MacPRO
o suplemento dos power users
Ano 4 - Nº29
Parte integrante da revista Macmania
Não pode ser vendido separadamente

ProNotas

Resolução do novo milênio

Scanner profissional cilíndrico tem resolução de 12 mil dpi

A Gutenberg, empresa especializada em máquinas e materiais gráficos, trouxe para o Brasil novos scanners profissionais de cilindro da **ICG** com resoluções superiores a 12 mil dpi.

São dois modelos, o **370HS** e **370S**, ambos com cilindros verticais e equipados com o ScanXact II, um software de escaneamento e separação de cores. Como opcionais, a Gutenberg oferece dois recursos: a tecnologia CopyDot (para reprodução de originais transparentes ou opacos para o sistema digital) e o software SendDirect (para transferir automaticamente o original escaneado para uma estação de retoque da rede). Os scanners geram arquivos nos formatos RGB ou CMYK, TIFF ou DCS, a partir da digitalização dos originais transparentes, com uma faixa de densidade de 0-4D, que garante, segundo os técnicos da Gutenberg, mais detalhes de sombras e saturação de cores para tonalidades escuras.

Os cilindros são montados na vertical, facilitando a sua troca e manutenção, e o dispositivo de foco é automático ou manual. A escala de ampliação vai de 20 a 6.000% no modelo 370HS e até 2.000% no 370S. Mais informações no site da Gutenberg.

Gutenberg: www.gutenberg.com.br

A evolução do Darwin

Apple modifica licença open source do OS X

Alguns desenvolvedores já acharam incrível a Apple ter liberado o código fonte do núcleo do Mac OS X, o **Darwin**, mesmo com várias restrições. Agora, as liberdades são bem maiores.

A Apple modificou a licença de uso para o código fonte público de seus produtos, Apple Public Source License 1.2, eliminando as amarras que não agradavam os programadores da comunidade open source, que consideravam a versão anterior da licença inaceitável. Segundo eles, a APSL 1.1 obrigava a publicação das mudanças feitas no programa mesmo, que elas fossem criadas exclusivamente para fins pessoais e não comerciais, e também a comunicar a Apple de qualquer alteração. Além disso, a empresa poderia terminar o contrato de licença a qualquer momento se o programador se tornasse alvo de problemas com violação de patentes.

A versão 1.2 da licença retirou essas restrições, não exigindo sequer a publicação das alterações, a não ser em caso de distribuição externa.

A Apple afirmou que a APSL 1.2 torna mais fácil a contribuição dos desenvolvedores e também o uso ▶

A primeira xícara de Cocoa

Veja como é fácil criar um programa no OS X

por Fabio G. Ribeiro

O Mac OS X não é uma revolução. São várias. E uma das principais é a revolução que ele traz na maneira de se criar programas para o Mac, com integração total com Java, compatibilidade com programas feitos para outros "sabores" de Unix e ferramentas modernas (e gratuitas). E é sobre essas ferramentas que iremos falar aqui. Para mostrar sua importância, basta dizer que a primeira coisa que Steve Jobs fez ao voltar a Apple em 97 foi subir no palco da Macworld em San Francisco para demonstrar as ferramentas de desenvolvimento do OpenStep, as mesmas que hoje fazem parte do kit de desenvolvimento do Mac OS X. Com elas, diz Jobs, *"os programadores poderão desenvolver programas com 20% do esforço que é necessário para programar para o Windows ou Mac OS"*.

Nesta série de matérias estaremos mostrando algumas das coisas interessantes que o desenvolvedor pode fazer usando as ferramentas gratuitas do Mac OS X.

O exemplo a seguir pode ser seguido utilizando o Mac OS X Server, Mac OS X DP4 ou o Mac OS X Public Beta 1. No caso deste último, você precisará das ferramentas de desenvolvimento disponíveis para membros filiados ao ADC (http://connect.apple.com). Filiados ao plano online têm acesso a essas ferramentas gratuitamente. Em virtude da ampla difusão da linguagem Java, ela será utilizada nesses exemplos. Se você não tem muita intimidade com Java, não se assuste; não chego a me aprofundar muito na linguagem. Qualquer pessoa poderá ler os comentários e entender com facilidade o que fiz. Mas é bom que os interessados em programar no Mac OS X se empenhem em conhecer Java ou Objective-C, as duas principais linguagens para desenvolvimento nesse sistema.

O Cocoa é a API e uma das bases do OS X

As ferramentas

Ao instalar o kit de desenvolvimento você verá no diretório **Mac OSX/Developer/Applications** as ferramentas que você precisa para começar sua vida de programador no OS X. As principais são:

Project Builder: permite que você crie, gerencie e edite seu código, construa e depure os aplicativos

Interface Builder: um ResEdit elevado à décima potência. Permite que você crie a interface, conecte elementos gráficos a ações do seu programa, crie classes etc. Veremos isso mais adiante.

Background

Ao programar no OS X, você verá com frequência os termos Cocoa Framework, Core Foundation e AppKit. **Cocoa** é a API (Application Programming Interface) do OS X. A Cocoa Framework encapsula o AppKit e o FoundationKit, que é o conjunto de estruturas, classes e funções que o sistema disponibiliza para a base do sistema operacional.

A **Kit** é responsável por funções como a manipulação de cadeias de caracteres (de modo que elas possam representar praticamente todas as línguas existentes no mundo), criação de threads, data, horário e fusos; enfim, aquelas relacionadas com a infra-estrutura do sistema/aplicativos.

O **AppKit** é onde trabalhamos com maior frequência num aplicativo. Contém as classes para que trabalhemos com a parte visual do aplicativo: janelas, menus, listas, botões, controles etc. Felizmente,é a parte mais fácil e é o maior foco da nossa série. ▶

A primeira xícara de Cocoa Continuação

A documentação de ambas está disponível gratuitamente no site da Apple:
http://developer.apple.com/techpubs/macosx/Cocoa/CocoaTopics.html
A organização das APIs em frameworks é uma aproximação da Apple. Entretanto, as tecnologias empacotadas num framework não são necessariamente dela, como o OpenGL, por exemplo.
Um framework para o programador pode conter uma ou mais versões da API com seus headers e bibliotecas. Tudo isso pode ser visto ao se importar um framework no Project Builder, o que facilita muito o trabalho em relação a outras ferramentas de desenvolvimento, como o CodeWarrior.

Project Builder

Nele você pode criar todo tipo de código possível para o OS X: aplicações, extensões de kernel, programas Cocoa, Bundles, Frameworks, Tool etc. O Mac OS X é todo escrito com o Project Builder (PBX).
Ele organiza,basicamente:
1. Arquivos *header* que contêm principalmente protótipos das funções que você cria ou que uma classe implementada.
2. Arquivos de interface criados pelo Interface Builder, chamados de arquivos *.nib*
3. Frameworks.
4. O código propriamente dito.

A interface melhorou muito em relação ao seu antecessor (a versão do Mac OS X Server). O fim da confusão de duas janelas para construir o programa e mais duas para depurar facilita bastante o trabalho do desenvolvedor iniciante.
O Project Builder do Mac OS X possui os principais conceitos da versão anterior; entretanto, foi reescrita do zero e muitas funcionalidades ainda devem ser adicionadas. A Apple possui uma lista de discussão para usuários do PBX (http://lists.apple.com). As principais funcionalidades são: integração com o Interface Builder, poder ser reescalonado através da inserção de compiladores adicionais (inclusive de terceiros), depurador intregrado etc. Virtualmente nada impede, portanto, que alguma empresa crie compiladores Pascal, Fortran, Assembler ou suporte para qualquer outra linguagem. Desenvolvedores que usam o MacApp, framework gratuito da Apple (C++), podem instalar a versão 15, lançada em janeiro, e serão capazes de criar aplicativos em C++ para Mac OS 9 e X (Carbon).
Uma grande vantagem é que ele usa padrões da indústria quanto ao desenvolvimento: compiladores (javac, jikes, gcc) e depuradores (gcc, jdb). É com ele ainda que você edita todas as propriedades do software que está criando, através de listas de propriedades (plist), e ele é o responsável pelo empacotamento dos aplicativos e frameworks que você estiver criando, da forma que ele for configurado.
Como o CodeWarrior, ele é capaz de trabalhar com diversos targets para um projeto e trabalhar com subprojetos, e você pode ligar a otimização do compilador. Claro que isso não é tudo para um programa mais rápido, mas ajuda substancialmente. Uma grande vantagem em relação ao CodeWarrior aparece no trabalho em grupo: o Project Builder guarda as preferências de projeto para cada usuário. Ao salvar os breakpoints para depuração, por exemplo, ele salva para o usuário que está trabalhando na questão, e eles não afetam a depuração do projeto quando ela for feita por outro usuário. Também possui suporte a SCM (Software Configuration Management), o que ajuda muito na hora de fazer controle de lançamento de softwares e versões num projeto grande.
E tudo isso por apenas *nada*. O PBX é gratuito. Além disso, a Apple está aberta a sugestões para melhoria do produto, para uma maior produtividade dos seus usuários, na lista de discussões do Project Builder, onde tira dúvidas quanto à ferramenta, e aceita (e realiza!) as sugestões propostas.

Interface Builder

É o ambiente onde você cria a parte visual do seu aplicativo e estabelece relações entre ela e o código. Ele permite que você crie as classes de seu programa, definindo métodos e variáveis de classe. Permite ainda definir as conexões, entre os elementos gráficos do aplicativo e a representação lógica destes elementos, que em geral são as próprias variáveis de classe.
No tutorial a seguir trabalharemos intensamente com as ferramentas de desenvolvimento Project Builder e Interface Builder.

Sobre o DNSSolver

Neste tutorial, vamos fazer um pequeno programa escrito em Java para resolver endereços de URLs. A intenção é mostrar como usar as ferramentas de desenvolvimento, criar classes, fazer conexões e estar familiarizado com os termos mais comuns no desenvolvimento no Mac OS X.

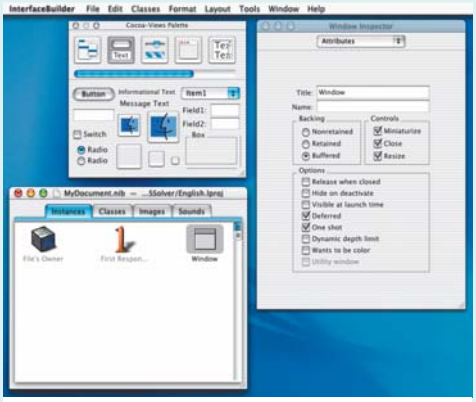
Criando um projeto

Abra o Project Builder e crie um novo projeto usando o menu File ▸ New Project. Selecione a opção Application ▸ Cocoa-Java Document-based Application, pressione o botão Next e dê o nome do projeto de DNSSolver, salvando-o no local mais conveniente (usando o botão Set)

Editando a interface

Na janela que surge, dê um duplo-clique em MyDocument.nib (que fica no grupo Resources), um dos arquivos do Interface Builder (IB) para este projeto. Não feche o projeto no Project Builder, pois precisaremos dele aberto. No IB você tem três janelas importantes: a do arquivo (a janela com quatro abas e o título MyDocument.nib, neste caso), a palete de itens que podem ser adicionados à interface do programa (Cocoa -Views-Palette) e a janela do Inspector, que exibe informações sobre o item selecionado.

A janela do documento 'MyDocument.nib'



As três principais janelas do Interface Builder

As quatro abas são:
Instances: mostra as instâncias de objetos que você cria no seu programa. Não são instâncias reais de objetos; são chamadas de *proxy instances* e usadas exclusivamente para se fazer conexões entre métodos, variáveis de instâncias e objetos, como veremos mais adiante.
Classes: possui as classes que compõem o AppKit e sua hierarquia, basicamente. Através dessa aba criamos nossas próprias classes, definindo os nomes das suas variáveis de instâncias e métodos.
Images e Sounds: nessas duas outras abas você pode armazenar sons e imagens que serão arquivadas no arquivo .nib em questão. Isso não será usado neste primeiro tutorial.
Na janela do arquivo MyDocument.nib, clique na aba Instances e dê um duplo-clique no ícone de janela (Window). Abra o Inspector (⌘1) e você será capaz de alterar atributos dessa janela; o menu pop-up da janela do Inspector dá acesso a outros atributos que podem ser alterados. Todos eles também podem ser alterados em tempo de execução com os métodos corretos. Tome algum tempo para inspecionar a janela do Inspector para a janela com as diversas opções que seu menu pop-up possui, e acostume-se a fazer isso para todos os outros itens (botões, listas, imagens, caixas de texto,etc.)

O paradigma MVC

É fortemente aconselhado pela Apple que, durante o desenvolvimento de softwares para Mac OS X, seja usado um paradigma que

chamamos de **MVC** (Model-View-Controller), que não temos muito espaço para discutir aqui, mas consiste basicamente em dividir o software em interface (View), parte lógica (Model) e a parte que liga as duas fazendo a interface com o usuário, podendo ser alterada caso exista alguma mudança nos dados. Suponha, por exemplo, que num programa cliente de FTP, dados chegam através de sua placa de rede e você tem como saber visualmente, através de uma barra de progresso, que agora ele tem 94% do download completo e não mais 93%. É possível fazer o caminho inverso: os dados do programa são alterados de maneira a refletir uma ação feita na interface, geralmente com o usuário ajustando um controle. Essa parte que une as duas é chamada de Controller.

Num programa simples como o que vamos fazer agora, esse paradigma não fica muito claro. Mas, em programas mais complexos, ele facilita muito o seu trabalho, sendo que algumas tecnologias novas do Mac OS X só podem ser implementadas com facilidade ao usá-lo. Para saber mais sobre esse paradigma, leia:

http://developer.apple.com/techpubs/macosx/Cocoa/ProgrammingTopics/AppDesign/applicationdesigntoc.html

A idéia do nosso programa

Nosso programa terá o seguinte objetivo:

1. Alguém digita um número IP ou URL e pressiona um botão.
2. O botão executa a ação de resolver o DNS (ou um *lookup* inverso, no caso de a pessoa ter digitado um IP).
3. No caso de sucesso, o resultado é mostrado num campo de texto e o título da janela fica igual ao IP ou nome solicitado.
4. No caso de falha, surge uma caixa de diálogo falando que não foi possivel resolver o endereço "meuEndereco.com".

Exemplo

Digitando `apple.com` e pressionando Solve DNS, você obtém como resposta `apple.com/17.254.3.183` sendo que o número é o IP da Apple (US).

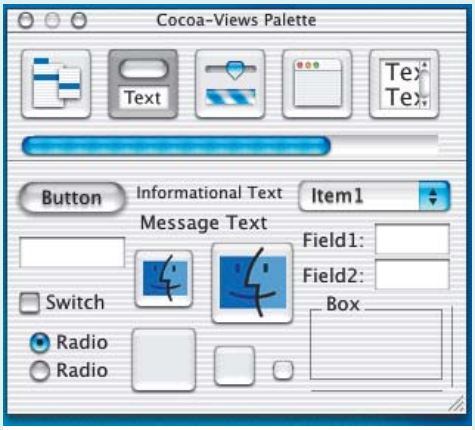
Para isso, temos que ter:

1 janela (já está lá!)
2 campos de texto
1 botão

Dê um duplo-clique na janela que está na parte Instances da janela do arquivo .nib e mova dois campos de texto e um botão para a janela que se abre. Deixe-os em um tamanho compatível. Sugiro fortemente, que sempre que você tenha dúvidas em relação ao layout de software no Mac OS consulte o livro **Inside Macintosh:Human Interface Guidelines** e também o novo **Adopting the Aqua Human Interface**

http://developer.apple.com/techpubs/macosx/SystemOverview/AdoptingAquaInterface.pdf

Mova ainda dois campos de texto estático para a janela, dê um duplo-clique e edite seus nomes para refletir o que eles devem conter. Todos os itens que você usará nesta parte estão na janela abaixo.



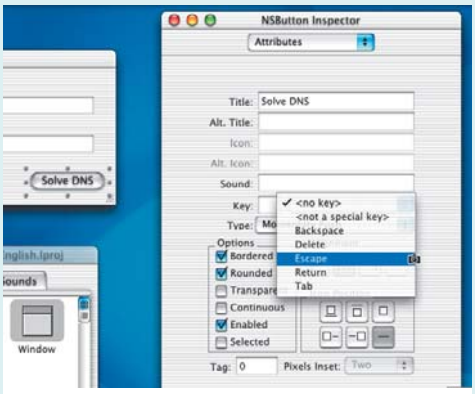
Cocoa-Views Palete com os elementos de interface usados neste exemplo

Minha janela ficou mais ou menos assim:

Janela do exemplo com os elementos necessários

Definindo um botão como padrão

Clique no botão e pressione ⌘1 para chamar o Inspector. Altere o menu que vem logo depois de Key de `<no key>` para `Return`. Isso torna esse o botão padrão para essa janela, deixando-o pulsando, e a ação relacionada a ele será disparada ao pressionarmos Return ou Enter.



Definindo um botão como o botão padrão

do programa e que as modificações foram feitas atendendo os pedidos da comunidade open source. Segundo os programadores, a nova licença da Apple ainda não é ideal, mas ela caminha na direção certa, acreditando que a quantidade de pessoas trabalhando no projeto poderá aumentar.

Darwin: `www.opensource.apple.com/apsl`

Programando para o futuro

Reunião de desenvolvedores para Mac deste ano vai se concentrar mais nos programas

A Apple divulgou como será o grande evento para desenvolvedores de Mac deste ano, a **Worldwide Developers Conference 2001** (WWDC). A reunião acontecerá entre os dias 21 e 25 de maio, na cidade de San Jose, Califórnia.

A conferência terá algumas palestras técnicas sobre equipamentos, mas o foco principal está centrado em software, como AppleScript, Aqua, Unix BSD, Carbon, Cocoa, Darwin e Java, entre outros. Durante o evento irá acontecer o sexto Apple Design Awards, que vai premiar os melhores aplicativos em quatro categorias (todas envolvendo o Mac OS X), como Melhor Produto e o Mais Inovador.

Para participar da WWDC 2001 é preciso estar registrado no grupo de desenvolvedores da Apple, que oferece as seguintes opções: Online (grátis), Select (US$ 500 por ano) ou Premier (US$ 3.500 por ano). A taxa de inscrição do WWDC é de US$ 1.595 para membros Online e US$ 1.395 para membros Select ou Premier. Esses preços valem até 20 de abril, e os primeiros 1.500 que se registrarem ganharão uma jaqueta de couro (que, segundo a Apple, vale US$ 250) com o logo do Mac OS X.

WWDC2001:
`www.apple.com/developer/wwdc2001`

OS X Server terá interface Aqua

Atualização do sistema operacional deve chegar no próximo trimestre

Que o Mac OS X está chegando, isso todo mundo sabe. Porém, a Apple revelou que uma atualização do seu sistema operacional para servidores, o **Mac OS X Server**, deverá sair no próximo trimestre, com modificações técnicas e cosméticas.

O novo OS X Server vai expandir as funções e ferramentas de adminstração do sistema operacional, como NetBoot, QuickTime Streaming Server e WebObjects. Adicionará um servidor DHCP, um firewall básico, e vai aceitar os updates do WebObjects (4.5.1 e o 5). Além disso, o OS X Server vai ter uma arquitetura de serviços de diretório (Directory Services Architecture) avançada, que usará ou o sistema de diretórios NetInfo embutido ou os servidores LDAP padrão. Uma versão do Samba (que permite troca de arquivos com clientes Windows) também estará incluída no sistema para servidores.

A novidade mais visível porém, está na cara do OS X Server, que vai ter a interface Aqua, a mesma do sistema operacional da Apple "para o resto de nós". A empresa ainda não definiu uma data de lançamento para a atualização, mas afirmou que isso deverá acontecer somente no próximo trimestre, depois que o Mac OS X tiver chegado às lojas.

Mac OS X Server:
`www.apple.com/macosx/server`

A primeira xícara de Cocoa Continuação

Mas que ação? Como eu associo uma ação a um botão ou me refiro a um item em particular na minha janela? Como eu controlo esses itens? Para isso, criaremos uma classe que controle a relação entre a interface e a sua relação com a parte lógica do software, o que será feito no próximo passo.

Criando uma nova classe

Para isso, clique na aba Classes e procure java.lang.Object (ele está na primeira linha), que é o objeto que está na raiz das classes Java (em última análise, todos os objetos Java são também objetos 'Object'). Para criar nossa classe, vamos criar uma subclasse dessa aqui. Selecione java.lang.Object e a partir do menu Classes ▸ Subclass. Mude o nome de MyObject para DNSSolverController; esse será o objeto que intermediará as ações do usuário com a parte lógica do programa e vice-versa.
Para fazer isso, seu programa deve acessar os dois campos de texto e disparar uma ação ao se clicar no botão que chamei de Solve DNS.

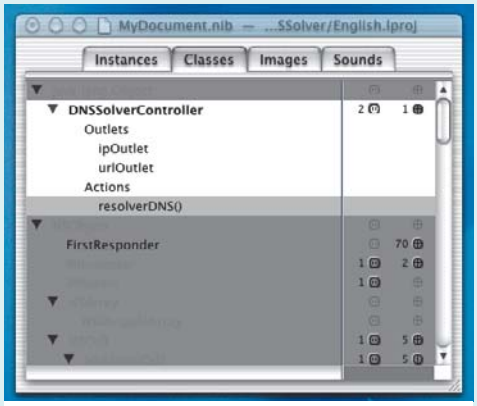
Criando outlets

Para ter acesso a um objeto (neste caso, cada um dos campos de texto), temos o conceito de *outlet*. Um outlet permite a comunicação entre a parte lógica do software e a parte de interface. Neste caso, devemos criar um outlet para cada campo de texto. Não precisamos alterar nenhuma propriedade do botão, como seu tamanho, ou acessar o seu título, mudar sua localização etc, mas precisamos apenas chamar uma função assim que ele é pressionado; não precisamos de um outlet. Isso ficará mais claro mais pra frente.
Selecione a classe DNSSolverController e depois o menu Classes ▸ Add Outlet. Mude o nome desse outlet para urlOutlet para o primeiro campo de texto, seguindo o mesmo passo para o segundo, chamando agora de ipOutlet.

Criando actions

A parte de outlets esta OK. Agora adicione uma ação que nosso objeto DNSSolverController possa realizar (no caso, resolverDNS). Selecione a classe DNSSolverController e depois o menu Classes ▸ Add Action, mude o nome da ação para resolverDNS() (os parênteses são colocados automaticamente se você se esquecer deles).
OK! Agora você tem os elementos necessários para começar a definir o que está ligado ao quê e quem chama a ação resolverDNS().
Você deve criar uma instância falsa (proxy instance) do objeto que você acabou de criar (DNSSolverController) para realizar as conexões. Para isso, selecione-o na aba Classes, e no menu Classes selecione Instantiate. Você é levado para a aba Instances e surge um novo objeto na sua palete: DNSSolverController. Abra a janela que está nessa palete (a que você editou algum tempo atrás).
Sua classe deve ter ficado assim:

Conectando os itens



A classe DNSSolverController com os outlets e a ação criada

Para dizer quem é o tal de urlOutlet no seu programa, você deve fazer algumas conexões. Para isso, pressione a tecla Control e arraste até o primeiro campo de texto. Assim que ele selecionar o campo, solte o mouse e faça a conexão na nova janela que surge com o item 'urlOutlet' que aparecerá nessa nova janela. Faça o mesmo para o segundo campo de texto, agora com o ipOutlet.
Para conectar a ação, a idéia é inversa. Use Control e arraste de quem chama a ação (botão Solve DNS) para o objeto (DNSSolverController). Confirme a conexão na janela que surge, desta vez com o método resolverDNS().
Agora, resta escrever o código. Salve tudo e crie os arquivos com o código fonte. O Interface Builder faz metade do trabalho para você. Vá até a aba Classes e selecione o objeto DNSSolverController. No menu Classes selecione Create Files. Na janela que surge, peça para inserir no seu projeto (deixe o checkbox marcado). Pressione o botão Choose assim que você selecionar a pasta que contém seu projeto. Isso é importante. Os arquivos com código fonte DEVEM estar no mesmo nível do file system onde está o seu projeto (o arquivo que termina com .prbx).
Vá para o Project Builder a analise o arquivo criado: DNSSolverController.java.
Ele deve ser mais ou menos assim:

```
/* DNSSolverController */

import com.apple.cocoa.foundation.*;
import com.apple.cocoa.appkit.*;

public class DNSSolverController
{
        Object              ipOutlet;
        Object              urlOutlet;

        public void resolverDNS(Object sender)
        {

        }
}
```

Editando o código

Faça as seguintes alterações (linhas em vermelho são comentários para você entender o que está fazendo e podem ser omitidas): ▸

Construindo e rodando o aplicativo

Basta clicar no botão com um martelo no Project Builder e aguardar a construção do seu programa. No caso de sucesso, a mensagem "Build succeeded" aparecerá no rodapé esquerdo da janela do projeto. No caso de um erro, você deve analisar se cometeu algum erro ao digitar, se esqueceu de alguma vírgula etc.
Para rodar o programa,basta clicar no terceiro botão da janela do projeto do Project Builder, que é o ícone de um monitor com algumas janelas abertas.
Rode o aplicativo e teste, por exemplo, com apple.com, www.macmania.com.br etc. Perceba que você já tem suporte a múltiplos documentos com o template de projeto que escolheu no início. Isso é fornecido através do arquivo MyDocument.java, que infelizmente não temos tempo de comentar aqui.
Mais uma referência obrigatória:
Mac OS X: System Overview
http://developer.apple.com/techpubs/macosx/SystemOverview/SystemOverview.pdf

Considerações finais

A idéia deste primeiro tutorial foi dar uma visão geral do desenvolvimento com o Project Builder e o Interface Builder, suas facilidades, e entender o que está envolvido no desenvolvimento no Mac OS X. Nas próximas edições espero me ater mais à API que às ferramentas propriamente ditas.
Aproveitamos para convidá-lo para a lista gratuita de discussões de desenvolvedores da Apple (drc@apple.com.br). Ela é a melhor forma dos desenvolvedores brasileiros obterem informações e fazerem perguntas. Até lá! M

FABIO G. RIBEIRO ("The Caffeine!")
fabio.g@apple.com.br

```
#import com.apple.cocoa.foundation.*;
#import com.apple.cocoa.application.*;
#import java.net.*;
public class DNSSolver {
    NSTextField ip;
    NSTextField url;
    public void resolverDNS(NSObject sender)
{
        InetAddress ipString;
        String addressString="";
        String enderecoResolvido="";
        NSWindow minhaJanela;
        addressString = url.stringValue();
        try
        {
            ipString =
InetAddress.getByName(addressString);
            enderecoResolvido =
ipString.toString(  );

ip.setStringValue(enderecoResolvido);
            minhaJanela = ip.window();

minhaJanela.setTitle(addressString);
        }
        catch (UnknownHostException e)
        {
            String errorString = "Não foi
possível resolver o host " + addressString +
".  ";

NSAlertPanel.runAlert(errorString, "Verifique
se você digitou a URL do Host corretamente ou
se a sua rede está de pé e tente novamente.",
"OK", null, null);
        }
    }

}
```

Importa a foundation kit

Importa a application kit

Importa java.net.

Classe principal do aplicativo que é responsavel por resolver o endereço

Outlet para o NSTextField 'IP' e 'URL Address:'

InetAddress que representa o endereçamento para Java

Endereço que o usuário digitou

Endereço resolvido

Uma referência para janela em questão

Obtenho o endereço que o usuário digitou

Tento resolver o endereço

Começo a tentativa de resolver o endereço de 'algumaempresa.com.br' para 'algumaempresa.com.br/000.000.000.000'

Converto um 'InetAddress' para 'algumaempresa.com.br/000.000.000.000'

Coloco no campo 'IP' o endereço resolvido

Obtenho uma referência para a janela na qual o campo 'ip' está. Esta é a forma mais simples de obter referências de uma janela, tendo a referência para um dos seus elementos. Neste caso 'ip' é um outlet para um NSTextField

Com a referência para a janela, altero seu título para o nome da url, ex: "apple.com"

Mostro uma exceção no caso de algum problema ao resolver o host

Monto a mensagem de erro

Mostro a mensagem de erro