Desenvolvimento Android

http://www.larback.com.br

Introdução ao Android

O Android é a resposta da Google ao mercado crescente de dispositivos móveis. É uma nova plataforma de desenvolvimento baseado no kernel 2.6 do sistema operacional linux.

Introdução ao Android

Embora o Google seja a grande referência do mundo Android, não é apenas ele quem mantém o sistema operacional. O Android é mantido pela Open Handset Alliance (OHA) – grupo que possui gigantes consagrados como a HTC, LG, Motorola, Samsung, ASUS, Intel e muito mais. A lista completa pode ser vista em http://www.openhandsetalliance.com/oha_members.html

Introdução ao Android

Para desenvolver aplicativos para Android, utilizamos a linguagem Java. A máquina virtual utilizada é a Dalvik e é otimizada para execução em dispositivos móveis.

Ao desenvolver, utilizaremos a linguagem Java e todos os seus recursos, mas em vez do tradicional .class, o fonte será convertido para o formato .dex (Dalvik Executable – que representa a aplicação Android compilada)

Introdução ao Android

Depois de compilado, os arquivos .dex e demais recursos da aplicação (imagens, xml etc) serão compactados em um único arquivo com a extensão .apk (Android Package File), que representa a aplicação final, pronta para ser distribuída e instalada.

Android: Preparando o ambiente

Para desenvolvermos aplicações no Android, utilizamos o Android SDK. Este vem com um emulador para simular o celular, ferramentas utilitárias e uma API completa para a linguagem Java, com todas as classes necessárias para desenvolver as aplicações.

Android: Preparando o ambiente

O SDK pode ser baixado em: http://developer.android.com/sdk/ . Podemos baixar isoladamente o SDK do Android, o Eclipse e o plugin ADT do Eclipse ou o ADT Bundle, que, em um único download traz o Eclipse, o SDK e o ADT (plugin eclipse) já configurados.

Android: Plataforma (versão)

Cada versão do sistema operacional Android é conhecida como plataforma. Cada plataforma possui um código identificador chamado de API Level:

API Level 1: Android 1.0

API Level 2: Android 1.1

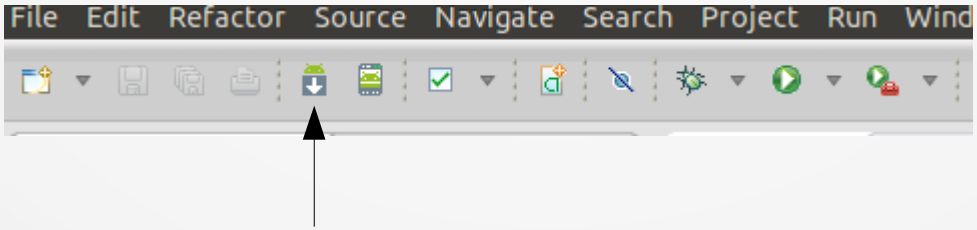
...

API Level 18: Android 4.3 (Jelly Bean)

Android: Plataforma (versão)

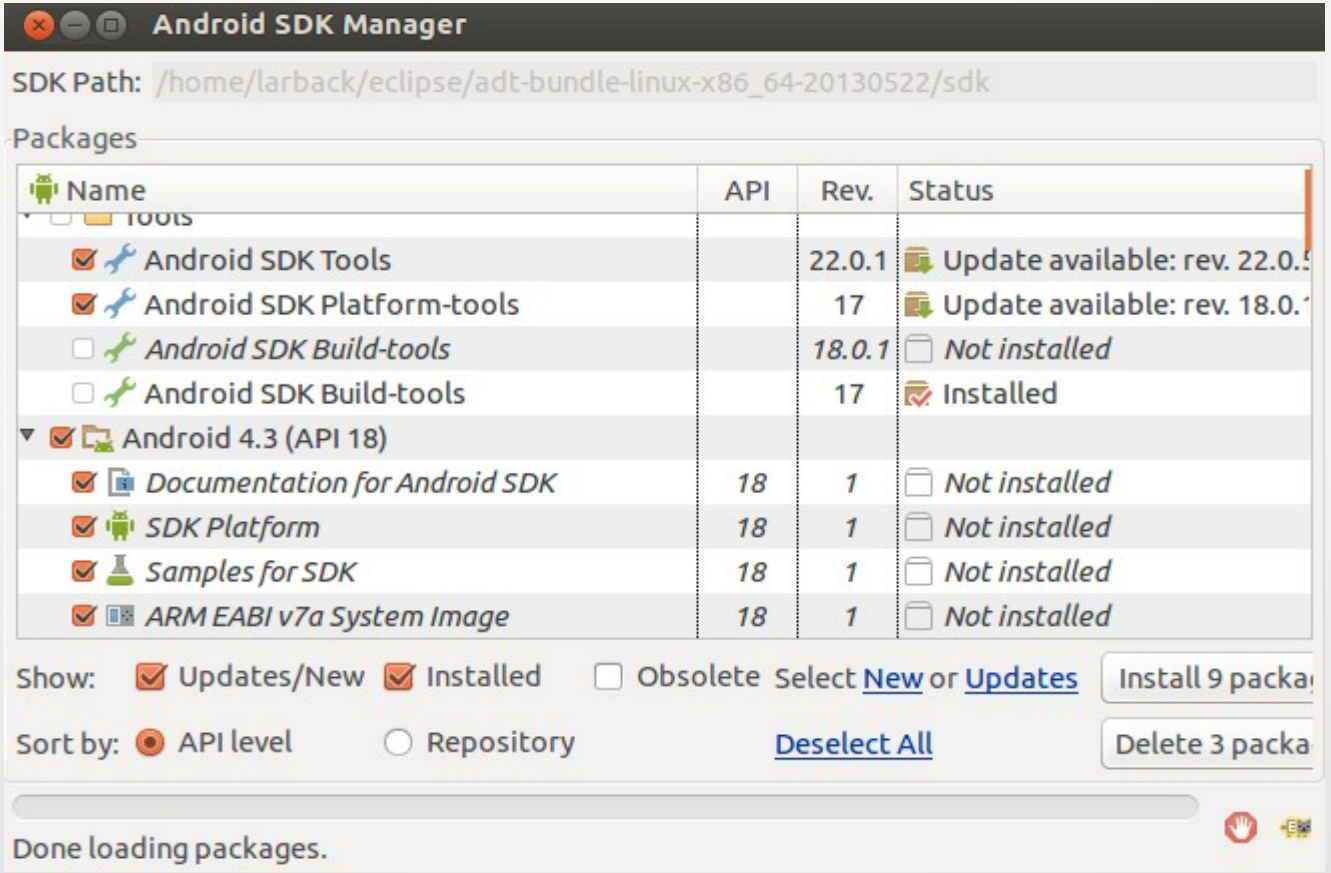
Quando instalamos o SDK ele vem vazio e desta forma é necessário instalar as plataformas desejadas. Para isto, utilizaremos o Android SDK Manager (SDK Manager.exe)

É possível abrir este aplicativo a partir do eclipse



Android: Plataforma (versão)

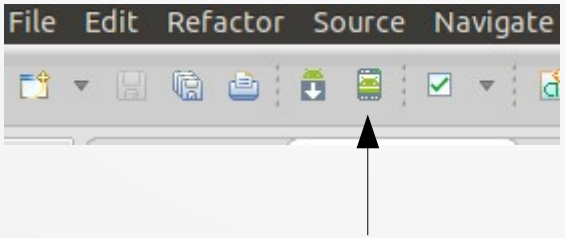
É recomendável sempre baixar a versão mais nova (atualmente 4.3) e compilar o projeto nesta versão.



Android: Configuração do AVD

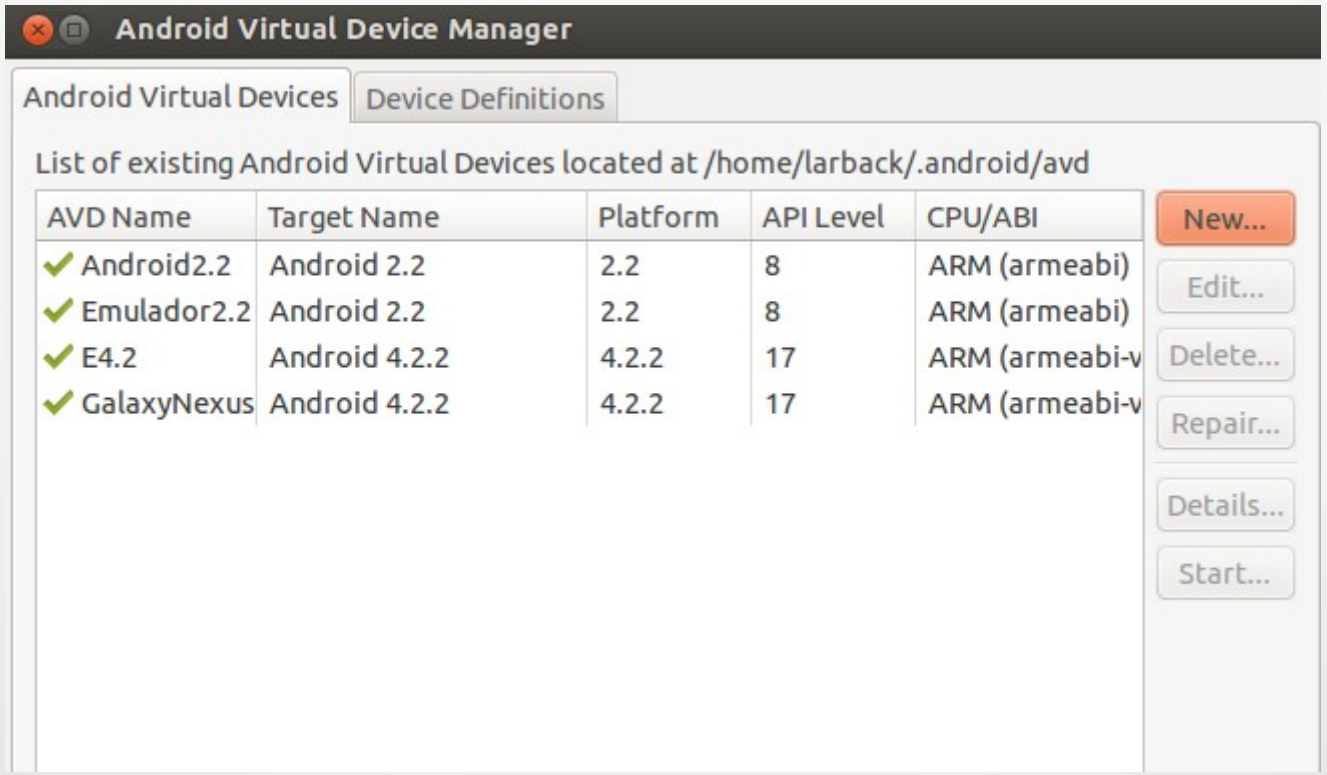
Após a instalação das plataformas, podemos criar dispositivos virtuais que simularão a configuração real de aparelhos celulares.

O utilitário de configuração pode ser aberto diretamente do eclipse:



Android: Configuração do AVD

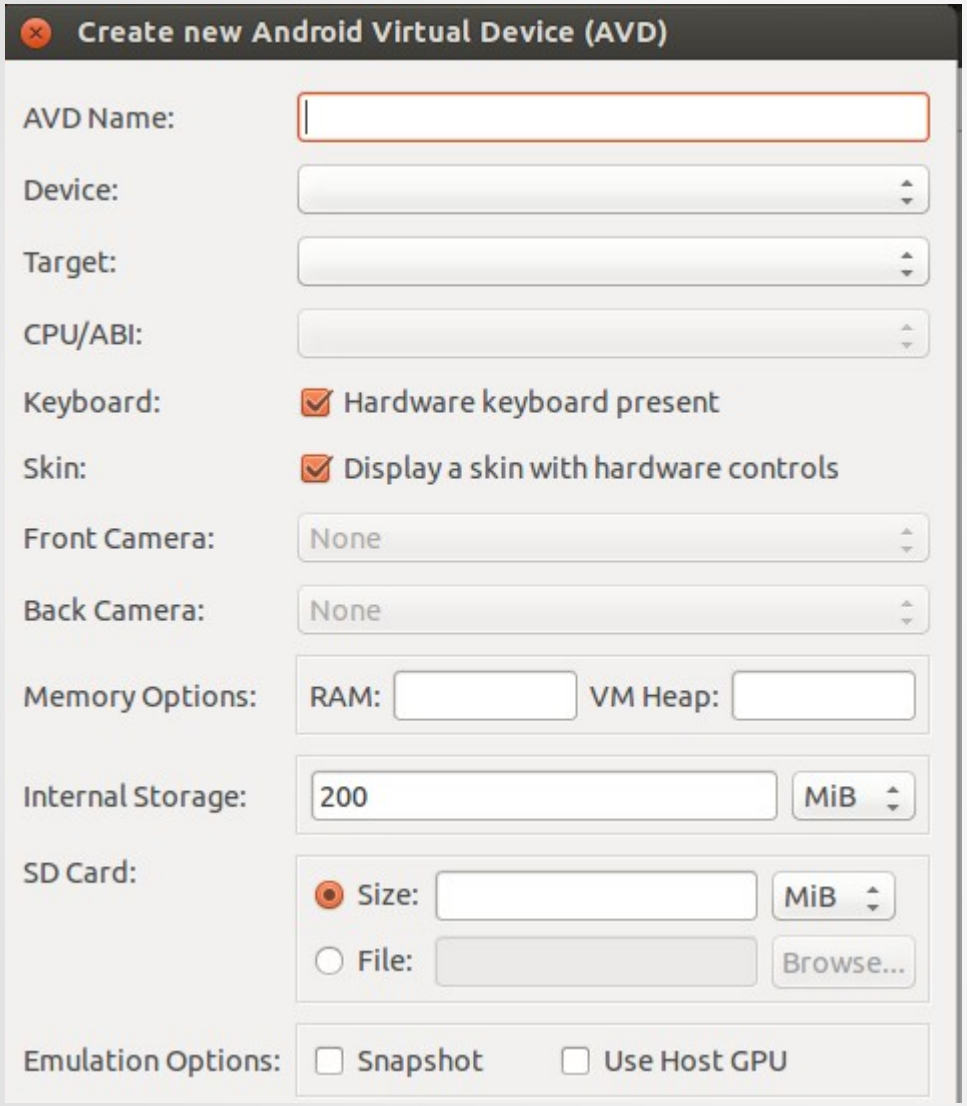
No utilitário, serão listado os AVDs já instalados. Para criar um novo basta clicar em New...



Android: Configuração do AVD

Clicando em New... você deverá informar as configurações do AVD que deseja criar.

Android: Configuração do AVD



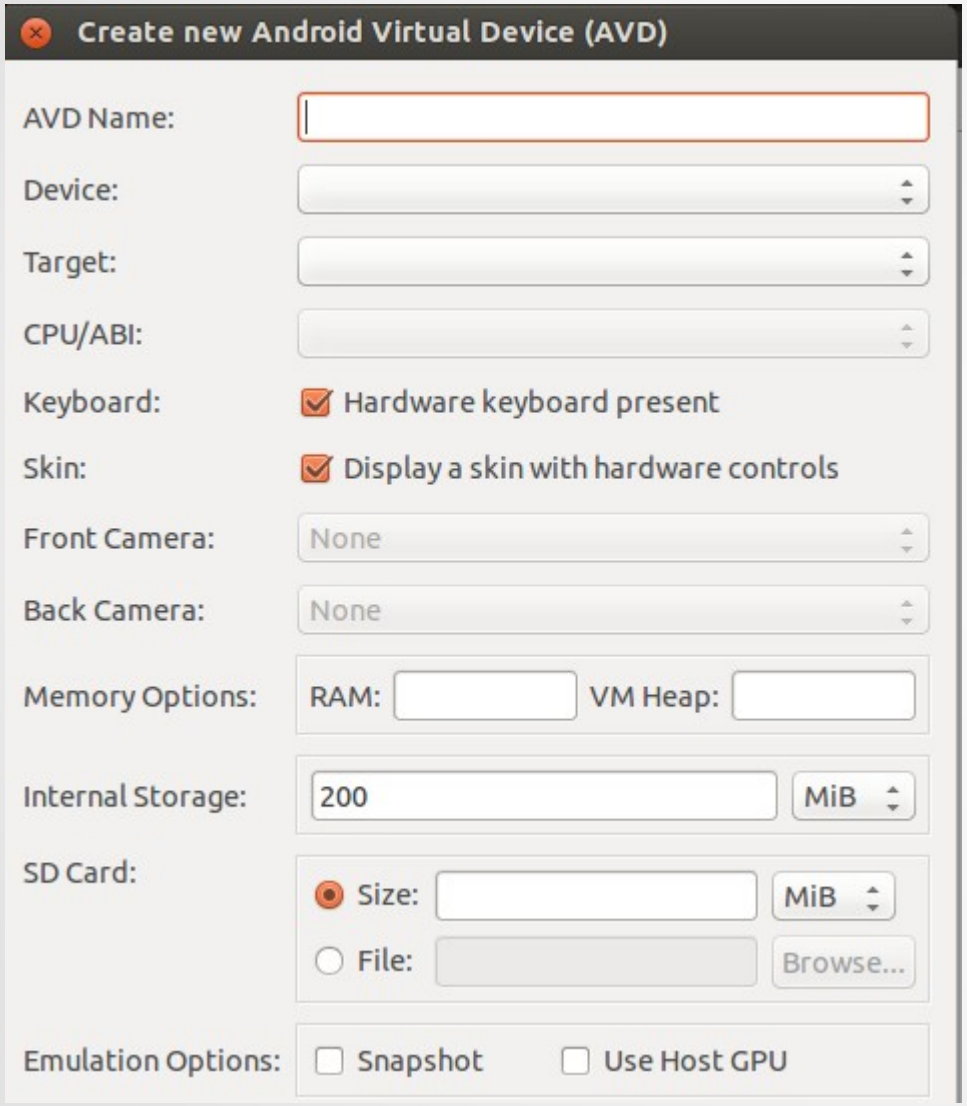
Em AVD Name, informe o nome do emulador que você vai criar.

No campo Device, você pode selecionar o tipo do dispositivo que deseja simular.

No campo Target, selecione a plataforma Android que deseja simular.

Selecione os checkbox Keyboard e Skin. O primeiro permitirá que você entre dados usando o teclado do computador (e não o virtual) e o segundo adicionará ao emulador os botões Home, Busca e Voltar

Android: Configuração do AVD



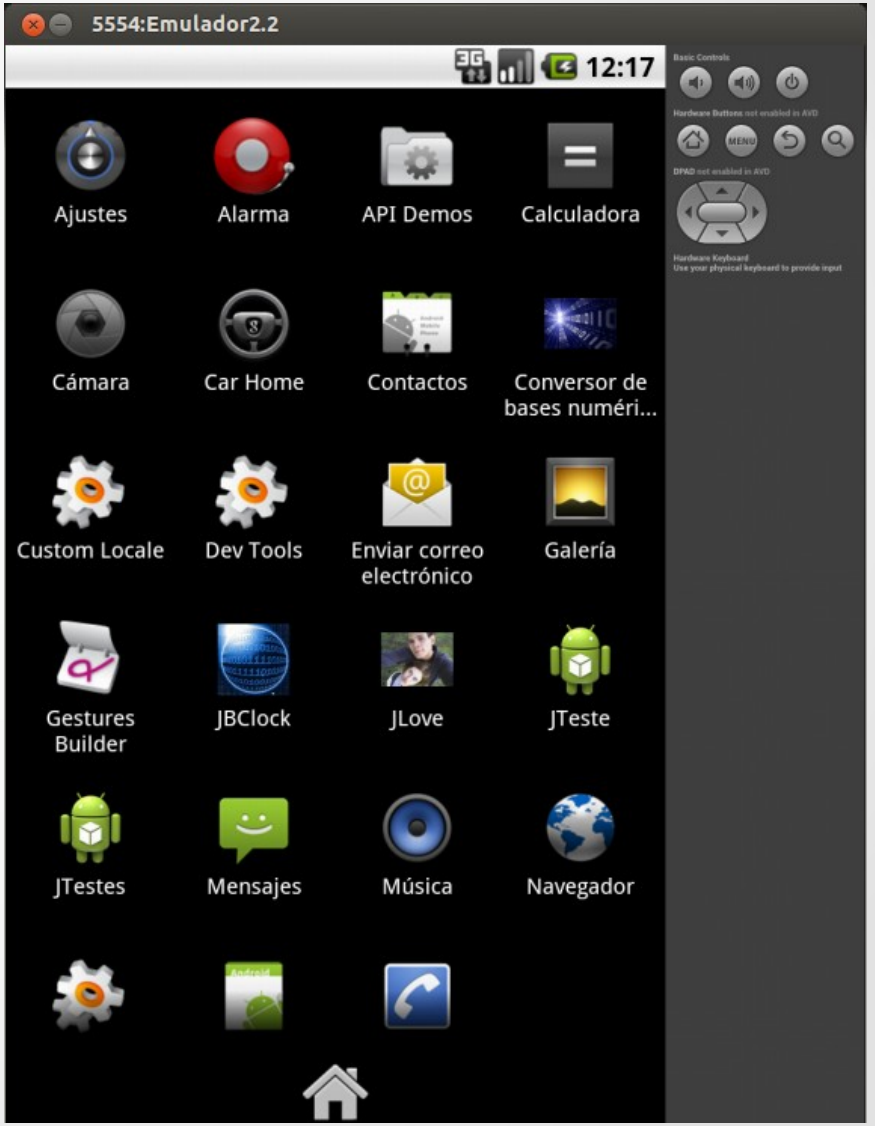
Há opções para utilizar a webcam como a câmera do Android, mas as experiências com este recurso não tem sido boas e eu não recomendo.

Deixe as opções de memória e armazenamento interno com os valores-padrão.

No SD Card, informe 60 ou um valor maior para simular um sdcard de 60Mb

Android: Configuração do AVD

Depois de configurado, para iniciar um simulador basta selecioná-lo na lista e clicar em Start...

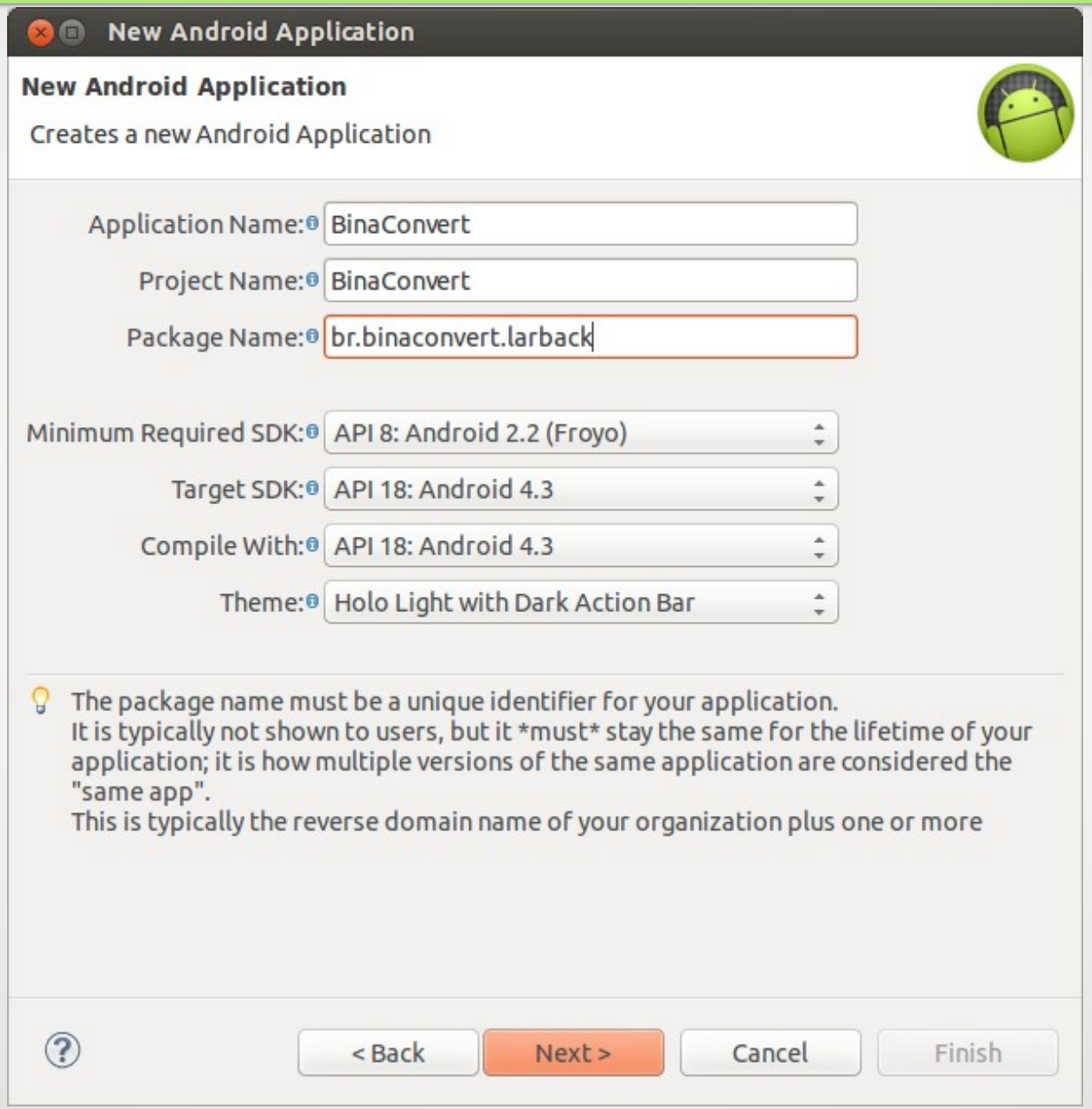


Android: Nosso primeiro projeto

No Eclipse (com o plugin ADT devidamente configurado)

- Clique em **File > New > Project**
- Na próxima janela, selecione o wizard **Android > Android Application Project** e clique em **Next**
- Insira o nome do projeto, o título da aplicação e o nome do pacote como mostrado a seguir.

Android: Nosso primeiro projeto



Android: Nosso primeiro projeto

- **Project Name**: Este é o nome do projeto que será criado no eclipse
- **Application Name**: É o título da aplicação, será visualizado na tela Home do emulador.
- **Package Name**: Nome do pacote que contém a activity principal. O nome do pacote é utilizado como identificador da aplicação pelo sistema Android.
- **Minimum Required SDK**: É a versão mínima que seu projeto vai suportar, ou seja, sua aplicação executará com esta versão ou superiores (Android 2.2 ainda é uma boa opção)

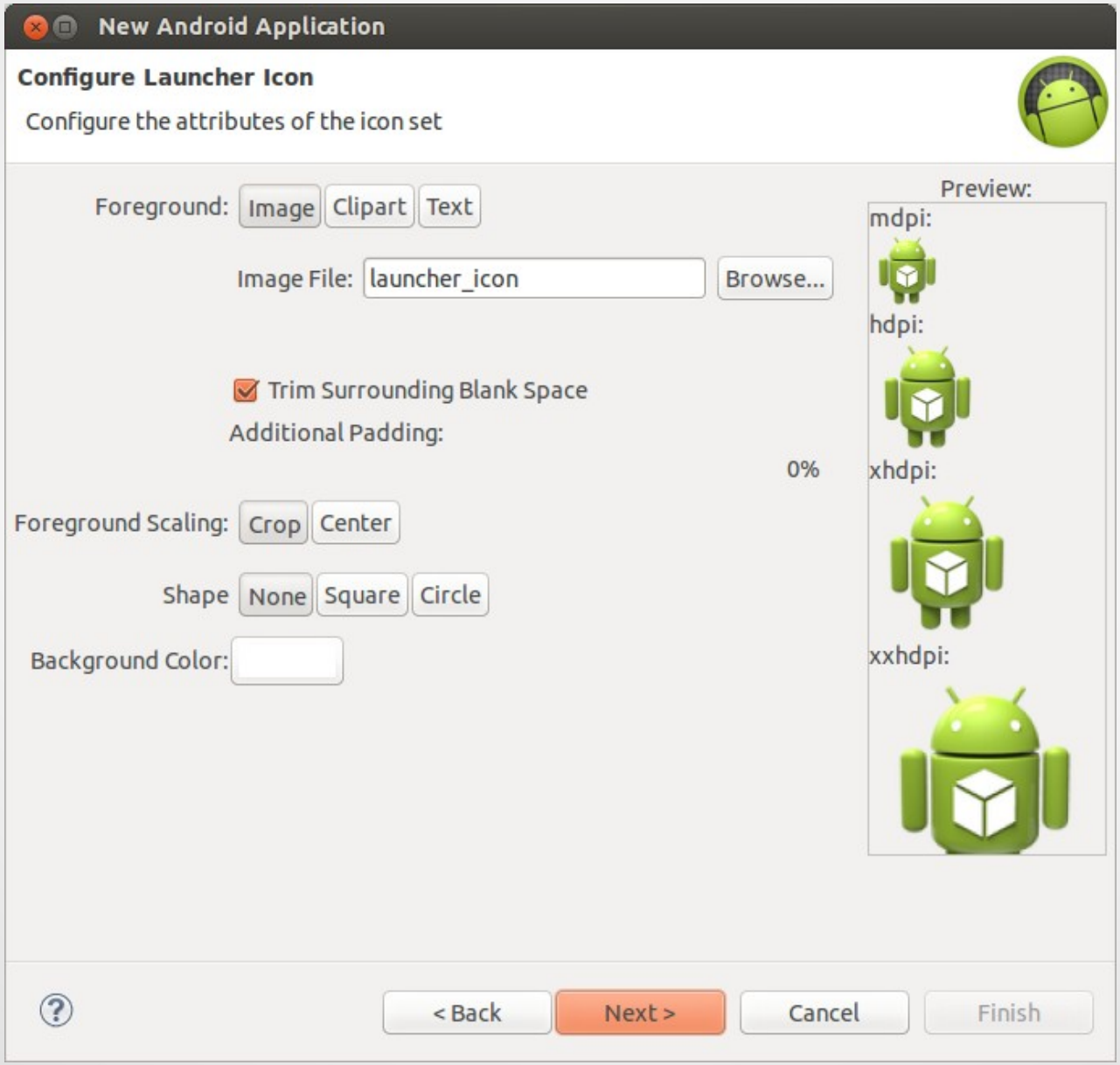
Android: Nosso primeiro projeto

- **Target SDK**: É a versão que será utilizada para otimizar o Build no momento da compilação. Selecione sempre a última versão disponível.
- **Compile With**: Versão utilizada para compilar as classes do projeto. Selecione também a última versão do Android.
- **Theme**: Selecione o tema-padrão que o wizard lhe oferecer. Clique em **Next**.
- Na próxima janela, deixe selecionado os checkboxes **Create Launcher Icon** e **Create Activity** e clique em **Next**.

Android: Nosso primeiro projeto

Na próxima janela podemos customizar os ícones do projeto. Por padrão, a imagem de ícone chama-se launcher_icon.jpg. Note que vários ícones serão criados, um para cada resolução de tela suportada pelos aparelhos Android. Clique em **Next**.

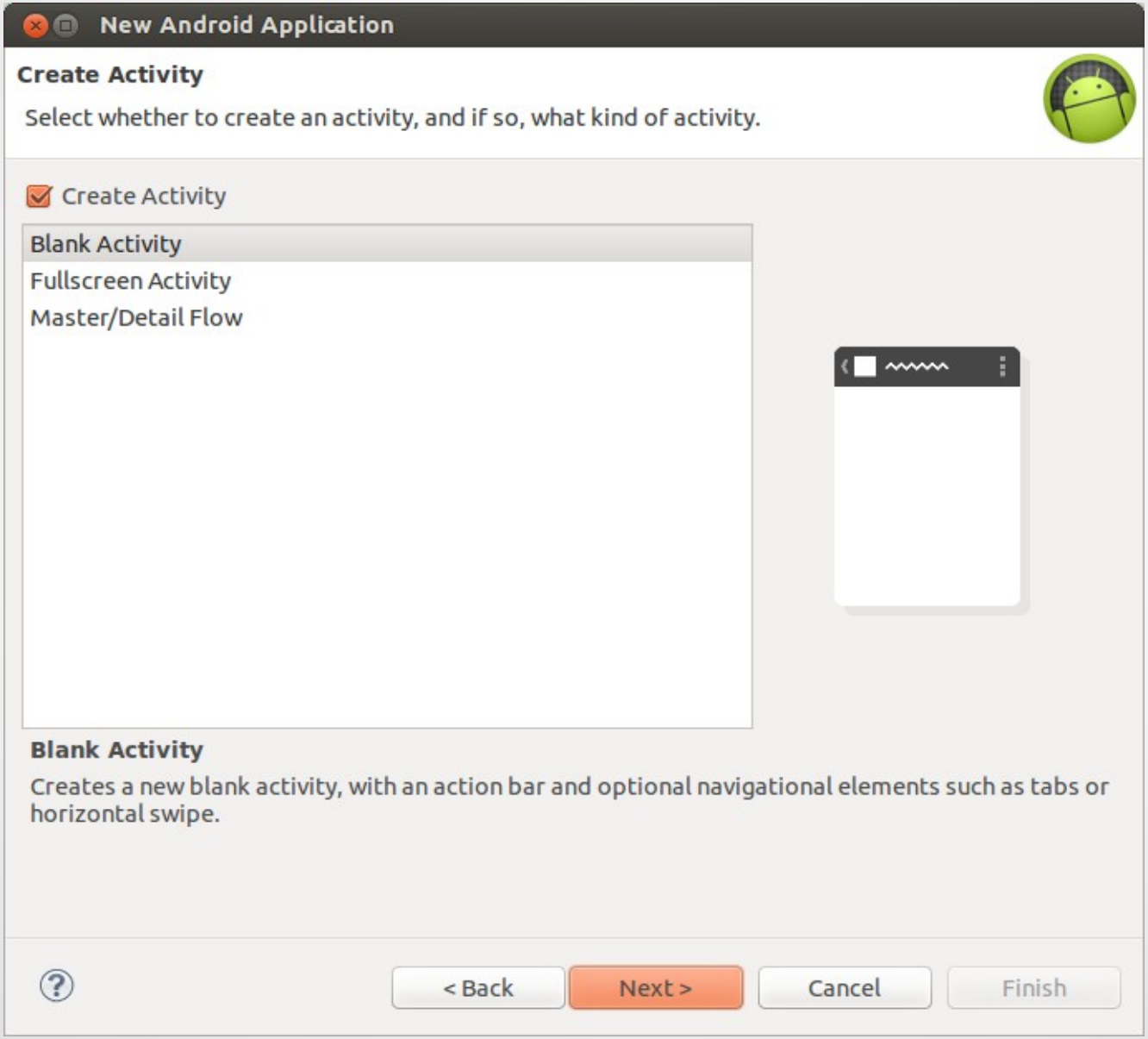
Android: Nosso primeiro projeto



Android: Nosso primeiro projeto

A próxima janela irá perguntar se deseja criar uma activity, que é uma classe que representa uma tela da aplicação. Deixe marcado o checkbox **Create Activity** e clique em **Next**. Existem alguns templates de tela que você pode escolher, mas no momento selecione o **Blank Activity**, que criará uma tela simples com uma mensagem de Hello World.

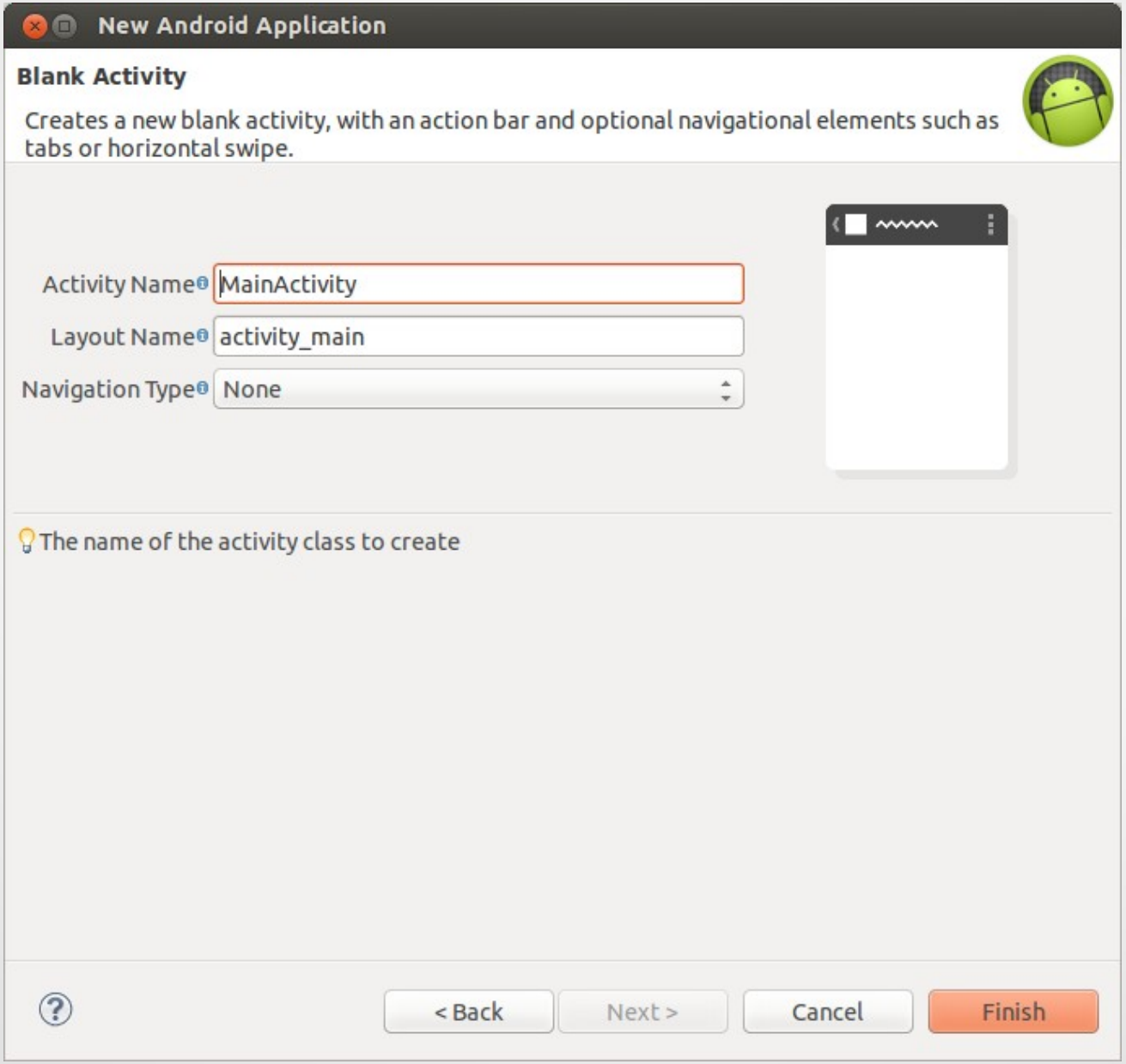
Android: Nosso primeiro projeto



Android: Nosso primeiro projeto

Na última janela do wizard, forneceremos o nome da classe. Por padrão, o nome da primeira activity é MainActivity e seu arquivo XML de layout chama-se activity_main. Na prática, isso vai gerar os arquivos MainActivity.java e activity_main.xml.

Android: Nosso primeiro projeto



Android: Nosso primeiro projeto

O projeto criado possui a seguinte estrutura de pastas:

src: Pasta que contém as classes Java. Contém a classe MainActivity que foi criada pelo wizard.

gen: Contém a classe R.java que é gerada automaticamente e permite que a aplicação acesse os recursos utilizados pelo projeto.

assets: Contém os arquivos opcionais ao projeto, como fontes personalizadas.

libs: Pasta em que podemos inserir arquivos .jars para serem adicionados ao classpath do projeto.

Android: Nosso primeiro projeto

res: Pasta com os recursos da aplicação. Tem quatro subpastas:

res/drawable: Contém as imagens da aplicação. Como existem celulares com diferentes resoluções de tela é possível customizar as imagens para ficarem no tamanho exato em cada resolução. Para isso, há quatro pastas para imagens: drawable-ldpi, drawable-mdpi, drawable-hdpi e drawable-xdpi.

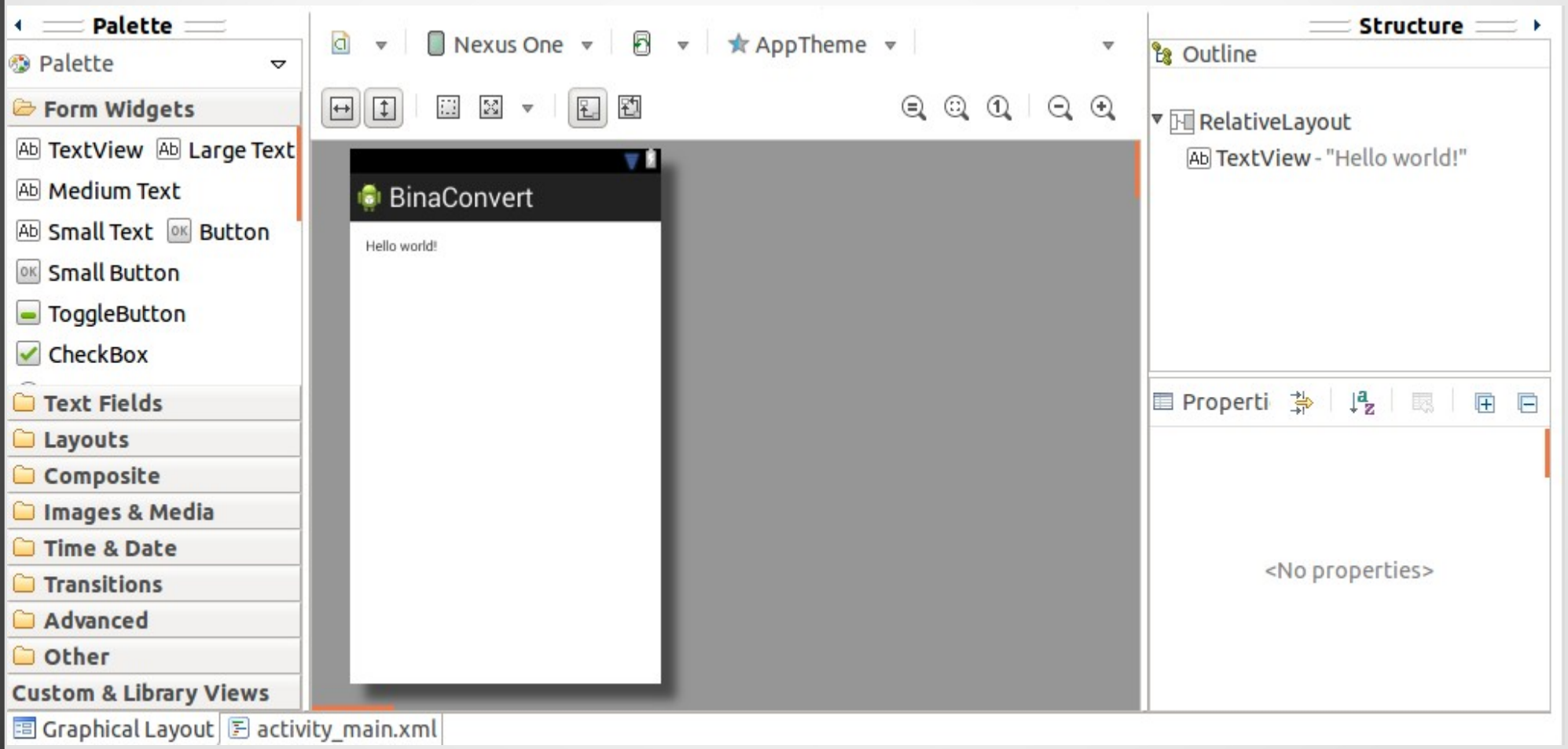
layout: Contém os arquivos XML de layouts de telas

menu: Contém os XML para criação dos menus da aplicação.

values: Contém os XML para a internacionalização da aplicação.

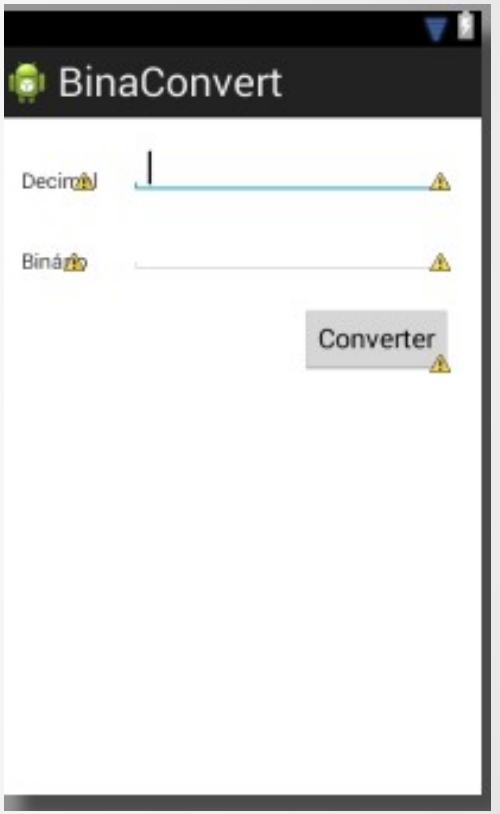
Android: Nosso primeiro projeto

Dentro da pasta layout, abra o arquivo activity_main.xml



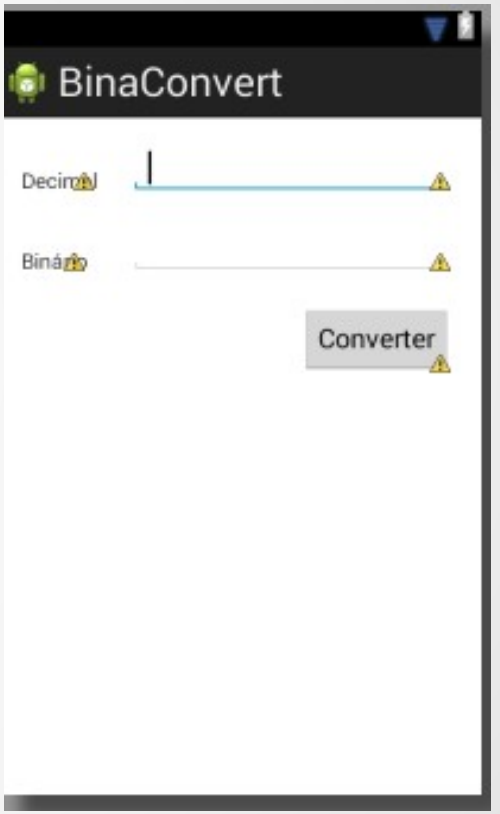
Android: Nosso primeiro projeto

O XML responsável pelo layout da activity pode ser alterado manualmente ou no modo gráfico. Imagino que você, louco por código como eu, deve estar louco para editá-lo em modo texto – entretanto, por ora, edite-o usando o assistente gráfico como mostrado na figura a seguir.



Android: Nosso primeiro projeto

Se você conhece alguma linguagem visual não terá dificuldade com o modo gráfico. Nosso layout é muito simples contendo apenas dois textView (Jlabel), dois editText (JtextField) e um button (Jbutton). Na janela de propriedades, altere os id dos editText para @+id/edtDecimal e @+id/edtBinario. O ADT irá atualizar automaticamente as referências na classe R.java



Android: Nosso primeiro projeto

Criado o layout, vamos ao código. Nosso objetivo é pegar o valor informado em decimal, convertê-lo para binário e exibi-lo no segundo editText. Como nosso aplicativo só possui uma tela (activity) todo seu código ficará no arquivo MainActivity.java. Este arquivo foi criado junto com o projeto e está dentro da pasta src.

Android: Nosso primeiro projeto

```
package br.binaconvert.larback;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

Android: Nosso primeiro projeto

public class MainActivity extends Activity {

Toda tela da sua aplicação Android é subclasse de Activity, ou seja, herda da classe Activity.java

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}
```

O método onCreate(Bundle) deve ser obrigatoriamente sobrescrito pelas subclasses de activity. Neste método, ha uma chamada ao método setContentView informando qual será o view a ser exibido na tela. Como estamos criando o layout em um arquivo XML, informamos ao método que ele deve carregar o XML activity_main.xml – Repare que utilizamos a classe R para intermediar o acesso ao XML.

Android: Nosso primeiro projeto

Vamos alterar o método, acrescentando uma resposta ao clique no botão com a conversão desejada.

Para isso, precisamos adicionar o botão ao OnClickListener – mas não é tão simples. O botão foi criado lá no XML, precisamos referencia-lo em nosso código e para isso utilizaremos novamente a classe R:

Button btConverter = (Button) findViewById(R.id.button1);

O código acima cria um objeto do tipo Button (btConverter) e utiliza o método findViewById() para associá-lo com o objeto criado no XML

Android: Nosso primeiro projeto

Após as alterações, o método ficará assim:

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    Button btConverter = (Button) findViewById(R.id.button1);
    btConverter.setOnClickListener(new OnClickListener() {
        public void onClick(View arg0) {
            EditText edtDecimal = (EditText) findViewById(R.id.edtDecimal);
            EditText edtBinario = (EditText) findViewById(R.id.edtBinario);
            int nDec = Integer.parseInt(edtDecimal.getText().toString());
            edtBinario.setText(Integer.toString(nDec,2));
        }
    });
}
```

•

Android: Nosso primeiro projeto

Até mais e obrigado pelos peixes.

Continua...