Programa de Dispositivos Móveis

Aula 02

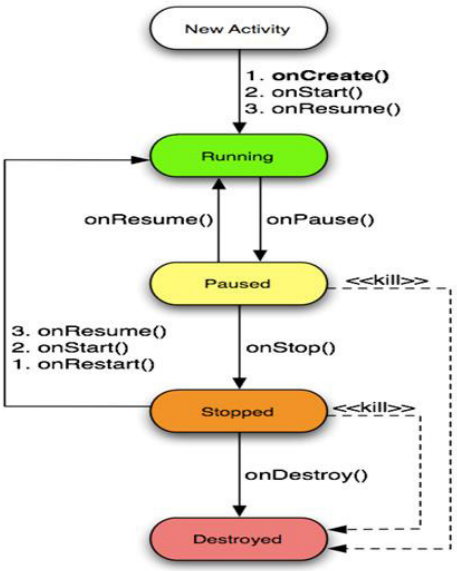
A Classe Activity

A classe Actvity é similar a classe JFrame do Swing representa basicamente uma tela. Quase todas as atividades interagem com o usuário, então ela se responsabiliza por criar a janela na qual você coloca a interface com o usuário. Uma tela é composta de vários elementos visuais, os quais são representados pela classe android.view.View(através do método setContentView(View)). Geralmente uma Atividade é apresentada ao usuário em fullscreen, mas nada impede que ela seja utilizada de outra maneira, como em telas flutuantes. Ao criar uma Atividade, sempre precisamos implementar dois métodos:

- onCreate(Bundle) – É aonde você inicia sua Atividade e define a UI (com setContentView(View) e o layout resource). Assim como você pode obter os widgets(elementos de tela) que você precisará manejar, através do método findViewById(int).
- onPause() – É quando o usuário sai da sua aplicação. É neste momento que todas as informações necessárias devem ser persistidas.É importante notar que todas as Atividades devem estar definidas em AndroidManifest.xml para que seja possível inicializá-las através do método Context.startActivity().

Um Activity tem essencialmente três estados:

1. Ele está ***active*** ou ***running***
2. Ele está ***paused*** ou
3. Ele está ***stopped***



Diretório de Recursos

Estes são os diretórios de recursos dentro de uma aplicação Android. Todos os recursos da aplicação, tais como, textos, imagens, sons, vídeos, etc. Devem obrigatoriamente estar dentro deste diretório "*res*" e em seus respectivos subdiretórios. Segue uma explicação detalhada de cada diretório:

Estrutura do projeto: src • Contém as classes Java da aplicação.Contém a classe PrincipalAloMundo que foi criada. gen • Contém a classe R que é gerada automaticamente e permite que a aplicação acesse qualquer recurso como arquivos e imagens. assets • Contém arquivos opcionais ao projeto, como por exemplo, uma fonte customizada. res • Pasta que contém os recursos da aplicação, como imagens, layouts de tela e arquivos de internacionalização (imagens detela, de botões, ícones etc.).. Tem três subpastas: drawable, layout e values.	

res/drawable • Imagens com resoluções diferentes. Como existem diversos celulares Android com resolução de tela diferente, é possível customizar as imagens para ficar com o tamanho exato. res/layout • Contém os arquivos XML utilizados para construir as tela. res/values • Contém todos os recursos relacionados a textos como os arquivos XML utilizados para a internacionalização da aplicação. • Descrição de cores, estilos, etc	

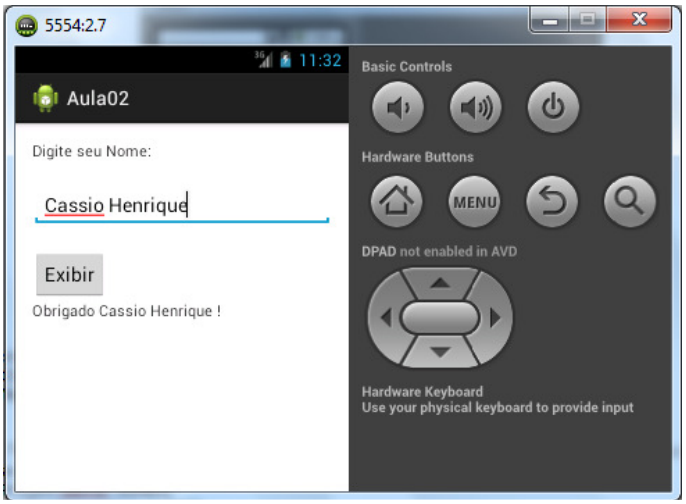
Tipos de Arquivos do Android

O Android possui os seguintes formatos de arquivos:

- **.dex** - São os bytecodes das classes Java compiladas.
- **.apk** - É a aplicação Android completa, empacotada, pronta para ser instalada em um dispositivo móvel, é gerado na pasta **bin**. Semelhante a um arquivo JAR; contém todos os recursos, todos os arquivos *.dex* e todos os arquivos de configuração e identificação necessários para instalação e execução da aplicação Android em um dispositivo móvel compatível.

Criando Aplicação Android (Aula 02)

O aplicativo que vamos desenvolver permitirá o usuário editar um nome numa caixa de texto ao clicar num botão, mostrar o seu nome dentro de uma caixa de mensagem. Vamos aprender como se faz a ligação dos elementos do layout xml (*main.xml*) com a linguagem de programação Java.



Vamos criar a nossa aplicação "Aula 02". No Eclipse, faça o seguinte:

1. Clique no menu "File" -> "New" e escolha a opção "Other..."
2. Nas opções que surgiram, selecione "Android Project" e clique em "Next"

A janela "New Android Project" que surgiu serve para que você digite os dados do projeto que será criado. Preencha-a da seguinte forma:

- ApplicationName: Aula 02
- Project Name: Aula_02
- Package name: com.pdm.aula_02
- Minimum Required SDK: API 8: Android 2.2 (Froyo)
- Target SDK: API 17: Android 4.2 (Jelly Bean)
- Compile With: API 17: Android 4.2 (Jelly Bean)
- Theme: Holo Light with Dark Action Bar
- Clique em "Next", nessa e nas proximas2 telas

- Blank Activity, "Next" novamente
- Activity Name: PrincipalActivity
- Layout Name: main
- E finalmente clique em "Finish"

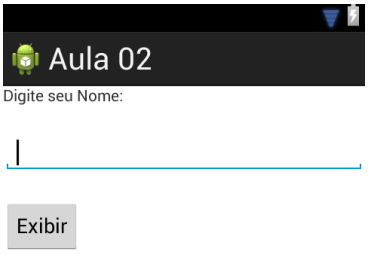
Resumo dos nomes ao iniciar um projeto:

- ApplicationName - Nome que aparecerá no menu do Android.
- Project Name - O nome do projeto que aparecerá no Package Explorer do Eclipse.
- PackageName - Nome único do pacote que identifica a aplicação.
- MinimumRiquired SDK - Versão mínima do Android necessária para funcionar a sua aplicação
- Target SDK – Versão do Android alvo da sua aplicação
- Compile With – Versão do Android que será utilizada para executar a aplicação
- Theme – Tema da aplicação

Código – Aula02

Os códigos básicos são gerados diretamente a partir do assistente New Project.

Em seguida vamos criar a tela proposta:



O arquivo main.xml, responsável pelo layout fica com o seguinte conteúdo.

Listagem 1. main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/LinearLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Digite seu Nome:"/>

    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10pt"
        android:id="@+id/editTextNome">
```

```
    <requestFocus></requestFocus>
</EditText>

<Button
    android:id="@+id/buttonExibir"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="10pt"
    android:text="Exibir"/>

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text=""
    android:id="@+id/textViewResultado"/>

</LinearLayout>
```

O arquivo PrincipalLab1.Java tem a classe PrincipalLab01 que estende a classe Activity. Ao estender a classe Activity temos acesso à interação com o sistema Android. Podemos associar um Actvity como sendo uma Tela do Android. Logo se tivermos várias Telas precisaremos da vários Activity´s.

```
public class PrincipalActivity extends Activity  { … }
```

Quando essa atividade é iniciada, o método *onCreate* é invocado, passando um *savedInstanceState*.O método *onCreate* é uma substituição do método de classes de atividade de mesmo nome. Ele chama o método *onCreate* da superclasse.

```
super.onCreate(savedInstanceState);
```

Uma chamada para *setContentView()* associa o layout da UI definido no arquivo *main.xml*. Tudo que estiver no *main.xml* e no *strings.xml* é automaticamente mapeado para as constantes definidas no arquivo de origem *R.java*. Nunca edite esse arquivo diretamente, já que ele é alterado em cada compilação.

```
setContentView(R.layout.main);
```

A nossa classe PrincipalLab01vai ter como atributos, os elementos com os quais queremos interagir. Estes são o campo de texto (*EditText*) que guarda o nome e o campo rótulo (*TextView*) que exibe o conteúdo do campo texto; o botão que vai preencher o campo rótulo com o nome digitado.O elemento *TextView* equivale ao *TLabel* do *Swing*.

```
edtNome = (EditText) findViewById(R.id.editTextNome);
txtResultado = (TextView) findViewById(R.id.textViewResultado);
btnExibir = (Button) findViewById(R.id.buttonExibir);
```

Agora que temos os atributos necessários vamos interagir com o sistema Android para isso temos que chamar os elementos que estão no nosso layout xml, isto é, no *main.xml*.

Como sabemos os elementos que aparecem na tela são criados editando os seus códigos ou criando-os através da tela gráfica. Todos os elementos que são referenciados no arquivo *PrincipalActivity.java* devem ser identificados, através dos seus id's criados no arquivo *main.xml*.No arquivo do código Java o método *findViewById(<id do elemento>)*faz uma cópia do elemento que aparece na tela para que o mesmo seja manipulado pelo programador.

Agora vamos ver quando o botão for clicado. Para isso temos de ter um listener do botão. Primeiro temos que identificar a interface *OnClickListener* na classe PrincipalLab01 e em seguida codificar o tratamento do evento. O evento onClick verifica primeiro se a caixa de texto está preenchida, caso negativo é enviada uma mensagem ao usuário pedindo que a preencha. Caso a caixa esteja preenchida é mostrado o conteúdo digitado utilizando o elemento view.

```
btnExibir.setOnClickListener(new Button.OnClickListener() {

    @Override
    public void onClick(View arg0) {
```

```
                String nome = edtNome.getText().toString();

                // Verifica se foi digitado o nome
                if (nome.equals("")) {
                        // Exibe uma mensagem de erro
                        Toast.makeText(getApplicationContext(), "Favor digitar o nome.",
Toast.LENGTH_SHORT).show();
                } else {
                        txtResultado.setText("Obrigado " + nome + " !");
                }
        }
    });
```

Listagem 2. PrincipalLab01.java

```
package com.pdm.aula_02;

import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class PrincipalActivity extends Activity {

    EditText edtNome;
    TextView txtResultado;
    Button btnExibir;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        edtNome = (EditText) findViewById(R.id.editTextNome);
        txtResultado = (TextView) findViewById(R.id.textViewResultado);
        btnExibir = (Button) findViewById(R.id.buttonExibir);

        btnExibir.setOnClickListener(new Button.OnClickListener() {

            @Override
            public void onClick(View arg0) {
                String nome = edtNome.getText().toString();

                // Verifica se foi digitado o nome
                if (nome.equals("")) {
                    // Exibe uma mensagem de erro
                    Toast.makeText(getApplicationContext(), "Favor digitar o
nome.", Toast.LENGTH_SHORT).show();
                } else {
                    txtResultado.setText("Obrigado " + nome + " !");
                }
            }
        });
    }
}
```

Listagem 3. strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
```

```
    <string name="app_name">Aula 02</string>
    <string name="action_settings">Settings</string>

</resources>
```

Listagem 4. AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.pdm.aula_02"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="17" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.pdm.aula_02.PrincipalActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

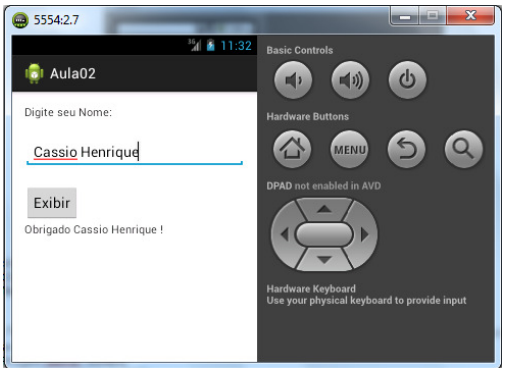
Fazendo Deploy da Aplicação para o Emulador

Primeiramente, vamos alterar a perspectiva do Eclipse para "Java EE"; para isto, clique em "Window" -> "Open Perspective" -> "Other...", selecione a opção "Java EE (default)" e clique em "Ok".

Para compilarmos a nossa aplicação e fazermos o deploy dela para o emulador, faça os seguintes passos:

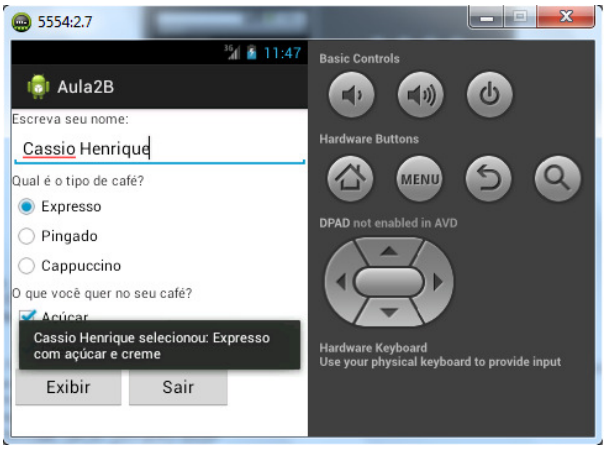
1. Certifique-se de que o emulador está em execução.
2. Clique com o botão esquerdo sobre **projeto** "Aula_02" do lado esquerdo da tela.
3. Selecione a opção "Run As" e "Android Application" A aplicação será compilada, empacotada e instalada no emulador do Android.

Acompanhe o progresso da operação através da view***Console*** do Eclipse.Quando estiver terminado, a aplicação surgirá na tela do emulador.



Criando Aplicação Android (Aula02B)

O aplicativo que vamos desenvolver permitirá o usuário editar um nome numa caixa de texto ao clicar num botão, mostrar o seu nome dentro de uma caixa de mensagem. Vamos aprender como se faz a ligação dos elementos do layout xml (AndroidManifest.xml) com a linguagem de programação Java.



No Eclipse, faça o seguinte:

1. Clique no menu "File" -> "New" e escolha a opção "Other..."
2. Nas opções que surgiram, selecione "Android Project" e clique em "Next"

A janela "New Android Project" que surgiu serve para que você digite os dados do projeto que será criado. Preencha-a da seguinte forma:

- Project Name: Aula02B
- Application name: Aula 02B
- Package name: com.pdm.aula_02B
- CreateActivity: PrincipalActivity
- Min SDK Version: 8
- Clique em "Finish"

Na Prática – Aula02B

Listagem 1. main.xml

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/LinearLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:id="@+id/textView1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Escreva seu nome:"/>

    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="1pt"
        android:id="@+id/editTextNome">
        <requestFocus></requestFocus>
      </EditText>
```

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="2pt"
    android:text="Qual é o tipo de café?"
    android:id="@+id/textView2"/>

<RadioGroup
    android:id="@+id/radioGroupTipo"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="1pt">

    <RadioButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/radioButtonExpresso"
        android:checked="true"
        android:text="Expresso"/>

    <RadioButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Pingado"
        android:id="@+id/radioButtonPingado"/>

    <RadioButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Cappuccino"
        android:id="@+id/radioButtonCappuccino"/>

</RadioGroup>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="2pt"
    android:id="@+id/textView2"
    android:text="O que você quer no café?"/>

<CheckBox
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="1pt"
    android:text="Açucar"
    android:id="@+id/checkBoxAcucar"/>

<CheckBox
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="3pt"
    android:text="Creme"
    android:id="@+id/checkBoxCreme"/>

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:id="@+id/linearLayout1"
    android:weightSum="1">

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
            android:layout_weight="0.28"
            android:text="Exibir"
            android:id="@+id/buttonExibir"/>

        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.28"
            android:text="Sair"
            android:id="@+id/buttonSair"/>

    </LinearLayout>

</LinearLayout>
```

Listagem 2. strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">Aula 02B</string>
    <string name="action_settings">Settings</string>

</resources>
```

Listagem 3. PrincipalActivity.java

```
package com.pdm.aula_02b;

import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.Toast;

public class PrincipalActivity extends Activity implements OnClickListener {

       private Button btnExibir;
       private Button btnSair;
       private CheckBox chbAcucar;
       private CheckBox chbCreme;
       private EditText edtNome;
       private RadioGroup rdgTipo;

       @Override
       public void onCreate(Bundle savedInstanceState) {
              super.onCreate(savedInstanceState);
              setContentView(R.layout.main);
              btnExibir = (Button) this.findViewById(R.id.buttonExibir);
              btnSair = (Button) this.findViewById(R.id.buttonSair);
              chbAcucar = (CheckBox) this.findViewById(R.id.checkBoxAcucar);
              chbCreme = (CheckBox) this.findViewById(R.id.checkBoxCreme);
              edtNome = (EditText) this.findViewById(R.id.editTextNome);
              rdgTipo = (RadioGroup) this.findViewById(R.id.radioGroupTipo);
              btnExibir.setOnClickListener(this);
              btnSair.setOnClickListener(this);
       }

       @Override
```

```
	public void onClick(View v) {
		if (v.getId() == btnExibir.getId()) {
			String tipo = "";
			String texto;

			int radId = rdgTipo.getCheckedRadioButtonId();
			RadioButton radSelecionado = (RadioButton) findViewById(radId);
			texto = radSelecionado.getText().toString();

			if (chbAcucar.isChecked()) {
				tipo = " com açúcar";
			}
			if (chbCreme.isChecked()) {
				tipo = " com creme";
			}
			if (chbAcucar.isChecked() && chbCreme.isChecked()) {
				tipo = " com açúcar e creme";
			}
			Toast.makeText(getApplicationContext(), edtNome.getText().toString() + "
selecionou: " + texto + " " + tipo, Toast.LENGTH_SHORT).show();
		}
		if (v.getId() == btnSair.getId()) {
			Toast.makeText(getApplicationContext(), "Saindo...",
Toast.LENGTH_SHORT).show();
//			this.e
		}
	}
}
```

O código básico é gerado diretamente a partir do assistente New Project:

- Ele faz parte de um pacote Java chamado com.pdm.aula_02B.
- Ele possui duas importações:
  - Uma para a classe de atividade
  - Uma para a classe de pacote configurável
- Quando essa atividade é iniciada, o método onCreate é invocado, passando um savedInstanceState. Não se preocupe com esse pacote configurável em relação aos nossos propósitos; ele será utilizado quando uma atividade for suspensa e depois continuada.
- O método onCreate é uma substituição do método de classes de atividade de mesmo nome. Ele chama o método onCreate da superclasse.
- Uma chamada para setContentView() associa o layout da UI definido no arquivo main.xml. Tudo que estiver no main.xml e no strings.xml é automaticamente mapeado para as constantes definidas no arquivo de origem R.java. Nunca edite esse arquivo diretamente, já que ele é alterado em cada compilação.

A configuração do arquivo AndroidManifest.xml para o aplicativo PrincipalLab02 é mostrada abaixo.

Listagem 4. AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.pdm.aula_02b"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="17" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
```

```
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.pdm.aula_02b.PrincipalActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Fazendo Deploy da Aplicação para o Emulador

Primeiramente, vamos alterar a perspectiva do Eclipse para "Java EE"; para isto, clique em "Window" -> "Open Perspective" -> "Other...", selecione a opção "Java EE (default) " e clique em "Ok".

Para compilarmos a nossa aplicação e fazermos o deploy dela para o emulador, faça os seguintes passos:

4. Certifique-se de que o emulador está em execução.
5. Clique com o botão esquerdo sobre **projeto** "Aula02B" do lado esquerdo da tela.
6. Selecione a opção "Run As" e "Android Application" A aplicação será compilada, empacotada e instalada no emulador do Android.

Acompanhe o progresso da operação através do Console do Eclipse.Quando estiver terminado, a aplicação surgirá na tela do emulador.