Programando o Android

Zigurd Mednieks, Laird Dornin, G. Blake Meike
e Masumi Nakamura

Novatec

Editor: Rubens Prates
Tradução: Rafael Zanolli
Revisão técnica: Edgard Damiani
Revisão gramatical: Marta Almeida de Sá
Editoração eletrônica: Carolina Kuwabata

ISBN: 978-85-7522-284-3

Histórico de impressões:

Abril/2012 Primeira edição

Novatec Editora Ltda.
Rua Luís Antônio dos Santos 110
02460-000 – São Paulo, SP – Brasil
Tel.: +55 11 2959-6529
Fax: +55 11 2950-8869
E-mail: novatec@novatec.com.br
Site: www.novatec.com.br
Twitter: twitter.com/novateceditora
Facebook: facebook.com/novatec
LinkedIn: linkedin.com/in/novatec
VC20120322

PARTE I

Ferramentas e conhecimentos básicos

A parte I deste livro mostra como instalar e utilizar suas ferramentas, além das informações necessárias sobre Java para que você possa escrever código funcional para o Android. O texto também fornece instruções sobre como projetar e utilizar bancos de dados SQL. Todos esses tópicos são elementos centrais não só do modelo de aplicativo Android que criaremos, mas também de seu sistema de persistência e da implementação de padrões de projeto essenciais em aplicativos Android.

CAPÍTULO 1

Seu kit de ferramentas

Este capítulo ensina a instalar o SDK (Software Development Kit, ou Kit de Desenvolvimento de Software) do Android e todos os softwares relacionados que possam vir a ser necessários. Ao final, você será capaz de executar um simples programa "Hello world" ("Olá mundo") em um emulador. Tanto sistemas Windows quanto Mac OS X e Linux podem ser utilizados no desenvolvimento de aplicativos para o Android. Neste capítulo, carregaremos o software, apresentaremos as ferramentas do SDK e indicaremos fontes de códigos de exemplo.

No decorrer deste livro, e especialmente neste capítulo, faremos referência às instruções disponíveis em muitos sites que explicam como instalar e atualizar as ferramentas utilizadas na criação de aplicativos Android. O local mais importante no qual você pode encontrar informações e links para ferramentas é o site de desenvolvedores do Android: *http://developer.android.com*.

Nosso foco é guiá-lo pelo processo de instalação, com explicações que deverão ajudá-lo a compreender como se encaixam os elementos do Android e de suas ferramentas de desenvolvimento, e até mesmo como os detalhes de cada elemento podem mudar.

Os links citados neste livro podem sofrer alterações com o tempo. Descrições e links atualizados podem ser encontrados no site do livro. Você pode encontrar um link para o site na página de catálogo deste livro (*http://oreilly.com/catalog/0636920010364*). Talvez seja interessante manter essa página aberta enquanto você lê o material, para que você possa consultar os links sem ter de digitar os URLs impressos no livro.

Instalação do SDK e dos prerrequisitos do Android

Para instalar o SDK do Android serão necessários dois outros sistemas de software que não fazem parte do SDK: o JDK (Java Development Kit, ou Kit de Desenvolvimento Java) e o IDE (Integrated Development Environment, ou Ambiente de Desenvolvimento Integrado) do Eclipse. Esses dois sistemas não acompanham o SDK do Android porque podem já estar instalados em seu sistema, sendo utilizados por

você com outro propósito que não o desenvolvimento de software para o Android – instalações redundantes desses sistemas podem resultar em conflitos de versões.

O SDK do Android é compatível com muitas versões recentes do JDK e do IDE Eclipse. Instalar a versão mais atual de cada uma dessas ferramentas deverá ser suficiente. Os requisitos exatos estão especificados na página de requisitos do sistema (System Requirements) disponível no site de desenvolvedores do Android: *http://developer.android.com/sdk/requirements.html*.

É possível utilizar outros IDEs, que não sejam o Eclipse, no desenvolvimento de software para o Android. Informações a esse respeito podem ser encontradas na documentação referente, neste endereço: *http://developer.android.com/guide/developing/other-ide.html*. Neste livro, escolhemos o Eclipse porque ele oferece suporte ao maior número de ferramentas do SDK do Android e outros plug-ins, e pelo fato de ser o IDE Java mais utilizado. Ainda assim, o IntelliJ IDEA também é uma alternativa empregada por muitos programadores que trabalham com Java.

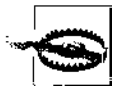
Java Development Kit (JDK)

Caso seu sistema tenha um Java Development Kit (JDK) atualizado instalado, você não terá de instalá-lo novamente. O JDK oferece ferramentas como o compilador Java, utilizado por IDEs e SDKs para desenvolvimento de programas em Java. O JDK também contém um Java Runtime Environment (JRE), que permite a execução de programas Java, como o Eclipse, em seu sistema.

Caso você esteja utilizando um Macintosh e uma versão do Mac OS X aceita pelo SDK do Android, o JDK já deve estar instalado.

Se você estiver utilizando um sistema operacional Linux Ubuntu, pode instalar o JDK por meio do gerenciador de pacotes, com o seguinte comando:

```
sudo apt-get install sun-java6-jdk
```



Caso esse comando informe que o pacote JDK não está disponível, talvez você tenha de habilitar os repositórios "partner" utilizando o utilitário Synaptic Package Manager no menu **System > Administration** (Sistema > Administração). Os repositórios "partner" estão listados na aba Other Software (Outros softwares), depois de **Settings > Repositories** (Configurações > Repositórios).

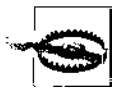
Esse é um dos poucos locais neste capítulo em que mencionaremos um número de versão. Nesse caso, isso é inevitável: o número de versão do JDK faz parte do nome do pacote. Mas, assim como todos os softwares mencionados neste capítulo, você deve consultar a documentação online atualizada para determinar qual versão é necessária.

Se você é um usuário do Windows ou se tem de instalar o JDK a partir do site da Oracle, pode encontrá-lo neste endereço: *http://www.oracle.com/technetwork/java/javase/downloads/index.html*.

A página de download detectará automaticamente seu sistema e oferecerá o download da versão correta. O instalador do qual você fará o download é um arquivo executável; basta executá-lo para instalar o JDK.

Para confirmar se o JDK foi instalado corretamente, utilize o comando a seguir na linha de comando (terminal no Linux e no Mac; prompt de comando no Windows):

```
javac -version
```



Se o comando `javac` não estiver em seu `PATH`, talvez seja necessário adicionar manualmente o diretório *bin* do JDK ao seu caminho.

Esse comando deve mostrar o número de versão correspondente à versão do JDK instalada. Se você instalou a revisão 20 do JDK do Java 6, o comando deve mostrar:

```
javac 1.6.0_20
```

Quando você estiver lendo este texto, dependendo da versão atual do JDK disponível, os números de versão podem ser diferentes dos que mostramos aqui.

Caso não esteja claro qual JRE você está utilizando, ou se você acredita que esteja executando o JRE errado em um sistema Linux, utilize o comando a seguir para mostrar os JREs disponíveis e selecionar o mais adequado:

```
sudo update-alternatives --config java
```

Integrated Development Environment (IDE) do Eclipse

O Eclipse é uma plataforma de tecnologia de propósito geral que, além de ser aplicada na criação de IDEs para diferentes linguagens, também pode ser utilizada na criação de IDEs personalizados para muitos SDKs especializados. Sua utilização também pode se estender ao trabalho com aplicativos que vão além das ferramentas de desenvolvimento de software, como o fornecimento de uma plataforma de cliente rica (Rich Client Plataform, ou RCP) para o Lotus Notes e outros casos.

O Eclipse é geralmente utilizado como um ambiente integrado de desenvolvimento (IDE) capaz de escrever, testar e depurar software, especialmente software Java. Há também muitos IDEs e SDKs derivados do Eclipse para diversos tipos de softwares Java. Em nosso caso, você pegará um pacote do Eclipse amplamente utilizado e adicionará um plug-in para usá-lo no desenvolvimento de softwares Android. Vamos, então, obter o pacote Eclipse e instalá-lo.

O download do Eclipse pode ser feito nesta página: *http://www.eclipse.org/downloads/*.

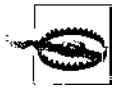
Você verá uma lista dos pacotes mais utilizados para o Eclipse. Um "pacote" do Eclipse é uma coleção pronta de módulos que o tornam mais capacitado para certos tipos de desenvolvimento de software. Geralmente, usuários do Eclipse iniciam suas atividades com um dos pacotes disponíveis para download nesse endereço e, depois, personalizam

sua opção com diversos plug-ins. É exatamente isso que você fará, ao adicionar o plug-in ADT à sua instalação do Eclipse. O artigo sobre requisitos do sistema no site de desenvolvedores do Android lista três opções de pacotes do Eclipse, como base para instalação e uso do IDE no desenvolvimento de softwares para o Android:

- Eclipse Classic (para o Eclipse 3.5 ou mais recente)
- Eclipse IDE for Java Developers
- Eclipse for RCP/Plug-in Developers

Qualquer uma dessas opções será suficiente, ainda que, a menos que você também esteja desenvolvendo plug-ins para o Eclipse, escolher o pacote Classic ou Java Developers (EE ou padrão) faça mais sentido. Os autores deste livro iniciaram utilizando o pacote Java EE Developers e as capturas de tela que utilizamos refletem essa escolha.

O site de download do Eclipse determinará automaticamente para você os downloads específicos disponíveis de acordo com seu sistema, mas você terá de escolher entre versões de 32 e 64 bits, de acordo com seu caso. Depois, você fará o download de um arquivo compactado. Para instalar o Eclipse, abra esse arquivo e copie a pasta *eclipse* para sua pasta base. O arquivo executável para iniciar o Eclipse em seu sistema pode ser encontrado na pasta que você acabou de extrair do arquivo.



Estamos falando sério quando dizemos que você deve instalar o Eclipse em sua pasta home (ou em alguma outra pasta à qual você tenha acesso), especialmente se você tem diversas contas de usuário em seu sistema. Lembre-se de que sua instalação do Eclipse será apenas um elemento dentre os muitos agrupamentos possíveis de plug-ins, e que você seguirá personalizando-a com plug-ins dedicados ao desenvolvimento de software para o Android. Além disso, atualizações do Eclipse são gerenciadas separadamente de outros softwares em seu sistema.

Por essas e por muitas outras razões, é muito difícil instalar e utilizar satisfatoriamente o Eclipse como um comando disponível para todos os usuários em seu sistema. Para completar adequadamente uma instalação da forma que descrevemos, você deve instalar o Eclipse em seu diretório home e inicializá-lo a partir desse endereço.

Se você está utilizando o Ubuntu ou outra distribuição Linux, não deve instalar o Eclipse a partir dos repositórios de sua distribuição. Se ele já estiver instalado dessa forma, você deve removê-lo e instalá-lo da forma que mostramos. A presença de um pacote "eclipse" nos repositórios do Ubuntu é uma herança dos repositórios Debian (base para o Ubuntu) e não representa uma abordagem recomendada para instalação e uso do Eclipse, uma vez que, na maioria dos casos, os repositórios de sua distribuição utilizam versões mais antigas do Eclipse.

Para confirmar se o Eclipse está instalado corretamente e se você tem um ambiente de tempo de execução Java que o aceita, inicie o arquivo executável na pasta Eclipse. Pode ser interessante criar um atalho para esse arquivo, permitindo que você inicie o Eclipse de modo mais prático. Você deverá ver a tela de boas-vindas mostrada na figura 1.1.

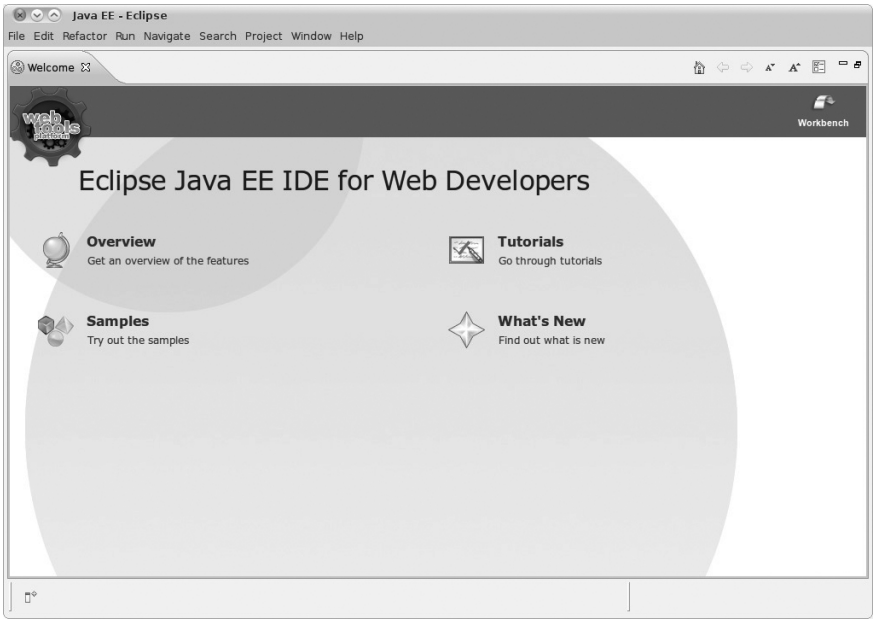


Figura 1.1 – Na primeira execução do Eclipse, você verá essa tela de boas-vindas.

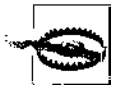
O Eclipse é implementado em Java e requer um ambiente de tempo de execução Java (*Java Runtime Environment*, ou JRE). O JDK que você instalou antes fornece um JRE. Se o Eclipse não puder ser executado, você deve verificar se o JDK foi instalado corretamente.

SDK do Android

Com o JDK e o Eclipse instalados, você agora tem os prerrequisitos para o SDK do Android e está pronto para instalá-lo. O SDK do Android é um conjunto de arquivos: bibliotecas, executáveis, scripts, documentação etc. Instalar o SDK significa fazer o download da versão do SDK para sua plataforma e colocar seus arquivos em uma pasta em seu diretório home.

Para instalar o SDK, faça o download do pacote que corresponde ao seu sistema nesta página: *http://developer.android.com/sdk/index.html*.

O download é um arquivo compactado. Abra-o e extraia a pasta contida nele para dentro de sua pasta home.



Se estiver utilizando uma versão de 64 bits do Linux, talvez seja necessário instalar o pacote `ia32-libs`. Para verificar se esse pacote é necessário, tente executar o comando `adb` (`~/android-sdk-linux_*/platform-tools/adb`). Caso seu sistema informe que o adb não pode ser encontrado (apesar de ele estar presente no diretório `platform-tools`), isso provavelmente significa que a versão atual do adb, e possivelmente de outras ferramentas, não poderá ser executada sem a instalação do pacote `ia32-libs`. O comando para instalar o pacote `ia32-libs` é o seguinte:

```
sudo apt-get install ia32-libs
```

O SDK contém uma ou duas pastas para ferramentas: uma chamada *tools* e, a partir da versão 8, outra chamada *platform-tools*. Essas pastas devem estar em seu path (uma lista de pastas nas quais seu sistema pesquisa arquivos executáveis quando você invoca um executável a partir da linha de comando). Em sistemas Macintosh e Linux, a definição da variável de ambiente PATH é feita no arquivo *.profile* (Ubuntu) ou *.bash_profile* (Mac OS X) em seu diretório home. Adicione uma linha a esse arquivo que instrua a variável de ambiente PATH a incluir o diretório *tools* do SDK (entradas individuais devem ser separadas por dois-pontos). Por exemplo, você poderia utilizar a linha a seguir (substituindo ambas as instâncias de *~/android-sdk-ARCH* pelo caminho completo de sua instalação do SDK do Android):

```
export PATH=$PATH:~/android-sdk-ARCH/tools:~/android-sdk-ARCH/platform-tools
```

Em sistemas Windows, clique em **Iniciar** e, depois, com o botão direito em **Meu Computador**, escolha **Propriedades**. Depois, clique em **Avançado** e clique no botão **Variáveis de ambiente**. Clique duas vezes na variável de sistema PATH e adicione o caminho das pastas. Faça isso no final do valor dessa variável (não altere nenhuma informação já presente!) e adicione os dois caminhos ao final da linha, separados por ponto e vírgula, sem espaços em branco antes ou depois deles. Por exemplo:

```
;C:\android-sdk-windows\tools;C:\android-sdk-windows\platform-tools
```

Depois de editar seu path no Windows, Mac ou Linux, feche e abra novamente os prompts de comando ou terminais abertos para que sejam reconhecidas as novas configurações da variável PATH (no Ubuntu, talvez você tenha de efetuar novamente seu login, a menos que seu programa de terminal esteja configurado como um shell de login).

Inclusão de alvos de compilação ao SDK

Antes que você possa criar um aplicativo Android, ou mesmo um projeto que tente criar um aplicativo desse tipo, você deve instalar um ou mais alvos de compilação (build targets). Para tanto, você utilizará o gerenciador de AVD e SDK do Android (**AVD and SDK Manager**). Essa ferramenta permite que você instale pacotes no SDK que oferecem suporte a diversas versões do sistema operacional Android e a vários níveis de API.

Assim que o plug-in ADT estiver instalado no Eclipse (processo que descreveremos na próxima seção), o gerenciador de AVD e SDK poderá ser invocado a partir do Eclipse. Ele também poderá ser invocado a partir da linha de comando, modo que utilizaremos aqui. Para tanto, utilize o seguinte comando:

```
android
```

A imagem da figura 1.2 mostra o gerenciador do AVD e do SDK, com todas as versões disponíveis do SDK selecionadas para instalação.

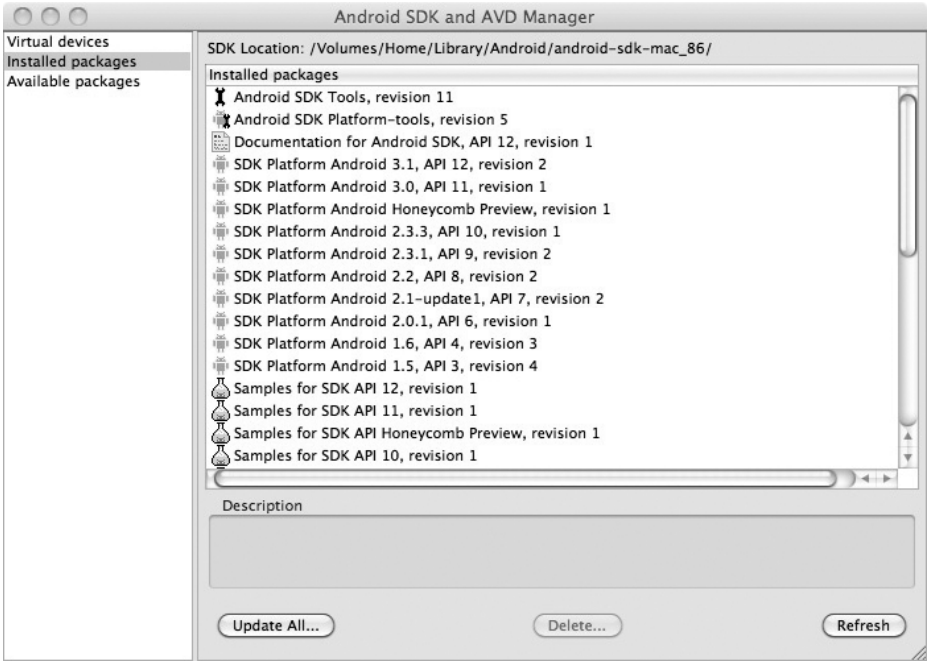


Figura 1.2 – Gerenciador de SDK e AVD que permite a instalação de níveis da API do Android.

Os pacotes intitulados "SDK platform" oferecem suporte à criação de aplicativos compatíveis com diferentes níveis de API do Android. Você deve instalar ao menos a versão mais recente (de número mais alto), mas também é uma boa opção instalar todos os níveis de API disponíveis, assim como todos os pacotes de add-ons da API do Google, caso algum dia você queira criar aplicativos que possam ser executados em versões mais antigas do Android. Você também deve instalar as versões mais recentes do pacote de aplicativos de exemplo. Também é necessário que você instale o pacote SDK Platform-Tools do Android.

Android Development Toolkit (ADT): plug-in para Eclipse

Agora que você instalou os arquivos do SDK, acompanhados do Eclipse e do JDK, há mais um componente crítico a ser instalado: o plug-in Android Developer Toolkit (ADT), que adiciona funcionalidades específicas do Android ao Eclipse.

O plug-in permite que o Eclipse compile aplicativos para o Android, inicie o emulador Android e se conecte aos seus serviços de depuração, além de permitir a edição de arquivos XML do Android, a edição e compilação de arquivos na Linguagem de Definição de Interface do Android (Android Interface Definition Language, ou AIDL), a criação de pacotes de aplicativos para o Android (arquivos *.apk*) e a realização de outras tarefas específicas.

Utilização do assistente Install New Software para download e instalação do plug-in ADT

Você inicia o assistente Install New Software (Instalar Novo Software) selecionando o menu Help > Install New Software (Figura 1.3). Para instalar o plug-in ADT, digite o URL a seguir, no campo Work With (Trabalhar com), e pressione Enter: *https://dl-ssl.google.com/android/eclipse/* (Figura 1.4).

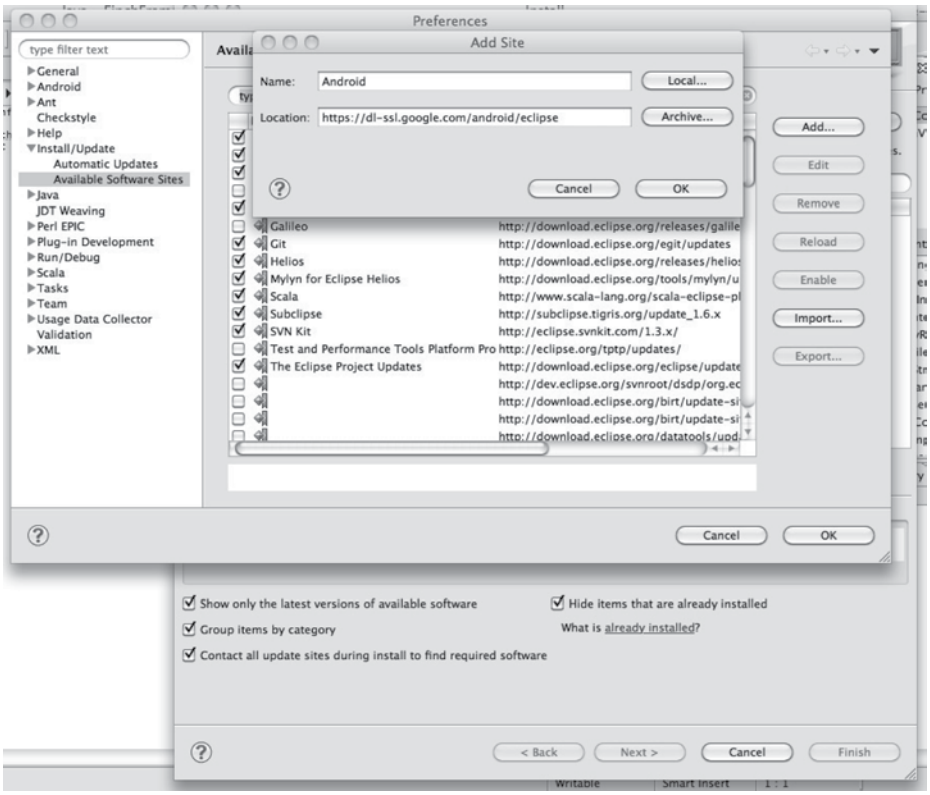


Figura 1.3 – Diálogo "Add Site" do Eclipse.

Mais informações sobre a instalação do plug-in ADT por meio do assistente "Install New Software" podem ser encontradas no site de desenvolvedores do Android, no endereço: *http://developer.android.com/sdk/eclipse-adt.html#downloading*. A documentação do Eclipse acerca desse assistente pode ser encontrada no site de documentação do próprio Eclipse: *http://help.eclipse.org/galileo/index.jsp?topic=/org.eclipse.platform.doc.user/tasks/tasks-124.htm*.

Assim que tiver adicionado o URL à lista de sites para aquisição de novos plug-ins, você verá uma entrada Developer Tools (Ferramentas do Desenvolvedor) na lista de softwares disponíveis.

Selecione o item Developer Tools, clicando na caixa de seleção ao seu lado, e clique no botão Next. A tela seguinte pedirá que você aceite a licença para esse software. Depois

de fazê-lo, clique em **Finish** e o ADT será instalado. Você terá de reiniciar o Eclipse para completar a instalação.

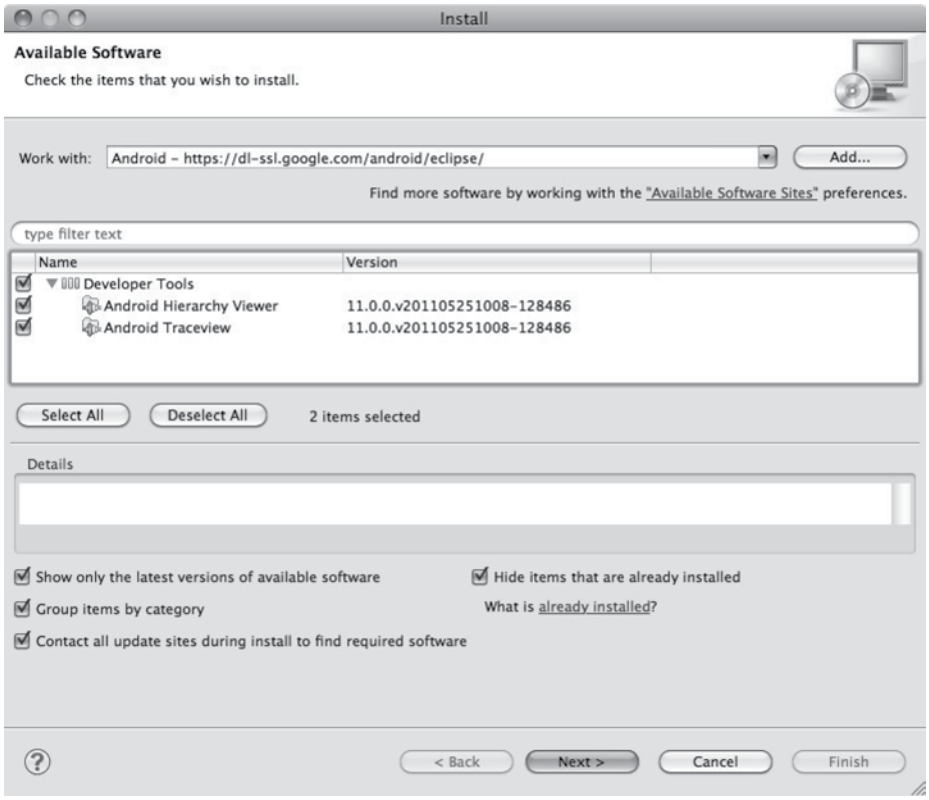


Figura 1.4 – Diálogo "Install New Software" do Eclipse mostrando o plug-in ADT como disponível.

Configuração do plug-in ADT

Mais um passo e concluiremos a instalação. Assim que você tiver instalado o plug-in ADT, terá de configurá-lo. Com esse plug-in instalado, diversas partes do Eclipse agora contêm caixas de diálogo, comandos de menu e outras ferramentas específicas ao desenvolvimento de softwares para o Android, incluindo a caixa de diálogo que você utilizará para configurar o plug-in ADT: acesse a caixa de diálogo de preferências em **Window > Preferences** (em sistemas Linux e Windows) ou a opção de menu **Eclipse > Preferences** (em sistemas Mac). Clique no item intitulado **Android** no painel esquerdo da caixa de diálogo de preferências.

Em sua primeira visita a essa seção, você será perguntado se deseja enviar estatísticas de utilização ao Google. Faça sua escolha e clique em **Proceed**.

Em seguida, um diálogo com as configurações do Android será mostrado. Nele, um campo de entrada de texto, **SDK location** (localização do SDK), surgirá próximo ao topo. Você deve digitar o caminho para o local em que colocou o SDK, ou pode navegar até a localização para selecionar o diretório (Figura 1.5). Clique em **Apply**. Note que os alvos de compilação que você instalou, como descrito antes neste capítulo, também estão listados aqui.

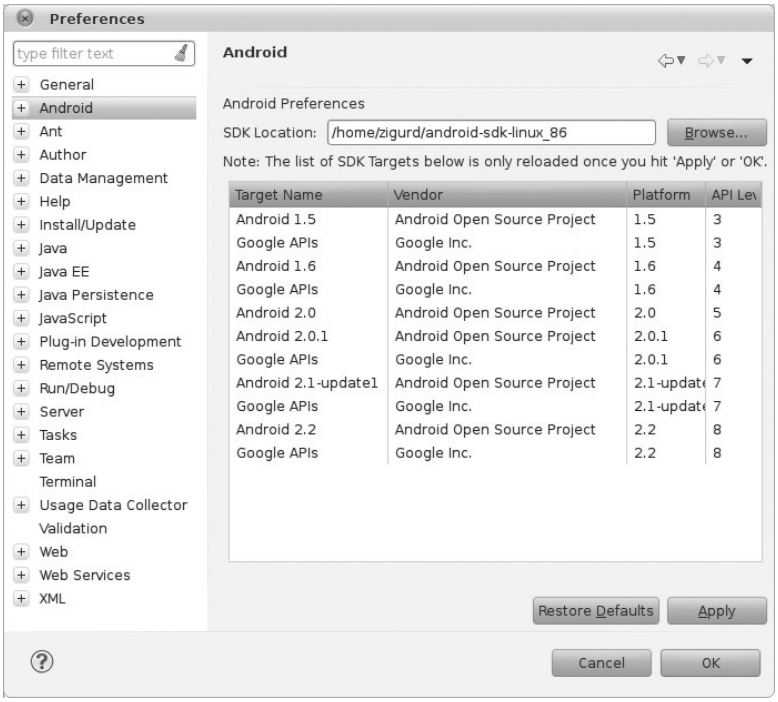


Figura 1.5 – Configuração da localização do SDK no plug-in ADT do Eclipse utilizando o diálogo de preferências do Android.

Com isso, sua instalação do SDK do Android está completa.

Test drive: confirme que sua instalação funciona

Caso você tenha seguido os passos mostrados neste capítulo e as instruções online a que fizemos referência, sua instalação do SDK do Android agora estará completa. Para confirmar se tudo está funcionando como pretendemos, vamos criar um simples aplicativo Android.

Criação de um projeto Android

O primeiro passo para desenvolver um aplicativo básico para o Android é criar um projeto Android. O Eclipse organiza seu trabalho em "projetos". Designando seu

projeto como um projeto Android, você diz ao Eclipse que o plug-in ADT e outras ferramentas do Android deverão ser utilizados em conjunto neste projeto.

Informações de referência e instruções online detalhadas para criação de um projeto Android podem ser encontradas neste endereço: *http://developer.android.com/guide/developing/eclipse-adt.html*.

Inicie seu novo projeto com o comando de menu File > New > Android Project. Localize a opção Android Project no diálogo New Project (ela deve estar sob uma seção intitulada Android). Clique em Next e a caixa de diálogo New Project surgirá (Figura 1.6).

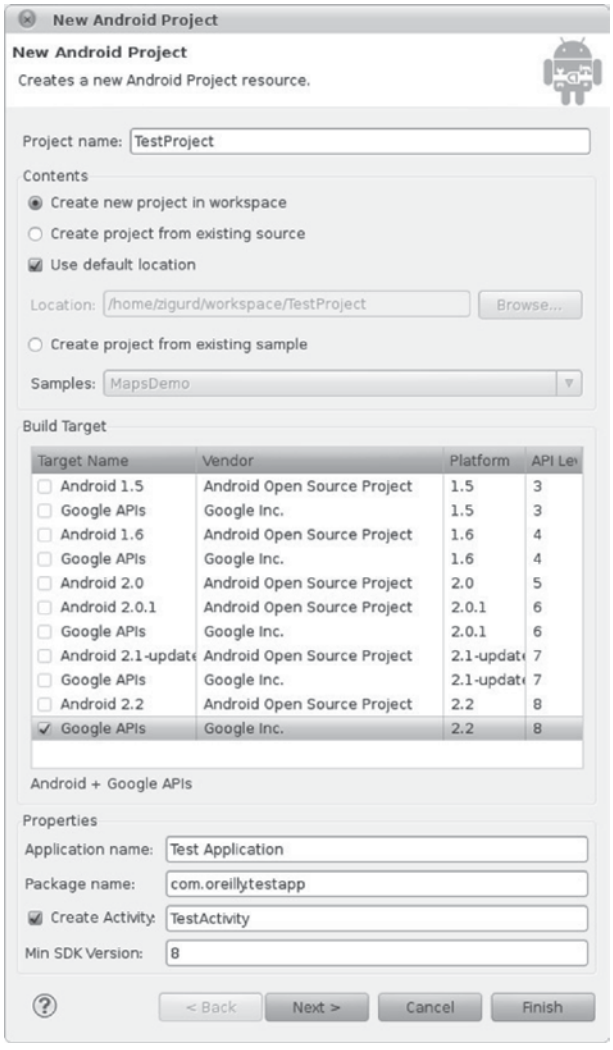


Figura 1.6 – Caixa de diálogo New Android Project.

Para criar seu projeto Android, você fornecerá as seguintes informações:

Project Name (Nome do projeto)

Esse é o nome do projeto (não do aplicativo) que surge no Eclipse. Digite `TestProject`, como na figura 1.6.

Workspace (Espaço de trabalho)

Um *workspace*, ou espaço de trabalho, é uma pasta que contém um conjunto de projetos do Eclipse; ao criar um novo projeto, você tem a escolha de fazê-lo em seu espaço de trabalho atual, ou de especificar uma localização diferente para seu projeto no sistema de arquivos. A menos que você tenha de colocar este projeto em uma localização específica, utilize as opções padrão (Create New Project in Workspace e Use Default Location).

Target (Alvo)

As imagens do sistema Android que você instalou no SDK surgem na lista de alvos de compilação. Você pode escolher uma delas, além do fornecedor, da plataforma (número de versão do sistema operacional Android) e do nível de API correspondentes ao seu caso, identificando a versão para a qual seu aplicativo foi construído. A plataforma e o nível de API representam os parâmetros de maior importância: são eles que governam a biblioteca da plataforma Android com a qual seu aplicativo será compilado e o nível de API aceito – APIs com um nível mais alto do que o selecionado não estarão disponíveis para o seu programa. Por ora, escolha a versão mais recente do sistema operacional Android e o nível de API que você instalou.

Application name (Nome do aplicativo)

Esse é o nome do aplicativo que será visto pelo usuário. Digite `Test Application` (Aplicativo de teste).

Package name (Nome do pacote)

O nome do pacote cria um namespace (espaço de nomes) para um pacote Java que identifica individualmente pacotes em seu aplicativo e que também deve identificar individualmente seu aplicativo Android dentre todos os outros aplicativos instalados. O nome do pacote consiste em um nome de domínio exclusivo – o nome de domínio de quem publicou o aplicativo – mais um nome específico para o aplicativo. Nem todos os namespaces de pacotes são exclusivos no Java, mas as convenções utilizadas diminuem a probabilidade de conflitos. Em nosso exemplo, utilizamos "*com.oreilly.testapp*", mas você pode escolher algo mais apropriado ao seu domínio (ou utilizar *com.example.testapp*, uma vez que *example.com* é um nome de domínio reservado a exemplos como esse).

Activity (Atividade)

Uma *atividade* é uma unidade de interface de usuário interativa de um aplicativo Android, que geralmente corresponde a um grupo de objetos de interface de usuário que ocupam a tela inteira. Opcionalmente, quando você cria um projeto, pode escolher a criação do esqueleto de uma atividade. Caso você esteja criando um aplicativo visual (e não um serviço, que pode ser destituído de representação visual na interface do usuário), essa é uma forma conveniente de criar a atividade com a qual o aplicativo iniciará. Nesse exemplo, você deve criar uma atividade com o nome `TestActivity`.

Minimum SDK Version (Versão mínima do SDK)

O campo **Min SDK Version** deve conter um número inteiro, que corresponde à versão mínima do SDK necessária para que seu aplicativo funcione, e é utilizado para inicializar o atributo "`uses-sdk`" no manifesto do aplicativo (arquivo que armazena os atributos do aplicativo). Consulte a seção "Android Manifest Editor", mais adiante neste capítulo, para mais informações. Na maioria dos casos, essa opção deve corresponder ao nível de API do alvo de compilação que você escolheu, mostrado na coluna mais à direita da lista de alvos de compilação (Figura 1.6).

Clique em **Finish** (e não em **Next**) para criar seu projeto Android e vê-lo listado no painel esquerdo do Eclipse (Figura 1.7).

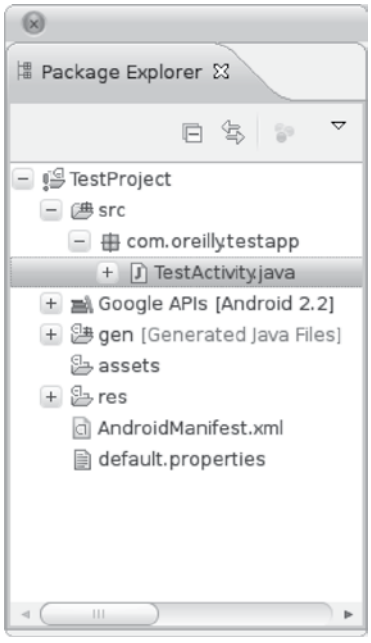


Figura 1.7 – Visualização do Package Explorer, mostrando os arquivos e componentes que fazem parte do projeto.

Caso você expanda a visualização da hierarquia de seu projeto, clicando no sinal + (no Windows) ou no triângulo (Mac e Linux) ao lado do nome dele, você verá as várias partes de um projeto Android. Expanda a pasta *src* e você verá um pacote Java com o nome que foi digitado no assistente. Expanda esse pacote e você encontrará a classe `Activity` que o assistente criou para você. Clique duas vezes nela e você verá o código Java de seu primeiro programa Android.

```
package com.oreilly.demo.pa.ch01.testapp;

import android.app.Activity;
import android.os.Bundle;
import com.oreilly.demo.pa.ch01.R;

public class TestActivity extends Activity {
    /** Chamado quando a atividade é criada */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

Se você vem acompanhando os passos que apresentamos e encontrou o mesmo resultado em seu computador, então sua instalação do SDK provavelmente está funcionando corretamente. Para nos certificarmos disso, vamos explorar mais um pouco o SDK, executando seu primeiro programa em um emulador e também em um dispositivo Android, caso você tenha um em mãos.

Criação de um Android Virtual Device (AVD)

O SDK do Android oferece um emulador, capaz de emular um dispositivo com uma CPU ARM executando um sistema operacional Android, para teste de programas em seu PC. Um dispositivo virtual Android (Android Virtual Device, ou AVD) é um conjunto de parâmetros que servem para configurar o emulador de modo a utilizar uma imagem específica de sistema – uma determinada versão do sistema operacional Android – e definir outros parâmetros que governam o tamanho da tela, o tamanho da memória e outras características do hardware emulado. Para saber mais, a documentação detalhada sobre AVDs pode ser encontrada neste endereço: *http://developer.android.com/guide/developing/tools/avd.html*. Para mais informações sobre o emulador, consulte o seguinte endereço: *http://developer.android.com/guide/developing/tools/emulator.html*.

Ao testarmos sua instalação do SDK, não abordaremos minuciosamente os AVDs, muito menos os detalhes do emulador, pelo menos não por enquanto. Aqui, utilizaremos o gerenciador de SDK e AVD do Android (Figura 1.8) para preparar um AVD,

com o intuito de executar o programa que acabamos de criar com o assistente New Android Project.

Figura 1.8 – Gerenciador de SDK e AVD.

Você terá de criar um AVD com uma imagem de sistema que não seja mais recente do que a versão especificada para o projeto que você criou. Primeiro, pressione o botão New.... Você verá a caixa de diálogo Create New Android Virtual Device (AVD), na qual você especifica os parâmetros de seu novo AVD (Figura 1.9):

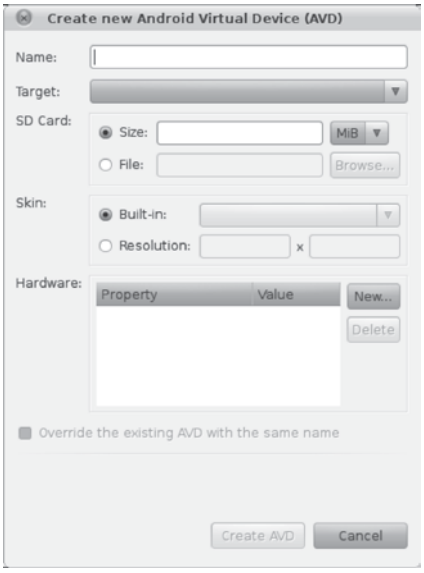


Figura 1.9 – Criação de um novo AVD.

Essa tela permite que você defina os parâmetros para seu novo AVD:

Name (Nome)

O nome de seu AVD. Você pode utilizar qualquer nome para um AVD, mas prefira um nome que indique a imagem de sistema utilizada.

Target (Alvo)

O parâmetro "Target" define qual imagem de sistema será utilizada nesse AVD. Ela deve ser a mesma, ou mais recente, que a versão selecionada como alvo de compilação em seu primeiro projeto para o Android.

SD Card (Cartão SD)

Alguns aplicativos requerem um cartão SD, capaz de estender a capacidade de armazenamento para além da memória flash presente em um dispositivo Android. A menos que você esteja planejando armazenar muitos dados em um cartão SD (arquivos de mídia, por exemplo), você pode criar um pequeno cartão SD virtual de, digamos, 100 MB, para os aplicativos que está desenvolvendo, mesmo que a maioria dos celulares esteja equipada com cartões SD capazes de armazenar muitos gigabytes.

Skin

O "skin" de um AVD define principalmente o tamanho da tela. Você não terá de alterar a configuração padrão para verificar se sua instalação funciona, mas é interessante emular diferentes tamanhos de tela para verificar se seus layouts funcionam em dispositivos de resoluções distintas.

Hardware

O campo Hardware da configuração do AVD permite que você defina parâmetros indicando qual hardware opcional está presente. Você não terá de alterar as configurações padrão para este projeto.

Preencha os campos Name, Target e SD Card e crie um novo AVD pressionando o botão Create AVD. Caso você não tenha criado um AVD com uma imagem que corresponda a, ou seja mais recente do que a versão especificada em seu projeto para o Android, você não será capaz de executar o programa.

Execução de um programa em um AVD

Agora que você tem um projeto que constrói um aplicativo e um AVD com uma imagem de sistema compatível com o alvo de compilação e com os requisitos do nível de API do aplicativo, você pode executar seu aplicativo e confirmar que o SDK produziu e é capaz de executar um aplicativo Android.

Para executar seu aplicativo, clique com o botão direito no projeto que você criou e, no menu de contexto, selecione Run As... > Android Application.

Se o AVD que você criou for compatível com o aplicativo criado, ele inicializará o sistema operacional Android, iniciando seu aplicativo. Você deverá ver seu aplicativo sendo executado no AVD de forma parecida com o que temos na figura 1.10.

Figura 1.10 – Aplicativo que você acabou de criar, executado em um AVD.

Caso você tenha mais de um AVD compatível configurado, a caixa de diálogo Android Device Chooser surgirá, pedindo que você selecione entre os AVDs ou entre os dispositivos Android conectados ao seu sistema (se houver), ou ainda que escolha um AVD para iniciar. A figura 1.11 exibe o Android Device Chooser, mostrando um AVD que está sendo executado e outro que pode ser iniciado.

Execução de um programa em um dispositivo Android

Você também pode executar o programa que acabou de criar na maioria dos dispositivos Android.

Será necessário que você conecte seu dispositivo ao PC utilizando um cabo USB e, se necessário, que instale um driver ou defina permissões para acessar o dispositivo conectado dessa forma.

Instruções específicas para Windows, além do driver necessário, estão disponíveis neste endereço: *http://developer.android.com/sdk/win-usb.html*.

Caso você esteja utilizando um sistema Linux, terá de criar um arquivo "rules" para seu dispositivo Android.

Caso você esteja utilizando um Mac OS X, nenhuma configuração será necessária.

Informações detalhadas de referência sobre depuração USB podem ser encontradas neste endereço: *http://developer.android.com/guide/developing/device.html*.

Você também terá de habilitar o recurso de depuração USB em seu dispositivo Android. Na maioria dos casos, você iniciará o aplicativo **Settings**, selecionará **Applications**, depois **Development** e então verá uma opção para habilitar ou desabilitar a depuração USB.

Se um AVD estiver configurado ou sendo executado, o Android Device Chooser surgirá, mostrando tanto o dispositivo Android conectado quanto o AVD.

Selecione o dispositivo e o aplicativo Android será carregado e executado nele.

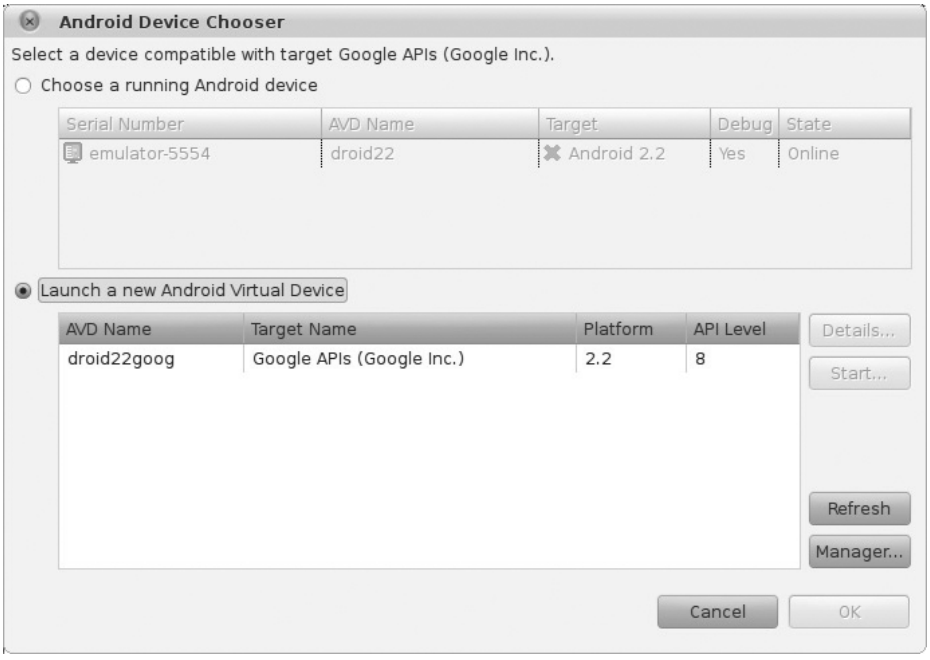


Figura 1.11 – Android Device Chooser.

Solução de problemas do SDK: alvos de compilação faltando

Se você foi incapaz de criar um novo projeto ou de importar um projeto de exemplo do SDK, pode ter se esquecido de instalar alvos de compilação em seu SDK. Leia novamente as instruções da seção "Inclusão de alvos de compilação ao SDK" e certifique-se de que a seção Android, na caixa de diálogo de preferências, lista os alvos de compilação instalados em seu SDK (Figura 1.5).

Componentes do SDK

O SDK do Android é composto principalmente de componentes prontos para uso, somados a outros, de propósitos específicos. Em muitos casos, configurações, plug-ins e extensões adaptam esses componentes ao Android. O SDK do Android é um exemplo perfeito do desenvolvimento eficiente de um SDK moderno e completo. O Google adotou essa abordagem para que pudesse disponibilizar rapidamente o Android no mercado. Você experimentará isso em primeira mão à medida que explora seus componentes. Eclipse, a linguagem Java, QEMU e outras plataformas existentes, ferramentas e tecnologias compõem algumas das partes mais importantes do SDK do Android.

Na criação do programa que confirma se sua instalação do SDK ocorreu corretamente, você já utilizou muitos dos componentes do SDK. Aqui, identificaremos e descreveremos os componentes do SDK envolvidos na criação de seu programa, assim como outras partes do SDK que você ainda utilizará.

Android Debug Bridge (adb)

O adb é um programa que permite que você controle tanto o emulador quanto os dispositivos e executa um shell para comandos no ambiente de um emulador ou dispositivo. Ele é especialmente útil para instalar e desinstalar programas nessas situações. Documentação referente a ele pode ser encontrada neste endereço: *http://developer.android.com/guide/developing/tools/adb.html*.

Dalvik Debug Monitor Server (DDMS)

O Dalvik Debug Monitor Server (DDMS) é um diretor de tráfego entre a porta individual que o Eclipse (e outros depuradores Java) procuraria para se conectar a uma Java Virtual Machine (JVM) e as várias portas que existem para cada dispositivo Android ou dispositivo virtual, assim como para cada instância da VM Dalvik em cada dispositivo. O DDMS também fornece muitas funcionalidades que podem ser acessadas por meio de uma interface de usuário independente ou por meio de uma interface incorporada ao Eclipse, utilizando o plug-in ADT.

Ao invocar o DDMS a partir da linha de comando, você deverá ver algo como a janela da figura 1.12.

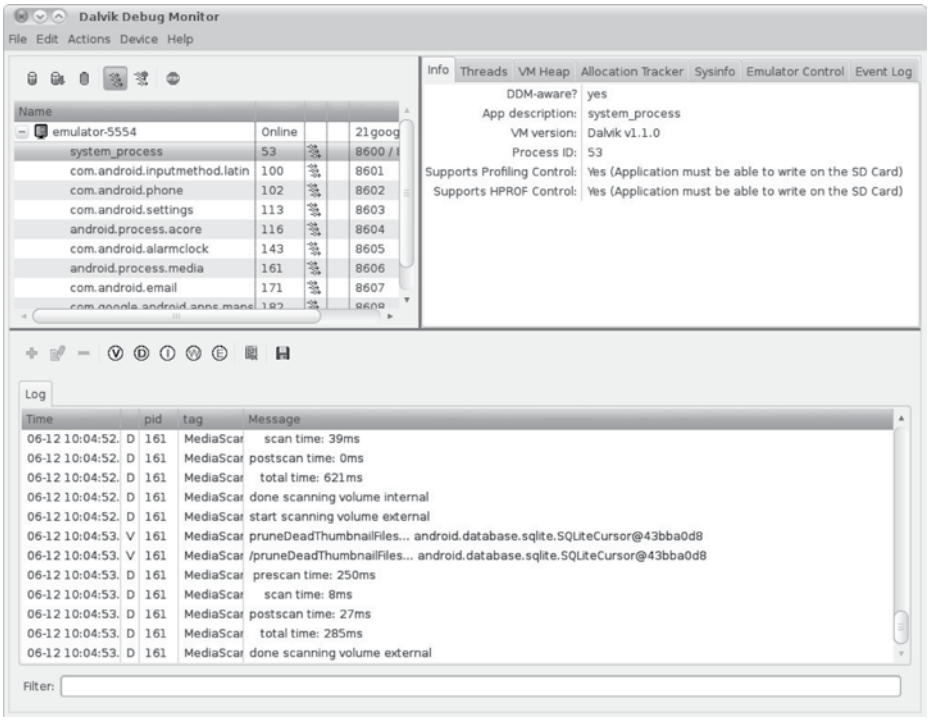


Figura 1.12 – Dalvik Debug Monitor, executado de forma independente.

A interface de usuário do DDMS fornece acesso a:

Uma lista de dispositivos e dispositivos virtuais, e às VMs executadas neles

No painel superior esquerdo da janela do DDMS, você verá os dispositivos Android conectados ao seu PC, além de dispositivos virtuais Android (AVDs) utilizados. Listadas sob cada dispositivo ou dispositivo virtual estão tarefas executadas nas máquinas virtuais (VMs) Dalvik.

Informações das VMs

Selecionando uma das VMs Dalvik executadas em um dispositivo ou dispositivo virtual, você tem acesso às informações referentes a essa VM no painel superior direito.

Informações de thread

Informações sobre as threads de cada processo podem ser acessadas na guia **Threads**, no painel superior direito da janela DDMS.

Gerenciador do sistema de arquivos

Você pode explorar o sistema de arquivos em um dispositivo ou dispositivo virtual utilizando o gerenciador de arquivos do DDMS, que pode ser acessado por meio do item de menu **File Explorer...**, no menu **Devices**. Ele mostra a hierarquia dos arquivos em uma janela como a da figura 1.13.

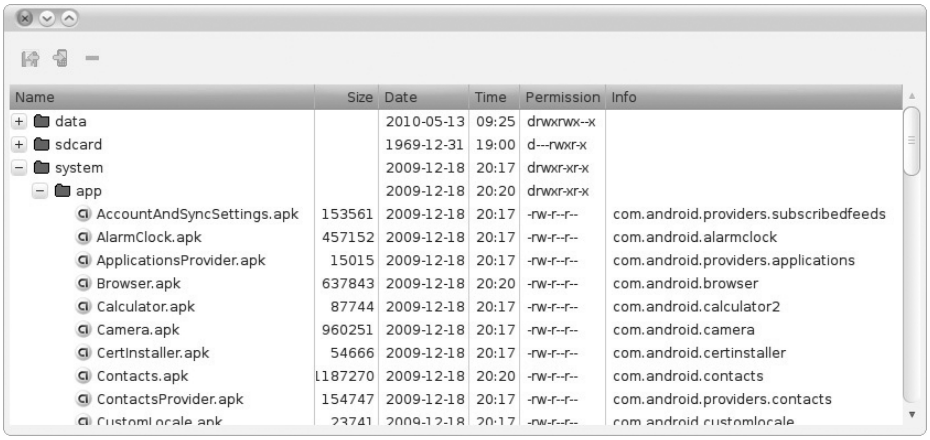


Figura 1.13 – Gerenciador do sistema de arquivos do DDMS.

Simulação de chamadas telefônicas

A guia **Emulator control**, no painel superior direito da janela do DDMS, permite que você "simule" uma chamada telefônica ou uma mensagem de texto em um emulador.

Captura de tela

O comando **Screen capture...**, no menu **Device**, captura uma imagem a partir da tela mostrada no dispositivo ou dispositivo virtual Android selecionado.

Logging

O painel inferior da janela do DDMS mostra um registro (log) da saída dos processos no dispositivo ou dispositivo virtual selecionado. Você pode filtrar essas informações selecionando uma opção a partir dos botões na barra de ferramentas acima do painel de registro.

Dumping de estado de dispositivos, aplicativos e do rádio móvel

Um conjunto de comandos no menu **Device** permite que você comande o dispositivo ou dispositivo virtual, efetuando um dumping de estado para todo o dispositivo, para um aplicativo ou para o rádio móvel.

Documentação detalhada sobre o DDMS pode ser encontrada em: *http://developer.android.com/guide/developing/tools/ddms.html*.

Componentes do plug-in ADT para o Eclipse

O Eclipse permite que você crie tipos específicos de projetos, incluindo diversos tipos de projetos Java. O plug-in ADT adiciona a capacidade de criar e utilizar projetos Android. Quando você cria um novo projeto Android, o plug-in ADT cria a hierarquia de arquivos do projeto e todos os arquivos necessários para que um projeto Android seja corretamente construído. No caso de projetos Android, o Eclipse aplica componentes do plug-in ADT para editar, compilar, executar e depurar tais projetos.

Em alguns casos, componentes do SDK podem ser utilizados com o Eclipse ou de modo independente. Mas, na maioria dos casos de desenvolvimento de aplicativos Android abordados neste livro, a forma como esses componentes são utilizados no, ou com, o Eclipse será mais relevante do que considerá-los de maneira independente.

O plug-in ADT tem inúmeros componentes separados, e, apesar de um "plug-in" ter a conotação de ser apenas uma melhoria modesta, nesse caso estamos falando de uma quantidade substancial de software. A seguir, descreveremos cada parte significativa do plug-in ADT, que você encontrará ao trabalhar com o Eclipse no desenvolvimento de softwares para o Android.

Android Layout Editor

Layouts para interfaces de usuário em aplicativos Android podem ser especificados em XML. O plug-in ADT adiciona um editor visual que auxilia na composição e visualização de layouts para o Android. Quando você abre um arquivo de layout, o plug-in ADT automaticamente inicia esse editor para visualizar e editar o arquivo. Guias na parte inferior do painel de edição permitem que você alterne entre o editor visual e o editor XML.

Em versões mais antigas do SDK do Android, o editor de layout era limitado demais para que tivesse utilidade. Atualmente, entretanto, você deve considerar a edição visual de layouts para o Android como a forma mais indicada de sua criação. A automação da especificação dos layouts torna muito provável que seus layouts funcionarão na maioria dos dispositivos Android.

Android Manifest Editor

Projetos Android incluem um arquivo de manifesto, que acompanha o software e os recursos do projeto quando este é construído e que diz ao sistema Android como instalar e utilizar o software. O arquivo de manifesto está em linguagem XML, e um editor XML especializado é fornecido pelo plug-in ADT para editá-lo.

Outros componentes do plug-in ADT para o Eclipse, como os compiladores de aplicativos, também podem modificar o manifesto.

Editores XML para outros arquivos XML do Android

Outros arquivos XML que armazenam informações, como especificações para menus, recursos (strings, por exemplo), ou que organizam recursos gráficos de um aplicativo, têm editores especializados que serão abertos quando tais arquivos forem abertos.

Compilação de aplicativos Android

Projetos para o Eclipse são, geralmente, compilados automaticamente. Isso significa que você normalmente não encontrará um passo individual, cuja função seja transformar o código-fonte e os recursos de um projeto em um resultado implantável. O Android exige passos específicos para compilar um arquivo que possa ser implantado em um emulador ou dispositivo Android, e o plug-in ADT fornece o software que executa esses passos. Para projetos Android, o resultado da compilação é um arquivo *.apk*. No caso do projeto de teste que criamos antes, você pode encontrar esse arquivo na subpasta *bin*, na hierarquia de arquivos do projeto em seu espaço de trabalho do Eclipse.

Os compiladores específicos do Android, fornecidos com o plug-in ADT, permitem que você utilize Java como a linguagem de criação de software para o Android, ao mesmo tempo em que executa esse software em uma máquina virtual Dalvik que processa seus próprios bytecodes.

Execução e depuração de aplicativos Android

Quando você executa ou depura um projeto Android dentro do Eclipse, o arquivo *.apk* desse projeto é implantado e iniciado em um AVD ou dispositivo Android. Nesse processo, utiliza-se o ADB e o DDMS para comunicação com o AVD ou dispositivo, e com o ambiente de tempo de execução Dalvik que executa o código do projeto. O plug-in ADT adiciona os componentes que permitem ao Eclipse realizar esse processo.

DDMS

Na seção "Dalvik Debug Monitor Server (DDMS)", que vimos antes neste capítulo, descrevemos o Dalvik Debug Monitor e mostramos como invocar a interface de usuário DDMS a partir da linha de comando. A interface de usuário DDMS também está disponível dentro do Eclipse. Você pode acessá-la utilizando o comando Window > Open Perspective > DDMS nos menus do Eclipse. Você também pode acessar separadamente cada visão que compõe a perspectiva do DDMS, utilizando o menu Window > Show View e selecionando, por exemplo, a visualização Logcat.

Dispositivos virtuais Android

Dispositivos virtuais Android (AVDs) utilizam o emulador QEMU como base e são capazes de emular o hardware de um dispositivo Android, além de imagens do sistema Android, que são softwares construídos para ser executados no hardware emulado. AVDs são configurados pelo gerenciador do AVD e do SDK, que define parâmetros como o tamanho dos dispositivos de armazenamento emulados e as dimensões de tela, e que permite que você especifique qual imagem do sistema Android será utilizada em cada dispositivo emulado.

AVDs permitem que você teste seu software em um escopo mais amplo de características de sistema do que o escopo que você provavelmente conseguiria adquirir e testar em dispositivos físicos. Uma vez que tanto os emuladores de hardware com base em QEMU quanto as imagens do sistema e os parâmetros dos AVDs são componentes intercambiáveis, você pode testar dispositivos e imagens do sistema mesmo antes que haja hardware disponível para executá-los.

QEMU

O QEMU é a base dos dispositivos virtuais Android. Além disso, trata-se de uma ferramenta de ampla utilidade, empregada em diversos sistemas de emulação, mesmo fora do SDK do Android. Ainda que você configure o QEMU indiretamente, por meio do gerenciador do AVD e do SDK, em algum momento você pode ter de ajustar as configurações de emulação de formas não aceitas pelas ferramentas do SDK, ou pode estar curioso quanto às capacidades e limitações do QEMU. Por sorte, ele tem uma comunidade de usuários extensa e muito ativa, que pode ser encontrada em *http://www.qemu.org*.

Gerenciador de AVD e SDK

O QEMU é um sistema emulador de uso geral. O SDK do Android permite controle da configuração do QEMU, que é uma funcionalidade interessante quando queremos criar emuladores que executam imagens do sistema Android. O gerenciador do AVD e do SDK fornece uma interface de usuário que permite controlar dispositivos virtuais Android com base em QEMU.

Outras ferramentas do SDK

Além das principais ferramentas que você utilizará, normalmente, na maioria de seus projetos de desenvolvimento, há diversas outras ferramentas no SDK. Aquelas utilizadas ou invocadas diretamente pelos desenvolvedores estão descritas aqui. Há ainda outros componentes do SDK listados no artigo Tools Overview (Visão Geral das Ferramentas), que pode ser encontrado na documentação do Android neste endereço: *http://developer.android.com/guide/developing/tools/index.html*.

Hierarchy Viewer

O visualizador de hierarquia exibe e permite a análise da hierarquia de visualização da atividade atual, ou de um dispositivo Android selecionado. Isso permite que você encontre e diagnostique problemas na hierarquia de suas visualizações, mesmo enquanto seu aplicativo está sendo executado, ou que você analise a hierarquia das visualizações de outros aplicativos para ver como foram projetados. Ele também permite que você visualize uma representação ampliada da tela, com orientações de alinhamento que ajudam a identificar problemas nos layouts. Informações detalhadas sobre o Hierarchy Viewer podem ser encontradas neste endereço: *http://developer.android.com/guide/developing/tools/hierarchy-viewer.html*.

Layoutopt

O Layoutopt é um analisador estático que opera nos arquivos de layout XML e que pode diagnosticar problemas com os layouts do Android. Informações detalhadas sobre o Layoutopt podem ser encontradas neste endereço: *http://developer.android.com/guide/developing/tools/layoutopt.html*.

Monkey

O Monkey é uma ferramenta de automação de testes, executada em seu emulador ou dispositivo, que é invocada utilizando outra ferramenta do SDK: o adb. O adb permite que você inicie um shell em um emulador ou dispositivo, a partir do qual o monkey é invocado de maneira semelhante a este exemplo:

```
adb shell monkey --wait-dbg -p your.package.name 500
```

Essa invocação envia 500 eventos aleatórios ao aplicativo especificado (pelo nome do pacote), aguardando até que um depurador seja anexado para disparar os eventos. Informações detalhadas sobre o monkey podem ser encontradas neste endereço: *http://developer.android.com/guide/developing/tools/monkey.html*.

sqlite3

O Android utiliza o SQLite como banco de dados em vários sistemas, e fornece APIs para os aplicativos, que tornam o SQLite conveniente para armazenamento e apresentação de dados. O SQLite também tem uma interface de linha de comando, e o comando `sqlite3` permite que desenvolvedores despejem (*dump*) esquemas de bancos de dados e realizem outras operações em bancos de dados do Android.

Esses bancos de dados estão, evidentemente, em um dispositivo Android ou contidos em um dispositivo virtual Android (AVD), e, portanto, o comando `sqlite3` está disponível no shell adb. Instruções detalhadas para acessar a linha de comando do

`sqlite3`, a partir do shell adb, podem ser encontradas neste endereço: *http://developer.android.com/guide/developing/tools/adb.html#shellcommands*. Falaremos do `sqlite3` na seção "Exemplo de manipulação do banco de dados utilizando sqlite3", no capítulo 10.

keytool

O `keytool` gera chaves criptografadas e é utilizado pelo plug-in ADT para criar chaves temporárias, com as quais ele assina o código para depuração. Na maioria dos casos, você utilizará essa ferramenta para criar um certificado de assinatura quando do lançamento de seus aplicativos, como descrito na seção "Criação de um certificado autoassinado", no capítulo 4.

Zipalign

O Zipalign permite acesso otimizado aos dados das versões de produção de aplicativos Android. Essa otimização deve ser realizada apenas depois de o aplicativo ter sido assinado para lançamento (release), uma vez que a assinatura afeta o alinhamento dos bytes. Informações detalhadas sobre o Zipalign podem ser encontradas neste endereço: *http://developer.android.com/guide/developing/tools/zipalign.html*.

Draw9patch

Um "*9 patch*" é um tipo especial de recurso do Android, composto de nove imagens, sendo útil quando você deseja, por exemplo, botões que aumentem de tamanho sem alterar o raio de seus cantos. O Draw9patch é um programa de desenho especializado para criação e visualização de recursos desses tipos. Detalhes sobre o Draw9patch podem ser encontrados neste endereço: *http://developer.android.com/guide/developing/tools/draw9patch.html*.

Android

O comando `android` pode ser utilizado para invocar o gerenciador de SDK e AVD a partir da linha de comando, como descrito nas instruções de instalação do SDK que vimos na seção "O SDK do Android", anteriormente neste capítulo. Ele também pode ser utilizado para criar um projeto Android a partir da linha de comando. Dessa forma, ele faz com que sejam geradas todas as pastas de projetos, o manifesto, as propriedades de compilação e o script ant para compilação do projeto. Mais detalhes sobre o uso do comando `android` podem ser encontrados neste endereço: *http://developer.android.com/guide/developing/other-ide.html#CreatingAProject*.

Mantendo-se atualizado

O JDK, o Eclipse e o SDK do Android são fornecidos separadamente. As ferramentas de desenvolvimento de software para o Android podem sofrer alterações muito rapidamente. É por isso que, neste livro, e especialmente neste capítulo, fazemos referência ao site de desenvolvedores do Android, para que você tenha acesso às informações mais atualizadas e às versões compatíveis mais recentes de suas ferramentas. Manter suas ferramentas atualizadas e compatíveis provavelmente será uma tarefa necessária, enquanto você aprende a desenvolver softwares para o Android.

Windows, Mac OS X e Linux têm mecanismos de atualização de sistema capazes de manter seus softwares atualizados. Mas, em virtude da forma como o SDK do Android é organizado, você terá de manter atualizados sistemas de software separados, utilizando mecanismos distintos.

Mantendo o SDK do Android atualizado

O SDK do Android não é parte do sistema operacional desktop que você utiliza nem do plug-in do Eclipse, e, portanto, o conteúdo da pasta SDK não é atualizado pelo sistema operacional nem pelo Eclipse. O SDK tem seu próprio mecanismo de atualização, com uma interface de usuário no gerenciador do AVD e do SDK. Assim como na figura 1.14, selecione **Installed Packages** (Pacotes Instalados), no painel da esquerda, para acessar uma lista dos componentes do SDK instalados em seu sistema. Clique no botão **Update All** para iniciar o processo de atualização e visualizar uma lista de atualizações disponíveis.

Figura 1.14 – Atualização do SDK com o gerenciador de SDK e AVD.

Geralmente, é aconselhável instalar todas as atualizações disponíveis.

Mantendo o Eclipse e o plug-in ADT atualizados

Enquanto o SDK tem de ser atualizado fora de seu sistema operacional e do Eclipse, o plug-in ADT e todos os outros componentes do Eclipse são atualizados utilizando o sistema de gerenciamento de atualizações. Para atualizar todos os componentes que você tem em seu ambiente do Eclipse, incluindo o plug-in ADT, utilize o comando **Check for Updates** (Verificar por Atualizações), no menu **Help**. Isso mostrará todas as atualizações disponíveis (Figura 1.15).

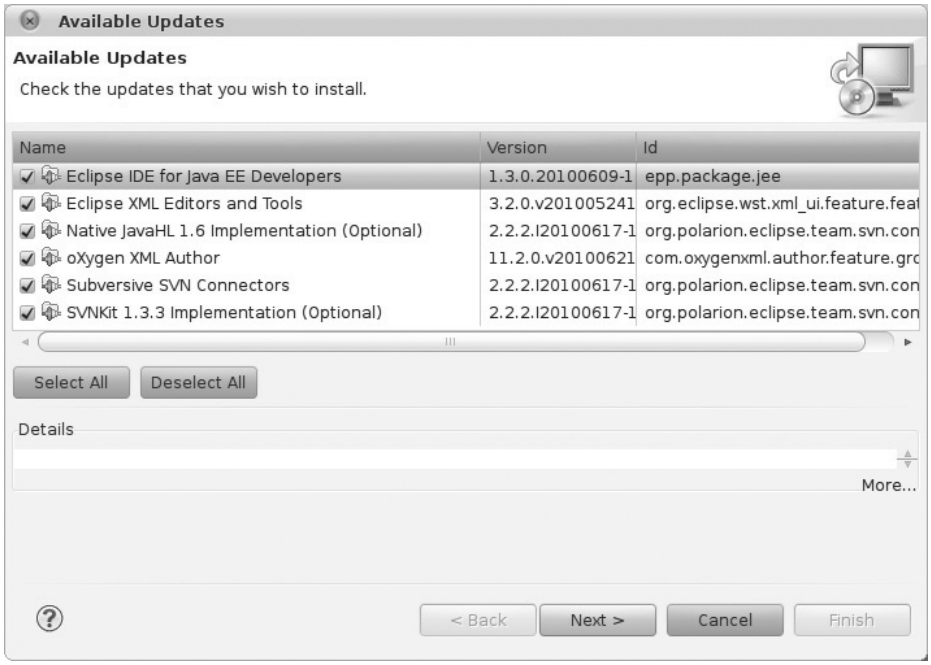


Figura 1.15 – Atualização de componentes do Eclipse e do plug-in ADT.

Normalmente, você poderá utilizar o botão **Select All** para instalar todas as atualizações disponíveis. As atualizações que você verá listadas em seu sistema dependem dos módulos do Eclipse instalados e da última atualização feita em seu Eclipse.

Mantendo o JDK atualizado

Você não terá de atualizar o Java com a mesma frequência do SDK, do plug-in ADT e de outros plug-ins do Eclipse. Mas, mesmo que o Java 7 ainda não tenha sido lançado quando você estiver lendo este texto, é provável que isso aconteça em breve e que afete o desenvolvimento para Android. Ao escolher a atualização do JDK, primeiro verifique a página de requisitos de sistema no site de desenvolvedores do Android: *http://developer.android.com/sdk/requirements.html*.

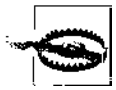
Se uma atualização for necessária e você estiver utilizando um sistema Mac ou Linux, verifique as atualizações disponíveis para seu sistema e confira se uma nova versão do JDK está incluída. Se o JDK foi instalado em seu sistema pelo fornecedor ou se você o instalou a partir de seus repositórios de distribuição Linux, atualizações estarão disponíveis por meio do mecanismo de atualizações em seu sistema.

Códigos de exemplo

Depois de instalar o SDK do Android e de verificar seu funcionamento, você estará pronto para iniciar sua exploração. Mesmo que você não esteja acostumado às classes do Android Framework e seja um novato em Java, explorar alguns códigos de exemplo servirá para aumentar sua confiança na instalação do SDK, antes de avançarmos às outras seções deste livro.

Códigos de exemplo do SDK

Os códigos de exemplo mais práticos acompanham o SDK. Você pode criar um novo projeto com base nos exemplos do SDK (Figura 1.16). O exemplo selecionado pode ser visto no painel esquerdo da janela do Eclipse, no qual você pode verificar os arquivos que o compõem e executá-lo para descobrir seu propósito. Caso você esteja acostumado a utilizar IDEs para depuração de código, pode ser interessante incluir alguns pontos de interrupção no exemplo de código, como forma de verificar quando os métodos são executados.



Nas imagens da figura 1.16 você deve selecionar um alvo de compilação antes de escolher um exemplo. Exemplos estão organizados pelo nível de API, e se você não escolheu um alvo de compilação, a lista suspensa estará vazia.

Cada aplicativo de exemplo que acompanha o SDK corresponde a um artigo do site de desenvolvedores do Android. Mais informações sobre cada exemplo podem ser encontradas nesse endereço. Todos os exemplos estão listados nesta página de documentação: *http://developer.android.com/resources/samples/index.html*.

Há mais de uma dúzia de aplicativos nessa página, sendo que um, o aplicativo de demonstração da API, aborda a maioria das APIs do Android. Criar alguns projetos com base nesses exemplos permitirá que você se familiarize com o funcionamento desses programas e ajudará a entender os próximos capítulos, mesmo que você ainda não entenda completamente o que está lendo.

Código de exemplo deste livro

Você pode fazer o download do código de exemplo deste livro neste site: *http://oreilly.com/catalog/0636920010364*.

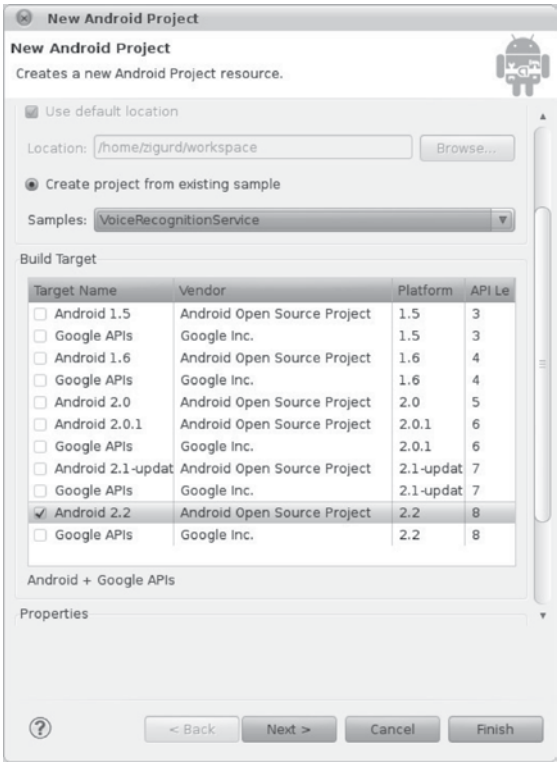


Figura 1.16 – Criação de um novo projeto utilizando códigos de exemplo do SDK.

Sobre a leitura do código

Bons programadores leem muito código. O código de exemplo fornecido pelos autores deste livro pretende ser tanto um exemplo da codificação em Java quanto do uso das capacidades da plataforma Android.

Alguns exemplos com os quais você terá contato ficarão aquém do que você necessita para criar o melhor e mais extensível software comercial possível. Muitos aplicativos de exemplo fazem escolhas justificáveis, quando o objetivo do programador é apenas criar um exemplo em uma única classe Java. Em muitos casos, aplicativos Android são versões inchadas dos códigos de exemplo, que acabam se tornando ilegíveis e de manutenção impossível. Ainda assim, isso não significa que você deva evitar a leitura de exemplos que sejam mais pontuais do que um aplicativo completo deveria ser.

O próximo capítulo explorará a linguagem Java, com o objetivo de fornecer-lhe os meios de avaliar o código de exemplo, considerando boas práticas de engenharia e projeto. Nosso objetivo é torná-lo capaz de utilizar esses exemplos e de melhorá-los, aplicando as ideias apresentadas nesses exemplos dentro de seu próprio código, visando à criação de produtos de alta qualidade.