do projecto da aplicação.
Agora, dentro da pasta "GWT_Projects" pode-se observar que existe uma nova pasta designada "HelloWorld" que contém todas as pastas e ficheiros gerados nos passos anteriores. Esta pasta consiste na sua nova aplicação GWT.



A partir deste momento já é possível executar a aplicação, embora esta ainda não tenha sido importada para o Eclipse. Pode se observar pela imagem acima que foram criados dois novos aplicativos, o "HelloWorld-compile" e o "HelloWorld-shell". O primeiro permite compilar todo o código Java para código JavaScript para que a aplicação possa então ser colocada e executada na Web enquanto que o segundo permite executar a aplicação em "Hosted-Mode".



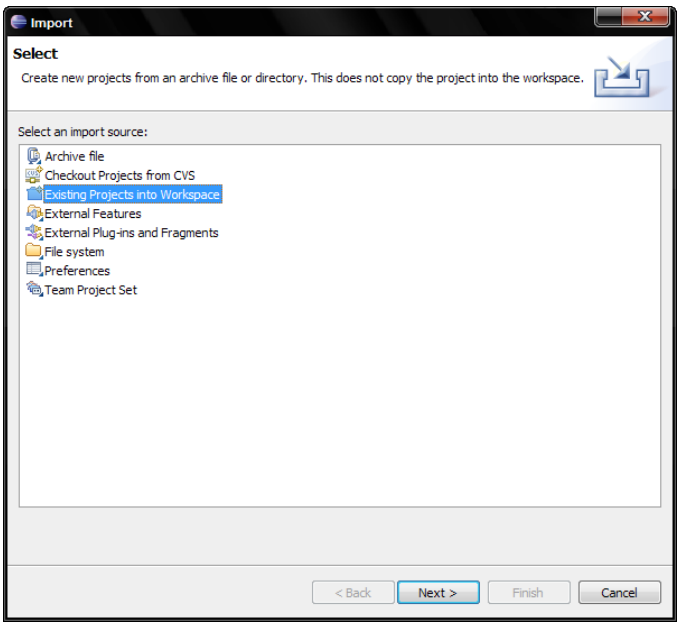
Mas... e o que significa "Hosted-Mode"?

Esta é outra das particularidades do GWT, que permite que as aplicações sobre ele desenvolvidas sejam executadas em dois modos: Web-Mode e Hosted-Mode.
Web-Mode consiste na execução da aplicação sobre a Web, onde todo o código Java já foi convertido para Javascript, ou seja, a aplicação já corre sobre a plataforma para a qual foi criada, a Web.
Hosted-Mode consiste num browser interno do GWT, usado durante a fase de desenvolvimento da aplicação e que permite o debbuging das aplicações em código Java. Ou seja, sempre que for preciso testar a aplicação não é necessário compilar todo o código Java para Javascript e executar a respectiva sobre a Web. Desta forma, toda a aplicação é desenvolvida e testada em Java, facilitando muito a tarefa do programador, ficando-se com a certeza de que quando esta for "migrada" para a Web, estará em pleno funcionamento.

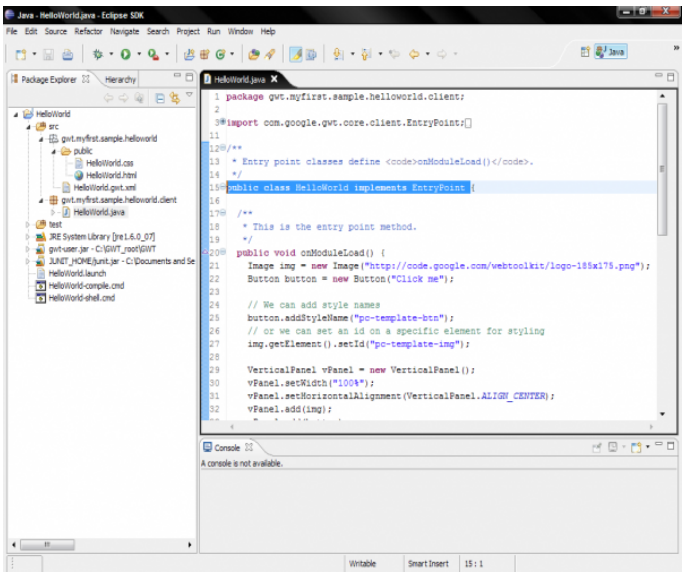
Passo 4: Importar e executar a aplicação no Eclipse

Agora que a estrutura da aplicação está criada, basta importá-la para o Eclipse. Para importar o projecto, siga os seguintes passos:

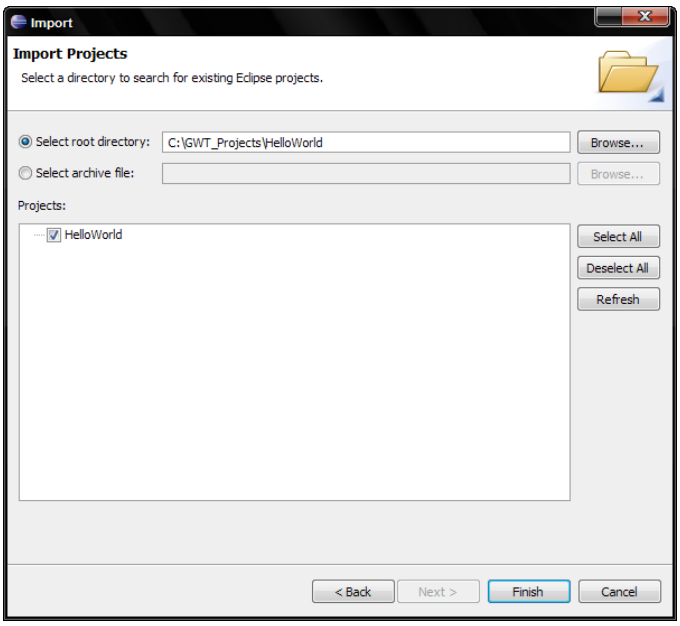
1. Abra o Eclipse e vá ao menu "File" "Import" (Irá surgir uma janela com o item "Existing Projects into Workspace" já seleccionado).



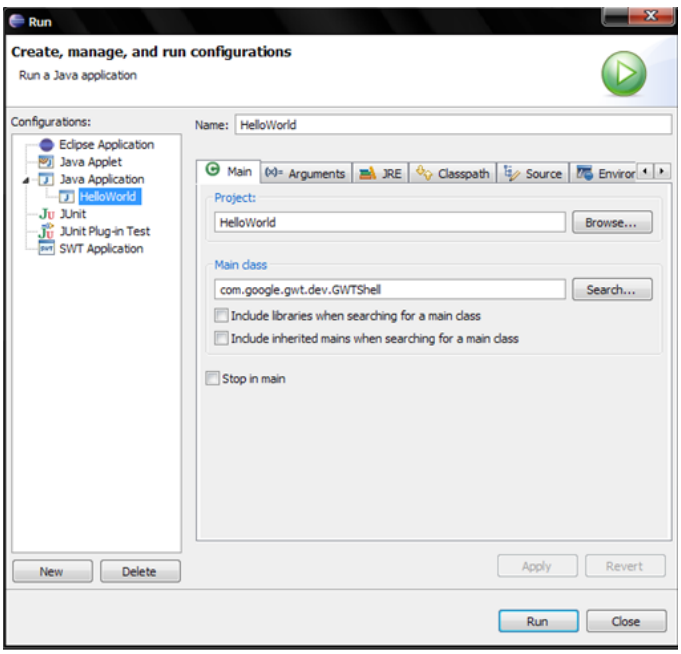
2. Clique "Next".
3. Surge novamente uma janela, onde deverá seleccionar a pasta onde se encontra o projecto que pretende importar. Vá à pasta "GWT_Projects" e seleccione a pasta "HelloWorld".



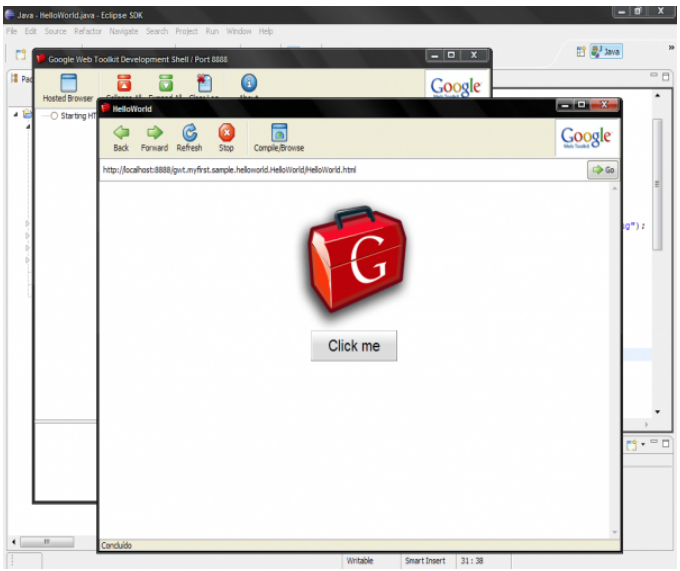
A classe "HelloWorld" implementa a interface designada "EntryPoint" que basicamente consiste no main de uma aplicação Java normal, ou seja, este é o ponto de entrada de execução da aplicação.
Para correr a aplicação em "Hosted-Mode" basta clicar no botão "Run". Irá surgir uma nova janela onde devem seleccionar o item à esquerda correspondente ao projecto, neste caso "HelloWorld".



4. Para concluir o processo clique em "Finish" e pronto, já está, a aplicação "Hello World" foi importada no Eclipse com sucesso.



Cliquem em "Run" e a aplicação será então executada através do Browser interno do GWT.



Esta é a aplicação por defeito. A partir daqui podem começar a desenvolver a vossa aplicação Web. Foi aqui demonstrado o "modo manual" para trabalhar com a plataforma para que pudessem entender um pouco a estrutura e o processo de criação de um projecto GWT. No entanto, já existem plugins como o "GWTDesigner", que infelizmente não é freeware mas na qual é disponibilizado um trial, e que dá suporte ao desenvolvimento de aplicações GWT sobre o Eclipse. O "IntelliJ Idea" (outro IDE Java) já permite também criar projectos GWT dando também muito e bom suporte ao desenvolvimento deste tipo de aplicações.

Concluindo...

O Google Web Toolkit é uma plataforma muito interessante e sem dúvida será cada vez mais requisitada na medida em que esta utiliza uma linguagem muito popular, o Java. Embora alguns dos seus componentes ainda tenham alguns bugs, está constantemente a ser corrigida e aperfeiçoada, tornando-se cada vez melhor a cada versão disponibilizada.
O objectivo deste artigo foi dar a conhecer o Google Web Toolkit e os primeiros passos com esta plataforma Web. Ficaram a conhecer de uma forma geral o que é, para que serve e como funciona.
No próximo artigo irá ser elaborada uma aplicação GWT mais complexa de forma a abordar mais aspectos desta plataforma, como por exemplo, comunicação cliente/servidor, desenho de interfaces, etc... Até lá poderá investigar mais sobre o GWT pois existem já muitos e bons livros que descrevem e ensinam de forma exaustiva todos os seus detalhes.
Até a próxima edição da Revista Programar.

Bibliografia

[DevGuide Java Compatibility]
http://code.google.com/p/google-web-toolkit-doc-1-4/wiki/DevGuideJavaCompatibility
[GWT Brasil]
http://www.gwt.com.br/
[GWT Tutorial - Introduction to GWT]
http://developerlife.com/tutorials/?p=80
[Google Documentation Reader – Google Web Toolkit]
http://code.google.com/docreader/#p=google-web-toolkit-doc-1-5&s=google-web-toolkit-doc-1-5&t=GettingStarted
[GWT Designer]
http://www.instantiations.com/gwtdesigner/
[IntelliJ Idea - GWT]
http://www.jetbrains.com/idea/features/gwt.html

SOBRE O AUTOR

Licenciado em Engenharia de Informática pela Universidade da Madeira, está agora a concluir o Mestrado no respectivo curso na mesma universidade.
Gosta de linguagens como Java e C#, mas no futuro profissional gostaria de enveredar pelo desenvolvimento de aplicações Web, dando foco a um tema ou aspecto que tem ganho cada vez mais relevância no Mundo do Software - a Usabilidade - neste caso concreto, a Usabilidade na Web.

cristian.goncalves@portugal-a-programar.org
Cristian Gonçalves

Processamento de texto em AWK

O que é?

O AWK é uma linguagem de programação criada nos anos 70 com o objectivo de processar dados baseados em texto. Esta linguagem baseia-se fortemente na manipulação de strings e no uso de expressões regulares.

Porquê usar AWK

O AWK tem a vantagem de permitir executar tarefas simples sobre texto utilizando programas mais compactos os que equivalentes escritos em linguagens imperativas como a linguagem C. Isto acontece devido à inexistência de uma função main e de declaração de variáveis.

Estrutura de um programa em AWK

Os programas em AWK são constituídos por uma série de pares condição-acção, com a seguinte sintaxe:
O programa vai ler o input sob a forma de registos

```
condição { acção }
```

(geralmente cada registo corresponde a uma linha) e dividir cada registo em campos (geralmente separados por espaços). Após isto, a condição vai ser testada para cada registo e nos casos em que seja verdadeira é executada a acção.

Exemplo simples: Hello World

Um programa "hello World" em AWK tem o seguinte aspecto:

```
BEGIN { print "Hello World" }
```

Em que print é uma palavra reservada que escreve na saída os seus argumentos e BEGIN é uma condição que indica que a acção correspondente que deve ser executada antes de percorrer o input (da mesma forma, existe um condição END cuja acção é executada no fim do input).

Este programa pode ser executado através de uma linha de comandos da seguinte forma:
Ou, sendo hello.awk um ficheiro contendo o código a executar:

```
awk 'BEGIN { print "Hello World" }'
```

Neste caso não existe input, mas caso exista ele pode ser enviado através do standard input.

```
awk -f hello.awk
```

Campos e variáveis

Como já foi dito, quando um registo é lido a partir do input, ele é dividido em campos (separados por espaços, por omissão). Cada um desses campos pode ser acedido através de uma variável de campo da forma `$N`, em que N é o número do campo (`$1` representa o primeiro campo, `$2` o segundo e assim sucessivamente). Existe o caso particular de `$0` que contém todo o registo.

Por exemplo, a seguinte acção escreve apenas o segundo campo de cada registo:

```
print $2
```

Para além das variáveis de campo é possível definir e manipular variáveis (geralmente para guardar valores intermédios). Estas variáveis não têm associado um tipo e são automaticamente inicializadas com o ou "" (string vazia), dependendo da forma como forem utilizadas. Existem ainda, definidas por omissão as seguintes variáveis especiais:

- NF: Número de campos do registo actual;
- NR: Número do registo actual;
- FS: Separador de campos (por omissão, espaço);
- RS: Separador de registos (por omissão, newline).

Por exemplo, a seguinte acção escreve o último e o primeiro campo de cada registo:

```
print $NF,$1
```

Condições e expressões regulares

As condições podem ser de diversos tipos, como expressões lógicas e aritméticas (envolvendo variáveis de campo ou variáveis especiais). No entanto, o mais frequente é utilizar uma expressão regular como condição.

As expressões regulares permitem descrever de uma forma flexível padrões em strings, sendo úteis para filtrar apenas as linhas que contenham uma determinada palavra. Por exemplo, a seguinte condição define as linhas do input contendo a string programar:

/programar/

Neste caso a expressão regular equivale a uma string, mas podem ser usadas expressões regulares mais complexas:

- /^programar/: Registos que começam com programar;
- /programar$/: Registos que terminam com programar;
- /^programar$/: Registos que sejam iguais a programar;
- /[Pp]rogramar/: Registos que contêm Programar ou programar;
- /p@p|programar/: Registos contendo p@p ou programar;
- /[a-z]/: Registos contendo uma letra minúscula;
- /[^a-z]/: Registos não contendo uma letra minúscula;
- /[a-zA-Z]/: Registos contendo uma letra maiúscula ou minúscula.

Estas condições procuram a expresão regular em todo o registo. Para a procurar apenas num campo utiliza-se o operador ~. Por exemplo, a seguinte condição verifica se o segundo campo do registo contém uma letra maiúscula:

```
$2~[A-Z]
```

É também possível definir duas condições para delimitar um bloco de texto. Por exemplo, a seguinte condição define o bloco entre duas linhas que começam com programar:

```
/^programar/,/^programar/
```

Exemplos de programas em AWK

Em seguida estão alguns exemplos simples de programas escritos em AWK:

- Contagem de palavras:

```
BEGIN { words = 0 }
{
   words += NF # Cada palavra é um campo.
}
END { print words }
```

- Contagem de linhas não-vazias:

```
BEGIN { lines = 0 }
NF>0 { lines++ }
END { print lines }
```

- Soma de uma lista de números (um número por linha):

```
BEGIN { sum = 0 }
{ sum += $1 }
END { print sum }
```

- O mesmo que a anterior, mas somando apenas os números menores que 20:

```
BEGIN { sum = 0 }
$1<20 { sum += $1 }
END { print sum }
```

- Média de uma lista de números:

```
BEGIN { sum = 0 }
{ sum += $1 }
END { print sum/NR }
```

- Números das linhas que contêm a palavra "programar" (case-sensitive):

```
/programar/ { print NR }
```

- O mesmo que a anterior, mas case-insensitive:

```
/[Pp][Rr][Oo][Gg][Rr][Aa][Mm][Aa][Rr]/
{ print NR }
```

- Selecção das linhas ímpares:

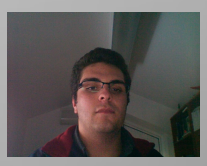
```
NR%2==1 { print }
```

- Selecção apenas das linhas entre "INÍCIO" e "FIM":

```
/^INÍCIO$/,/^FIM$/ { print }
```

Estes são apenas exemplos simples com o objectivo de indicar como as capacidades do AWK. É no entanto possível construir programas mais complexos contendo vários pares condição-acção, utilizando variáveis para guardar parte de registos ou alterando os separadores de campo e registo. Estes tópicos mais avançados não são abrangidos neste artigo e podem ser consultados no guia do utilizador, disponível em http://www.gnu.org/software/gawk/manual.

SOBRE O AUTOR



Fábio Ferreira frequenta desde 2005 o curso de Engenharia Informática e de Computadores no Instituo Superior Técnico, estando actualmente na área de especialização de Sistemas Inteligentes. Para além disto, tem interesse por algoritmia, Python e sistemas Unix.

fabio.ferreira@portugal-a-programar.org

Fábio Ferreira

Microsoft Windows Powershell

Introdução

O Powershell é basicamente uma "linguagem" de scripting criada pela Microsoft para administradores de sistemas, a sua syntax é um misto de perl,c++ e c#. O Powershell é distríbuido livremente e tem como grande objectivo a substituição de vbscript e batch scripting. Windows Powershell é um "ambiente" para cmdlets, funções, filtros, executáveis e aliases.

O PowerShell é baseado na Framework .NET. Assim, tem como principal vantagem a integração quase directa de componentes .NET.

Enquadramento

Este pequeno artigo tem como principal objectivo ilustrar as potencialidades desta "linguagem". Pessoalmente, já desde 2007 que tento influenciar todos aqueles administradores de sistemas ou helpdesk's para usarem esta ferramenta, pois acho que é extremamente poderosa e de enorme valia.

Historicamente, o cmd da Microsoft sempre foi bastante limitado, isto deve-se ao facto da Microsoft definir como "target" do seu sistema operativo, o utilizador comum que acaba por não ser muito dotado tecnicamente. Grande parte dos esforços no desenvolvimento do Windows foram apontados para o interface e a sua usabilidade, em detrimento da criação de um ambiente mais orientado para profissionais. Em suma, o PowerShell veio culmatar a falha que existia na gestão do sistema operativo Windows por linha de comandos.

Instalação

Para instalar o PowerShell, necessitam da Framework 2.0 da Microsoft, e de fazer download do respectivo pacote PowerShell
http://www.microsoft.com/windowsserver2003/technologies/management/powershell/download.mspx.
Após a instalação do PowerShell, devem aceder à consola do Windows PowerShell e executar o seguinte comando.

```
Set-Executionpolicy Unrestricted
```

Com este comando "vão" retirar todas as restrições de execução de scripts PowerShell na vossa máquina. O Powershell possui um mecanismo de segurança que assenta em assinaturas de scripts e permissões de execução. Fica aqui um apanhado dos possíveis perfis:

- Restricted – Não podem ser executadas scripts. O Windows PowerShell só pode ser usado através do powershell command mode.
- AllSigned – Apenas scripts assinadas podem ser executadas
- RemoteSigned – Downloaded Scripts têm que ser assinados para poderem ser executados.
- Unrestricted – Sem qualquer restrição de execução.

Breve Introdução

Façamos uma breve introdução à syntaxe da linguagem...

Variáveis

Têm que começar pelo simbolo $

```
$a=32
```

Pode ser escrito da seguinte forma para forçar o tipo de dados.

```
[int]$a=32
```

Arrays, hash's

```
# Isto é um comentário
$a= 1,2,3,4,5  # Array
$a= @(1,2,3,4,5) # Array

$loc = @{"Braga" = "Portugal";
"Madrid" = "Espanha"} #Hash
```

Comandos Essenciais

```
Get-Help
Get-Service
Get-Command
```

Criação de Instâncias

```
$d = New-Object -Type System.DateTime
2006,12,25
$d.get_DayOfWeek()
```

Output para a consola

```
Write-Host $a -foregroundcolor "green"
$a
```

Capturar input

```
$a = Read-Host "Nome ?"
Write-Host "Olá " $a
```

Quebra de linha

```
Get-Process | `
Select-Object `
name, ID Comments
```

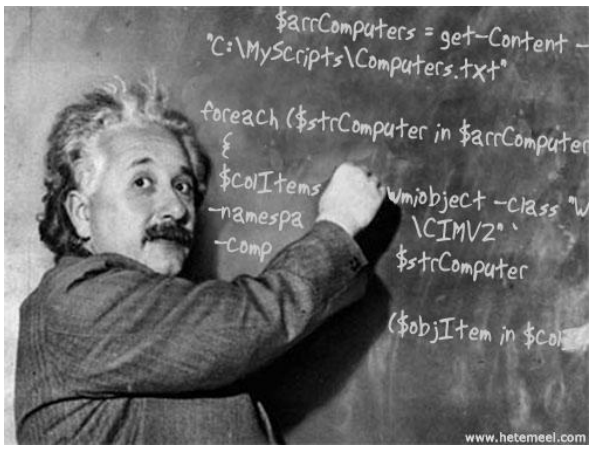
Ciclos, Loops

```
$a=1
Do
{$a; $a++}
While ($a -lt 10)

$a=1 Do {$a; $a++} Until ($a -gt 10)

For ($a=1; $a -le 10; $a++)
{$a}

Foreach ($i in Get-Childitem
c:\windows)
{$i.name; $i.creationtime}
```

Controle If-Then-Else

```
$a = "white"
if ($a -eq "red") {"Vermelho"}
elseif ($a -eq "white") {"Branco"}
else {"nao sei"}

$a = "red" switch ($a) {
"red" {"vermelho"}
"white"{"branco"}
default{"nao sei"} }
```

Programming, Programming ...



Nota: O conceito de pipe "|" está muito enraizado no uso do Powershell !

- Imprimir todos os processos da minha máquina.

```
get-process | ForEach-Object { write-
host $_.ProcessName $_.CPU}

# Ordenando-os por tempo de CPU
get-process |Sort-Object CPU

#Guardando o resultado dos primeiros
10 numa variavel
$P = get-process | Sort-Object CPU -
descending | select-object -first 10

#fazendo DUMP da variavel para um
ficheiro
$P > c:\A4.txt
```

O cmdlet get-process retorna todos uma lista de processos em execução. No exemplo anterior, bastou redireccionar a saída("output") do comando para um loop foreach para ser possível "trabalhar" o resultado.

A variável interna $_ armazena a saída do "|".

Como é que se sabe quais as propriedades que o objecto $_ possui? simples, usa-se o cmdlet get-member . Este cmdlet permite retornar todas as propriedades e métodos de um dado objecto.

Experimentem o seguinte :

```
get-process | select-object -first 1
| get-member
```

Se desejarem apenas saber quais as propriedades, basta filtrarem pelo -MemberType

```
get-process  | select-object -first 1
| get-member -MemberType property
```

Eis o "dump" do último comando ...

Name	MemberType
BasePriority	Property
Container	Property
Id	Property
MachineName	Property
MainModule	Property
.....	Property
.....	Property
Threads	Property
TotalProcessorTime	Property
UserProcessorTime	Property
VirtualMemorySize	Property
VirtualMemorySize64	Property
WorkingSet	Property
WorkingSet64	Property

• Ver o tamanho do meu event log "System"

```
get-eventlog -list | where-object
{$_.logdisplayname -eq "System"}
```

• Listar os últimos 3 eventos "System", formatando-os

```
get-eventlog system -newest 3 |
format-list
```

• Listar todos os ficheiros excepto os temporários, imprimindo o nome e o tamanho

```
Get-childitem c:\* -exclude *.tmp |
select-object name, length
```

• Listar todos os serviços e exportá-los para HTML

```
get-service | convertto-html

#exportá-los  para um ficheiro
get-service  |  convertto-html  >
.\a10.html
```

• Algo mais elaborado... Listar todos os meus serviços, exportá-los para HTML, separando por cores os que estão a correr dos que estão parados.

```
get-service | ConvertTo-Html -Property
Name,Status | `
foreach { if($_ -like
"*<td>Running</td>*") {$_ -replace
"<td>", "<tr bgcolor=green>"} `
else {$_ -replace "<tr>", "<tr
bgcolor=red>"}} >.\get-service.html
```

Com estes pequenos exemplos verificamos as enormes potencialidades desta "linguagem".
Agora um exemplo maior:

```
# Todos os ficheiros, recursivamente,
cujo tamanho seja maior que 20MB
get-childitem -recurse c:\temp |
where-object {$_.length -gt 2000000}

# Todos os ficheiros *.Doc que nÃ£o
sejam readonly
get-childitem *.doc | foreach-object
{$_.Isreadonly = 0}

# Ler um ficheiro
$a = Get-Content "c:\servers.txt"
foreach ($i in $a) {$i}

# Criar um ficheiro excel (2003) com
os dados de um serviÃ§o.
$a = new-object -comobject
excel.application
$a.Visible = $True
$b = $a.Workbooks.Add()
$c = $b.Worksheets.Item(1)
$c.Cells.Item(1,1) = "Nome Serviço"
$c.Cells.Item(1,2) = "Estado"
$i = 2
get-service | foreach-object{
$c.cells.item($i,1) = $_.name;
$c.cells.item($i,2) = $_.status;
$i=$i+1}
$b.SaveAs("c:\Test.xls")
$a.Quit()

# Who Am I
[System.Security.Principal.WindowsIdent
ity]::GetCurrent().Name
```

Juntemo-nos ao "lado negro da força..."

Construamos alguns scripts mais vocacionados para administração da máquina propriamente dita.

- Firewall

```
$profile = (new-object -com
HNetCfg.FwMgr).LocalPolicy.CurrentProfi
le

# Lista portas abertas
$profile.Services | ? {$_.Enabled} |
select -expand GloballyOpenPorts

# Lista aplicações Autorizadas
$profile.AuthorizedApplications | ?
{$_.Enabled} | ft name
# Lista serviços
$profile.Services | ? {$_.Enabled} |
ft name
```

- HotFixes instalados no sistema

```
$strComputer = "."

$colItems = get-wmiobject -class
"Win32_QuickFixEngineering" -namespace
"root\CIMV2" -computername
$strComputer

foreach ($objItem in $colItems) {
      write-host "Origem: "
$objItem.Caption
      write-host "CS: "
$objItem.CSName
      write-host "Descrição: "
$objItem.Description
      write-host "Comentários: "
$objItem.FixComments
      write-host "HotFix ID: "
$objItem.HotFixID
      write-host "Data Instalação: "
$objItem.InstallDate
      write-host "Nome: "
$objItem.Name
      write-host "Service Pack : "
$objItem.ServicePackInEffect
      write-host "Estado: "
$objItem.Status
      write-host
}
```

Neste último exemplo foi usado o cmdlet get-wmiobject. Este cmdlet faz o interface com o WMI (Windows Management Instrumentation) do Windows. O WMI é uma API do Sistema Operativo Windows que permite a obtenção da mais variada informação sobre a máquina em si. O WMI funciona por perguntas à diversas classes. Para mais informações sobre esta API usem o url http://msdn.microsoft.com/en-us/library/aa384642(VS.85).aspx

- Listar as partilhas de rede

```
$strComputer = "."

$colItems = get-wmiobject -class
"Win32_Share" -namespace "root\CIMV2"
-computername $strComputer

foreach ($objItem in $colItems) {
      write-host "Máscara Acesso: "
$objItem.AccessMask
      write-host "Maximo Permitido: "
$objItem.AllowMaximum
      write-host "Descrição: "
$objItem.Description
      write-host "Nome: "
$objItem.Name
      write-host "Path: "
$objItem.Path
      write-host "Estado: "
$objItem.Status
      write-host "Tipo: "
$objItem.Type
      write-host
}
```

• LoggOff ShutDown, Reboot

```
# Fazer Logoff na própria máquina
$strComputer = "."
(Get-WmiObject -Class
Win32_OperatingSystem -ComputerName
$strComputer).InvokeMethod("Win32Shutdo
wn",0)

# Outra forma de fazer loggOff
(gwmi
Win32_OperatingSystem).Win32Shutdown(0)


# Fazer Logoff a uma máquina remota
$strComputer = "Servidor-Blade"
(gwmi win32_operatingsystem -
ComputerName $strComputer
).Win32Shutdown(0)

# Fazer Logoff a uma máquina remota
pedindo credencias
(gwmi win32_operatingsystem -
ComputerName $strComputer  -cred (get-
credential)).Win32Shutdown(0)

# Fazer Shutdown
(gwmi
win32_operatingsystem).Win32Shutdown(8)


# Forçar Shutdown
(gwmi
win32_operatingsystem).Win32Shutdown(12
)

# Fazer Reboot
(gwmi
win32_operatingsystem).Win32Shutdown(2)


# Forçar Reboot
(gwmi
win32_operatingsystem).Win32Shutdown(6)
```

Variáveis especiais

Variável	Descrição
$_	Resultado do Pipeline; Usado em blocos de scripts,filtros, funções, where-object for-each, etc
$$	Contém o último token do último input
$?	Contém o estado de sucesso/falha da última operação
$Args	Usado na criação de funções ao qual é exigida uma passagem de parâmetros
$Error	Contém o último erro gerado
$foreach	Refere-se ao enumerador do ciclo for-
$HOME	A directoria do utilizador
$input	Input pipe para uma função ou bloco de código
$Match	Tabela de Hash com os elementos encontrados pelo operador -match
$MyInvocation	Informação sobre o script actual
$Host	Host actual
$LastExitCode	O codigo de saída da última aplicação nativa que foi executada
$true	TRUE
$false	FALSE
$null	NULL
$OFS	Campo Separador de Output
$ShellID	O Identificador da SHELL
$StackTrace	Contém o StackTrace do último erro

Ajudas

Caso necessitemos de aceder à ajuda de algum cmdlet, basta invocar o comando get-Help

```
#ajuda sobre o cmdlet export-csv
get-Help export-Csv -full

#listar todos os comandos disponiveis
que contenham a string "A"
get-Command | Where {$_.name -match
"A"}
```

Alguns Providers

Normalmente para pesquisar todos os Aliases basta usarmos o seguinte comando:

```
Get-Alias
```

Nome	Aliases	Descrição
Get-Location	gl	Directoria corrent
Set-Location	sl	Muda de directoria
Copy-Item	cpi	Copia ficheiros
Remove-Item	ri	Remove um ficheiro/directoria
Move-Item	mi	Move um ficheiro
Rename-Item	rni	Muda o nome a um ficheiro
New-Item	ni	Cria uma nova directoria
Clear-Item	cli	Limpa o conteudo de um ficheiro
Set-Item	si	Define o conteudo de um ficheiro
Mkdir	n/a	Cria uma directoria
Get-Content	gc	Envia o conteudo de um ficheiro para o output
Set-Content	sc	Define o conteudo de um ficheiro

Conclusão

Os benefícios desta linguagem são óbvios. Tendo sido desenvolvida da robustez e extensibilidade da framework .NET , o PowerShell permite-nos desenvolver poderosos componentes, de uma forma eficaz e rápida. Pessoalmente penso que é um enorme upgrade em diversos níveis. Espero que vos seja útil.

Referências

Windows PowerShell blog http://blogs.msdn.com/PowerShell/

PowerShell NewsGroup http://www.microsoft.com/communities/newsgroups/list/en-us/default.aspx?dg=microsoft.public.windows.powershell

Windows PowerShell http://msdn.microsoft.com/en-us/library/aa830112.aspx

PowerGui http://www.powergui.org/

SOBRE O AUTOR

Paulo Cabral é Engenheiro de Sistemas e Informática.
Tem 10 anos de experiência na área, estando actualmente inserido num projecto de I&D numa instituição de ensino superior.

paulo.cabral@portugal-a-programar.org

Paulo Cabral

Linux Graphics Stack

Este artigo tenta explicar a arquitectura da camada de gráficos do Linux e fornecer algum contexto sobre a sua evolução, os seus problemas e futuras soluções em desenvolvimento pela comunidade de Software Livre.

No início...

A camada de gráficos do Linux assenta principalmente sobre o servidor gráfico X. Este servidor gráfico começou o seu desenvolvimento em 1984 e quando apareceu tinha algumas capacidades técnicas inovadoras. Uma delas é a sua arquitectura baseada no conceito de comunicação cliente-servidor. Através desta arquitectura, consegue características como transparência de rede, isto é, a máquina onde a aplicação corre não precisa de ser a mesma onde a aplicação está a ser apresentada.

Outra característica interessante é que o servidor gráfico está separado do sistema de janelas, o que permite a criação de diferentes gestores de janelas conforme as necessidades dos utilizadores do sistema. Permite também a criação de vários servidores numa mesma máquina, podendo assim servir vários utilizadores.

Até há uns anos atrás, a implementação mais conhecida e utilizada do X era o Xfree86. Esta era mesmo reconhecida como a implementação "de-facto" e praticamente todos os sistemas baseados em Linux ou mesmo outros núcleos de sistemas operativos baseados em Unix (como a família BSD) utilizavam esta implementação.

Um servidor gráfico como o X é um projecto composto por muitos sub-projectos diferentes (gestores de dispositivos - "drivers", sistemas de janelas - "window managers", entre outros). Ao longo dos anos também se foi acumulando muito código (milhões de linhas de código!). Entre outros factores, isto tornou o projecto bastante complicado de gerir e entender.

Revolúcion, Revolúcion!

Em 2004, devido a problemas na organização, problemas de estagnação e incapacidade de de inovação, aliados a uma mudança de licença (para uma incompatível com a mais utilizada do Software Livre, a GPL), os membros do Xfree86 criaram um novo ramo do projecto. Esse novo ramo é hoje conhecido como X.Org, e em relativamente poucos anos, conseguiu destronar o Xfree86 como implementação de referência do X, e como implementação mais usada nas distribuições de sistemas operativos Unix.

Uma das grandes novidades do X.Org foi a modularização do código, isto é, a separação dos componentes em módulos com interfaces bem definidas. Embora esta funcionalidade por si só não traga vantagens concretas para os utilizadores, a longo prazo permitiu uma evolução do servidor sem causar grande instabilidade. Também permitiu lançar assincronamente novas versões dos componentes.

Apesar de, até hoje, ter ocorrido uma grande evolução desde os dias do Xfree86, a arquitectura da camada de gráficos do Linux continua obsoleta em comparação com outros sistemas operativos. Vamos analisar as suas componentes.

Arquitectura actual do sistema

Numa visão geral simplista, o servidor gráfico comunica com o núcleo do sistema operativo (Linux kernel) para aceder e comunicar com o hardware gráfico (placa gráfica), enviando comandos para serem processados pela placa. Esta interacção envolve tecnologias do kernel e do X.

No kernel, o módulo responsável pela comunicação com o mundo exterior é conhecido como DRM (não, não é aquela tecnologia que restringe os direitos dos ficheiros multimédia, mas sim o Direct Rendering Manager). Este módulo tem várias responsabilidades, como a sincronização dos pedidos para a placa gráfica por diferentes programas e um controlo rigoroso de acesso dos recursos disponibilizados pela mesma. Este módulo necessita de algum código específico para cada família de placas gráficas.

Do lado do servidor as coisas são um pouco mais complexas. Existem mais componentes e a sua interacção por vezes não é muito clara. O módulo que efectua a comunicação com o DRM é conhecido como DRI (Direct Rendering Infrastructure) e comunica com a placa, enviando os pedidos dos clientes do servidor X. Os gestores de dispositivos (drivers) têm de comunicar com esta componente para fornecer aceleração gráfica 3D.

Os drivers e a camada DRI também estão dependentes de outra componente, Mesa 3D. Esta componente é uma implementação completamente em software (sem recurso da placa gráfica) da biblioteca gráfica OpenGL. Assim, é possível ter suporte para gráficos 3D mesmo sem uma placa gráfica, embora obviamente seja bastante mais lento. Cada driver implementa uma pequena camada da sua funcionalidade no Mesa, que transforma os pedidos dos programas no formato da biblioteca OpenGL em instruções

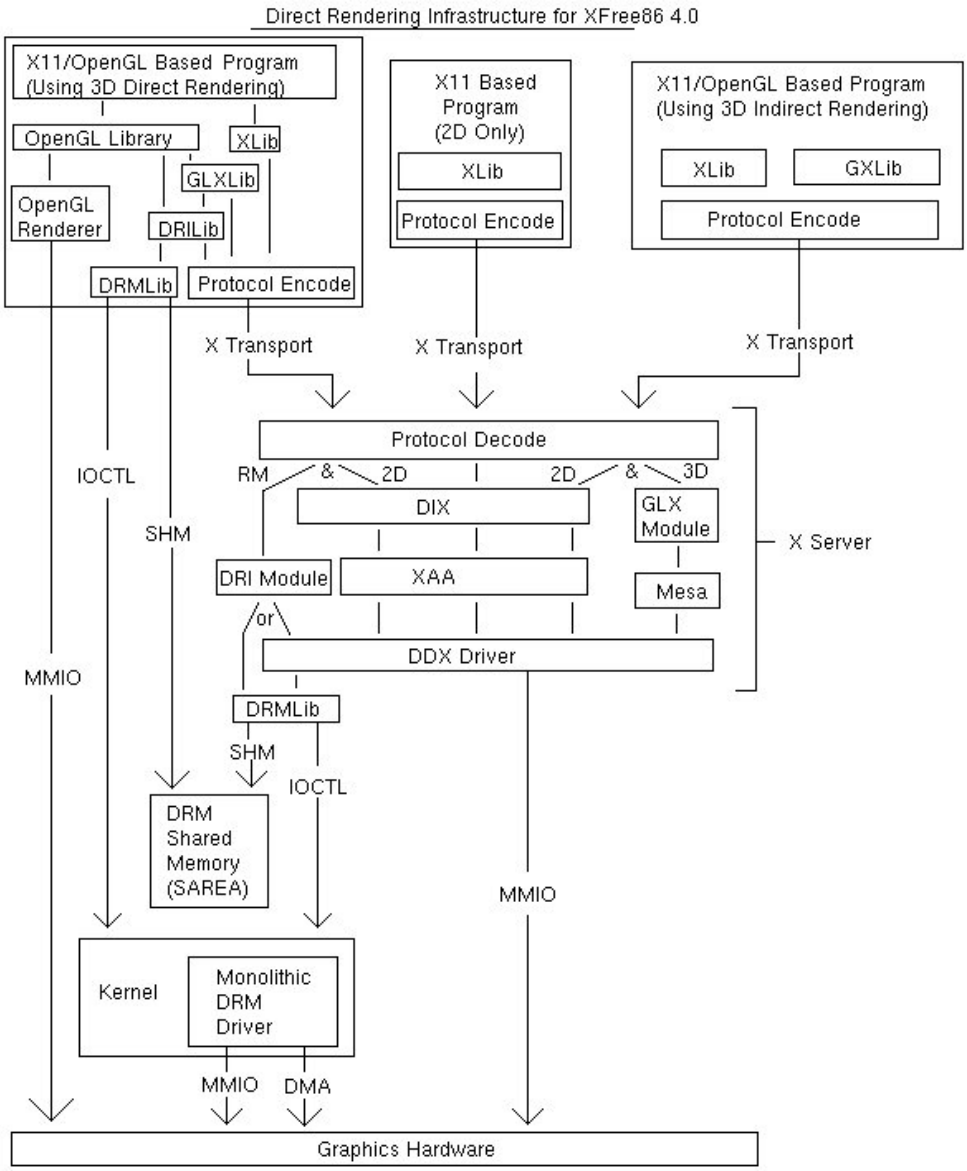
específicas para cada placa gráfica, atingindo assim aceleração gráfica 3D. As funcionalidades não suportadas pela placa gráfica podem ser "emuladas" no processador, utilizando a implementação do OpenGL via software, como previamente referido.

Hoje em dia as placas gráficas são bem diferentes do que eram há uns anos atrás, quando grande parte da arquitectura do sistema gráfico X foi desenhada. Enquanto antigamente a placa gráfica dedicava grande parte da área de silício a funcionalidades relacionadas com gráficos 2D, hoje em dia essas funcionalidades são menos exploradas. Grande parte da placa gráfica é dedicada para a apresentação de gráficos 3D e processamento de shaders. As placas gráficas modernas também apresentam capacidades consideráveis de memória de vídeo (chamada VRAM) que precisa de ser gerida, tal como a memória principal do sistema "RAM".

As novas versões do OpenGL também fazem uso de funcionalidades como FBOs ("Framebuffer objects") que permitem fazer render directamente para uma zona de memória da placa gráfica. Estas funcionalidades não são bem suportadas pela arquitectura actual, pois não existe um gestor de memória global para a placa gráfica. É por isso que grande parte dos drivers existentes no X não suportam versões recente do OpenGL, à excepção dos drivers da NVIDIA, que não utilizam grande parte da plataforma do X.

A arquitectura do X não foi desenhada com nenhum destes factores em conta, o que leva a que ao longo dos anos tenham sido aplicados muitos remendos e soluções pouco eficientes para conseguir adicionar as novas funcionalidades suportadas pelos novos modelos de placas gráficas constantemente lançados no mercado.

Vamos analisar mais profundamente a interacção do hardware gráfico com o kernel e com o sistema gráfico.



Arranque do computador

Quando o computador arranca (depois da BIOS), o kernel começa por inicializar o hardware. Em computadores com placa gráfica, esta também é inicializada. Como cada placa tem uma constituição interna diferente (registos de hardware, por exemplo) cada uma precisa de um código de inicialização diferente (até mesmo em famílias de hardware, este código tem ligeiras alterações). Esta fase de inicialização do hardware é chamada de modesetting, e até há pouco tempo, era feito completamente no servidor X. Então, perguntam vocês, como é que quando ligam o computador, obtêm imagem no monitor antes de iniciar o servidor X? Ou até mesmo antes de arrancar o kernel, na BIOS?

Isto é possível devido ao standard VGA definido pela IBM em 1987, que consiste numa interface para mostrar texto e gráficos no ecrã. Esta interface é implementada por todos os fabricantes de chipsets gráficos, o que permite ao kernel comunicar directamente com o hardware, sem ter de escrever código para cada modelo diferente. Quando o computador arranca, a placa é inicializada em VGA (modo texto), com uma resolução de 640x480 (existem outras disponíveis).

O problema do VGA é que como já se devem ter apercebido, tem limitações nos modos gráficos disponíveis e as resoluções que suporta são impensáveis nos dias de hoje. Para contornar estas limitações, foi criado outro standard, chamado VBE, ou VESA BIOS Extensions (http://en.wikipedia.org/wiki/VESA_BIOS_Extensions), que fornece mais modos gráficos. O código para mudar de modos continua a ser implementado por cada placa gráfica (na Video BIOS). Um dos problemas deste standard, é que a comunicação com a placa gráfica tem de ser feito em modo real (16-bits) e na altura em que foi definido, os sistemas operativos estavam todos a transitar para modo protegido (32-bits), o que dificulta a utilização deste standard e sofre de problemas de performance devido à transição entre modos.

Em termos do kernel do Linux, isto é implementado nos drivers de framebuffer (framebuffer drivers). Para perceber o que é isto do framebuffer, vamos olhar para a documentação do kernel (http://git.kernel.org/?p=linux/kernel/git/torvalds/linux-2.6.git;a=blob_plain;f=Documentation/fb/framebuffer.txt):

The frame buffer device provides an abstraction for the graphics hardware. It represents the frame buffer of some video hardware and allows application software to access the graphics hardware through a well-defined interface, so the software doesn't need to know anything about the low-level (hardware register) stuff.

Existem diferentes drivers de framebuffer para cada placa gráfica, alguns utilizando os standards que já referi, como o VGA e VESA. E outros para cada família de chipsets gráficos (da NVIDIA, Intel, ATI, etc). Mais concretamente, no código fonte do kernel, facilmente se encontram:

- vga16fb (VGA 16 cores)
- vesafb (VBE 2.0)
- uvesafb (VBE 2.0+)
- intelfb (chipsets da Intel)
- entre outros...

Então, como já vimos, o kernel quando arranca, carrega um destes módulos que vão ser responsáveis pela inicialização da placa gráfica e output de imagem para o ecrã. Enquanto se trabalha sem o servidor gráfico X, tudo funciona relativamente bem. Quando se inicializa o X, este vai re-inicializar a placa gráfica, utilizando o código de modesetting que existe em cada driver do X.

O mesmo acontece quando se muda do X para um terminal em modo texto (o chamado VT switching). O kernel vai reinicializar a placa utilizando o código do driver de framebuffer. É por esta razão que a mudança de terminais virtuais (VTs) quando envolve o X é tão lenta e cheia de "tremeliques". Nos portáveis e outros dispositivos móveis, quando se suspende ou hiberna, acontece o mesmo problema, estando a placa constantemente a ser reinicializada. Isto leva a que existam conflitos de acesso entre o kernel e o X e que muitas vezes o resumo do computador não seja efectuado correctamente, pois a placa fica num estado de inicialização incorrecto.

Remodelação de arquitectura de gráficos

Os responsáveis pelo desenvolvimento dos gráficos no servidor X e no Linux têm vindo a desenvolver novas soluções para cada um dos problemas anteriormente referidos. Esta transição tem sido gradual, até porque não podia ser feita doutra maneira num universo de Software Livre, e as grandes novidades estão quase a chegar nas novas versões do kernel e do X.

Um dos primeiros componentes que foi desenvolvido foi um gestor de memória para as necessidades das placas gráficas modernas. Um primeiro gestor de memória, chamado TTM (Translation Table Maps) foi arquitectado e desenvolvido pela Tungsten Graphics (http://www.tungstengraphics.com/wiki/index.php/TTM_Memory_Manager).

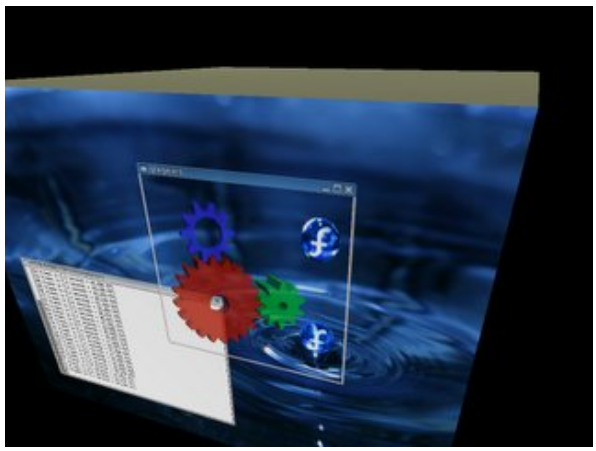
A Intel, por achar a interface do TTM demasiado complexa, desenvolveu outro gestor de memória, chamado GEM (Graphics Execution Manager, http://en.wikipedia.org/wiki/Graphics_Execution_Manager). Podem encontrar uma comparação entre os dois gestores de memória aqui: http://lwn.net/Articles/283793/. O GEM foi

incorporado na versão 2.26.28 do kernel (http://wiki.kernelnewbies.org/Linux_2_6_28) e já foi adoptado no driver da Intel e na versão em desenvolvimento do driver radeon (driver de código aberto para chipsets ATI).

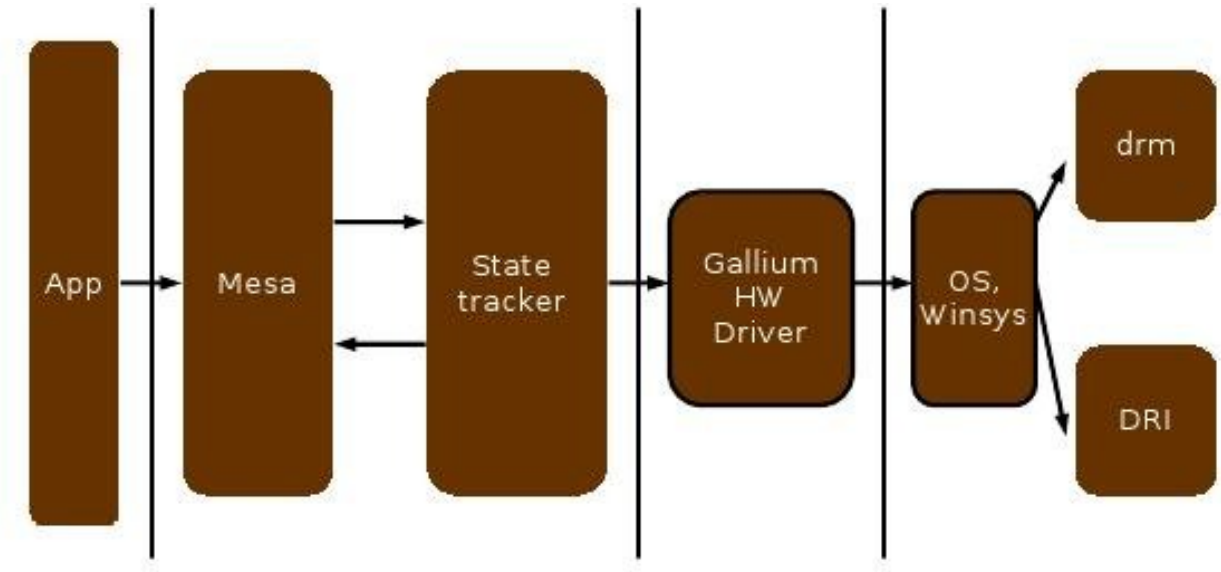
Ainda no kernel, todo o código relacionado com a inicialização e mudança de modos da placa gráfica (modesetting) foi transferido para o kernel, em vez da misturada que existe agora, em que o X e o kernel estão constantemente a re-inicializar os modos da placa gráfica. Esta funcionalidade foi integrada no kernel do Linux na versão 2.6.29, cuja versão final está prestes a sair.

Por agora, ainda só existe suporte para chipts Intel, mas já se encontram em desenvolvimento soluções para NVIDIA e ATI. Com este "kernel modesetting" vai ser possível efectuar mudanças entre VTs instantâneamente, arranque sem "tremeliques" e sem mudanças entre modo texto e modo gráfico, suporte para "Fast-user switching" e melhor capacidade de suspensão. Relacionado ainda com esta componente, a Red Hat encontra-se a desenvolver um novo gestor gráfico de arranque (não é um boot loader, como o GRUB!) para tirar partido destes novos desenvolvimentos. Chama-se Plymouth, e tudo indica que será adoptado pelas distribuições Fedora e Ubuntu nas suas próximas versões.

Outro projecto de remodelação na camada de gráficos em Linux, é a re-arquitectura do componente DRI, responsável pela comunicação com o DRM no kernel e com o driver específico para a placa no Mesa. Foi remodelado para suportar aceleração OpenGL "indirecta". Isto permite que jogos ou aplicações 3D funcionem bem com um gestor de janelas "compósito", como o Compiz.



O sistema de drivers do X e Mesa também está a sofrer uma remodelação, para suportar funcionalidades existentes em versões recentes do OpenGL, facilitar o desenvolvimento de novos drivers, remoção de código duplicado, maior optimização e suporte para várias APIs gráficas. Esta nova "framework" chama-se Gallium3D, e está a ser desenvolvida principalmente pela Tungsten Graphics.

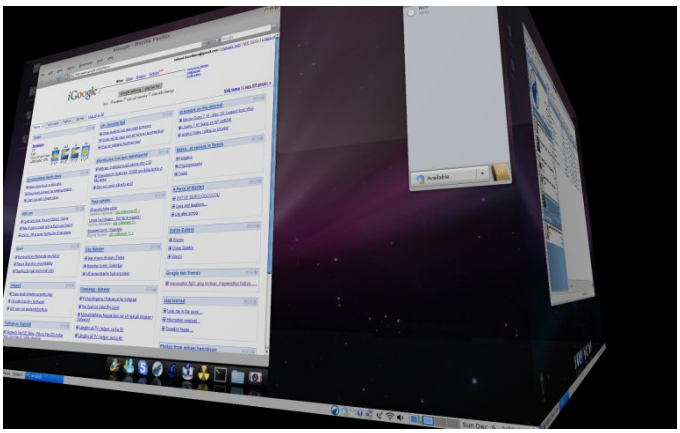


Esta biblioteca abstrai as diferentes APIs gráficas do modelo de programação do driver do chipset da placa gráfica,permitindo futuras adições de APIs aos drivers sem esforço adicional de programação. Um exemplo seria criar um "state tracker" para a API gráfica da Microsoft, Direct3D. Isto seria útil para executar jogos através do Wine, que tem de converter chamadas dos jogos para a API em chamadas equivalentes em OpenGL. Ou até mesmo para facilitar o "porte" de jogos de Windows para Linux sem necessidade de re-escrever o código do motor de jogo.

Outro aspecto interessante e inovador do Gallium3D, é que utiliza outra biblioteca de código fonte livre chamada LLVM (Low-level Virtual Machine) para optimizar o código dos programas ("shaders") que correm nas placas gráficas. O LLVM suporta muitos passos de optimização, e está a usar financiado e usado pela Apple para este efeito no sistema operativo Mac OS X e também como optimizador de código nos seus compiladores (o código gerado para o iPhone, por exemplo, é optimizado por esta biblioteca).

Outros desenvolvimentos a que se têm assistido "in the graphics land" são o aparecimento de novas APIs para aceleração de decoding de vídeos utilizando a capacidade de processamento da placa gráfica para libertar o processador dessa carga. Por parte da NVIDIA, houve o lançamento da API VDPAU (Video Decode and Presentation API for Unix) que suporta vários formatos modernos (H.264, VC-1). Por parte da Intel também apareceu o VA-API (Video Acceleration API), outra API semelhante. A AMD (ATI) também se encontra a desenvolver uma API para o mesmo efeito, chamada XvBA (X-Video Bitstream Acceleration). O tempo dirá qual irá prevalecer.

E ainda há quem esteja a desenvolver outro servidor gráfico, livre do X, para tirar partido de todas estas novas



tecnologias. O X já é maduro, e tem sido remodelado constantemente ao longo dos últimos anos, mas nada como tentar uma nova abordagem de raiz. Este novo servidor chama-se Wayland e está a ser desenvolvido pelo mesmo programador que fez grande parte do trabalho no componente DRI2. Tal como o próprio autor diz, o objectivo não é substituir o X, mas sim tentar novas abordagens e ideias que mais tarde possam ser utilizadas no X.

Estado actual da remodelação

Actualmente, diferentes famílias de placas gráficas suportam diferentes funcionalidades. Algumas estão bem encaminhadas nesta remodelação, outras nem por isso, principalmente por falta de investimento das empresas. A comunidade muitas vezes também não pode ajudar porque as empresas não lançam os manuais de programação dos chips das placas, pelo que se torna uma tarefa exageradamente difícil criar um driver de raiz.

A Intel tem contribuído bastante para toda esta remodelação, com o seu gestor de memória GEM e código para kernel modesetting nos seus chips, ambos já integrados na próxima versão do kernel do Linux (2.26.29). Além disso, continuam a lançar novas versões de drivers

para corrigir bugs e melhorar a performance. A Tugsten Graphics também tem um driver desenvolvido para a nova arquitectura de drivers, Gallium3D, que suporta os chipsets da Intel.

Os chipsets da ATI também têm vindo a ser cada vez melhorsuportados em Linux e no X. A AMD (que comprou a ATI no ano passado) recentemente lançou guias de programação para os chipsets gráficos, o que permitiu à comunidade de Software Livre melhorar consideravelmente os drivers de código aberto. Além disso, em parceria com a Novell, ajuda no desenvolvimento de um novo driver de código aberto para chipsets mais recentes (o driver chama-se RadeonHD). Este driver tem evoluido bastante nos últimos meses. Com esta documentação, será possível no futuro construir drivers Gallium3D para os chips da AMD/ATI.

Por parte da NVIDIA, não existe qualquer envolvimento com a comunidade de Software Livre nesta re-arquitectura da camada de gráficos. Existe um driver estável e com suporte

para versões recentes do OpenGL (3.1!) de código fechado (proprietário) e também um driver 2D bastante obfuscado de código aberto, ambos mantidos pela NVIDIA. O driver proprietário é considerado dos melhores drivers para o X e um dos únicos que suportam funcionalidades disponíveis em versões recentes do OpenGL.

Ainda falando dos chips da NVIDIA, existe um projecto da comunidade para desenvolver um driver completamente livre e código aberto. Chama-se nouveau e tem tido um progresso fantástico, tendo em conta que não existe qualquer documentação disponível dos chips (a NVIDIA não lança os guias de programação!) e o desenvolvimento tem de ser feito com ajuda de "reverse-engeneering" do driver proprietário. Para mais informações visitem o site do projecto (http://nouveau.freedesktop.org).

Conclusão

Podem continuar a seguir todos os desenvolvimentos no site Phoronix (http://www.phoronix.com) que nos últimos tempos se tem tornado uma excelente fonte de notícias para todos os assuntos relacionados com gráficos em Linux.

Espero que tenham gostado tanto deste artigo, como eu gostei de o escrever. Como devem ter percebido, a arquitectura dos gráficos em Linux (e em qualquer outro sistema operativo!) é complexa e em constante desenvolvimento, logo é normal que tenha digo algum disparate. Agradeço que me contactem com todas as vossas correcções, sugestões e críticas.

SOBRE O AUTOR

João Matos, também conhecido por triton, frequenta o segundo ano do curso de Engenharia Informática e Computadores no IST. Tem como principais interesses as áreas de Sistemas Operativos, Computação Gráfica e Arquitectura de Computadores.

joao.matos@portugal-a-programar.org

João Matos

Smashing Magazine

Blog dedicado a oferecer aos seus leitores dicas e a informá-los das últimas tendências e técnicas em todos os assuntos relacionados com o webdevelopment. Um óptimo repositório de ideias gráficas, assim como de imagens grátis e técnicas para a criação e manutenção de websites.

http://www.smashingmagazine.com/

ThemeForest

ThemeForest é um site de compra e vendas de templates. Aqui é possível encontrar, a baixos preços, templates para um website - desde templates exclusivas para Wordpress a simples templates (x)HTML. Existe uma enorme variedade de temas, onde se destacam as templates para painéis de administração, tão difíceis de encontrar na internet. Possibilita também a venda de templates, recebendo o seu autor uma percentagem do lucro, aumentando a mesma caso a template seja um sucesso de vendas.

http://www.themeforest.net/

Wiki P@P

A Wiki P@P é plataforma Wiki da comunidade portuguesa de programadores, Portugal-a-Programar. Nesta poderão ser encontrados todo o tipo de recursos de teor informativo, tutoriais ou snippets, fruto do contributo dos utilizadores.
Qualquer um pode contribuir para o projecto, pois só assim este vive e subsiste, isto porque uma plataforma Wiki é uma plataforma de trabalho cooperativo.

http://wiki.portugal-a-programar.org/

Joel on Software

Joel on Software é um blog que fala sobre o desenvolvimento de software, gestão de negócios, e da Internet desde 2000, sem aquela teoria maçadora, mostrando o lado mais prático das situações. O seu autor já escreveu quatro livros, sobre temas falados no seu blog, e dá várias palestras por ano, para os interessados na área.

http://www.joelonsoftware.com/

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

Vai a

www.revista-programar.info

para mais informações em como participar,
ou então contacta-nos por

revistaprogramar
@portugal-a-programar.org

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

contamos com a tua ajuda!

Equipa PROGRAMAR

Um projecto Portugal-a-Programar.org