Programa de Dispositivos Móveis

Aula 02 - Exercício

Criando Aplicação Android (BasicView)

Vamos continuar a criar interfaces gráficas (UI) básicas com Android, utilizando o Eclipse como IDE de desenvolvimento.

Como sabemos o Android oferece dois modos de criar interfaces gráficas:

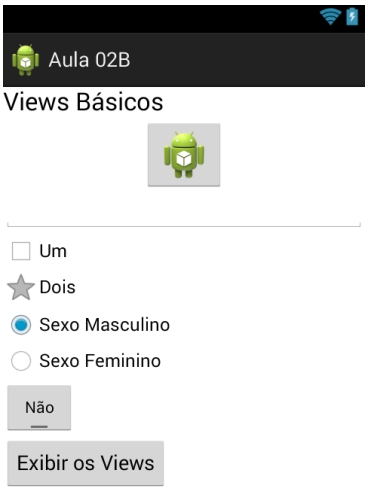
- Uma é através da definição de um arquivo XML. Para configurar este arquivo pode ser usado um modo gráfico, que consiste em arrastando componentes visual (views) para dentro dele, ou codificar diretamente os componentes no arquivo. Este arquivo (xml) será carregado no startup da aplicação e a renderização da tela é construída em tempo de execução.
- O outro modo é através da codificação do componente diretamente dentro da classe Activity. Os componentes são codificados dentro do método onCreate que é executado toda vez que o Activity for ativado.

Na maioria dos casos o desenvolvedor usará as duas maneiras.

Views

Vamos aprender sobre os componentes representados pelas classes que herdam de View. Neste exercício vamos criar um novo projeto no Eclipse, chamado BasicView2, para criarmos exercícios de alguns views .

Os views básicos que vamos exercitar são:
TextView • EditText • Button • ImageButton • CheckBox • ToggleButton • RadioButton • RadioGroup



Vamos editar nosso main.xml do projeto. Reescreva-o com as seguintes informações:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/scrollView1"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <LinearLayout
```

```
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:orientation="vertical" >

<TextView
    android:id="@+id/txtView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Views Básicos"
    android:textAppearance="?android:attr/textAppearanceLarge"/>

<ImageButton
    android:id="@+id/imageButtonUm"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:layout_marginTop="1dp"
    android:src="@drawable/ic_launcher"/>

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

<CheckBox
    android:id="@+id/checkBox1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="1dp"
    android:text="Um" />

<CheckBox
    android:id="@+id/checkBox2"
    style="?android:attr/starStyle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Dois" />

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

    <RadioButton
        android:id="@+id/radioButtonMasculino"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:checked="true"
        android:text="Sexo Masculino" />

    <RadioButton
        android:id="@+id/radioButtonFeminino"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Sexo Feminino" />

</RadioGroup>

<ToggleButton
    android:id="@+id/toggleButton1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="1dp"
```

```
            android:text="ToggleButton"
            android:textOff="Não"
            android:textOn="Sim" />

        <Button
            android:id="@+id/buttonExibir"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="1dp"
            android:text="Exibir os Views" />

    </LinearLayout>

</ScrollView>
```

Note que utilizamos um componente chamada ScrollView para caso a tela do Android não consiga exibir todos os views será criada uma barra de rolamento(scrollbar) na lateral da tela para facilitar a visualização de todos os elementos.

Após o ScroolView e o LinearLayout, o primeiro componente é um TextView. Utilizado para exibir um texto na tela do Android.

O segundo componente é um ImageButton, mesclando uma imagem ao botão tradicional. Com esta view sempre devemos definir a fonte da imagem, no nosso exemplo utilizamos:

```
android:src="@drawable/ic_launcher"
```

Neste caso acessamos recursos através do XML, sendo assim devemos definir um '@', a localização do recurso (drawable, layout ou values), barra e o nome do recurso. Já na classe Activity usamos a seguinte linha de código: `setContentView(R.layout.main);` o recurso é acessado através do código Java. Neste caso, você indica a classe R e, o caminho referente ao recurso desejado.

O terceiro componente é um campo de texto, representado pela classe EditView. O EditText tem atributos importantes. Em alguns casos, o programador adiciona um campo de texto onde o usuário informa sua senha, sendo assim, o texto deve ser substituído por asteriscos. Para atingir este objetivo, existe uma propriedade que pode ser adicionada ao EditText:

```
android:password="true"
```

O quarto e o quinto componente é um simples CheckBox. O quinto componente que também é um CheckBox, mas tem uma especificidade. Redefinimos seu estilo, veja:

```
style="?android:attr/starStyle"
```

O sexto e sétimo componente são instâncias de RadioButton, dentro de um RadioGroup.

O oitavo componente é um ToggleButton, um botão estilizado com dois estados: on e off. Também podemos mudar os textos do ToggleButton através de mais duas propriedades: `android:textOff="Não"` `android:textOn="Sim"` Com isso o componente apresenta textos definidos pelo programador, e não aqueles padrões.

Por fim demos um Button utilizado para exibir uma mensagem a ser acionado.

Tratamento de Eventos

Como já falamos rapidamente a interface gráfica de um aplicativo pode ser construída de duas maneiras: diretamente com o XML ou codificando cada componente. Por exemplo, já usamos o TextView no XML, porém, as classes dos views podem ser instanciada e tratada diretamente via código.

Mesmo quando optamos por criar a interface via XML, que é mais indicada inclusive pelo site de desenvolvedores do Android, vamos usar a codificação pura em Java para tratar os eventos. Veja a classe PrincipalActivity, localizada na pasta src:

Na classe PrincipalActivity temos:

```
package br.com.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.ImageButton;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.RadioGroup.OnCheckedChangeListener;
import android.widget.Toast;
import android.widget.ToggleButton;


public class PrincipalActivity extends Activity  implements OnClickListener,
OnCheckedChangeListener {

       ImageButton imgBotao;
       EditText edtNome;
       CheckBox chkUm,chkDois;
       RadioGroup radGrupo;
       RadioButton radMasculino,radFeminino;
       ToggleButton tgbUm;
       Button btnExibir;

      @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        btnExibir = (Button) findViewById(R.id.buttonExibir);
        edtNome = (EditText) findViewById(R.id.editTextNome);
        chkUm = (CheckBox) findViewById(R.id.checkBox1);
        chkDois = (CheckBox) findViewById(R.id.checkBox2);
        radMasculino = (RadioButton) findViewById(R.id.radioButtonMasculino);
        radFeminino = (RadioButton) findViewById(R.id.radioButtonFeminino);
        tgbUm = (ToggleButton) findViewById(R.id.toggleButton1);
        imgBotao = (ImageButton) findViewById(R.id.imageButtonUm);
        btnExibir.setOnClickListener(this);
        imgBotao.setOnClickListener(this);

        radGrupo = (RadioGroup) findViewById(R.id.radioGroupUm);
        radGrupo.setOnCheckedChangeListener(this);
      }


    public void onClick(View v) {
      String nome = edtNome.getText().toString();
      String checkUm = "";
      String checkDois = "";
      String sexo="";
      String toggleUm;

      if (v.getId() == imgBotao.getId()) {
            Toast.makeText(getBaseContext(), "Clicou no ImageButton",
Toast.LENGTH_SHORT).show();
      }
      if (v.getId() == btnExibir.getId()) {
```

```
            if (chkUm.isChecked()) {
                  checkUm = "Um ligado";
            } else {
                  checkUm = "?";
            }
            if (chkDois.isChecked()) {
                  checkDois = "Dois ligado";
            } else {
                  checkDois = "?";
            }
            if (radMasculino.isChecked()) {
                  sexo = "Masculino";
            } else {
                  sexo = "Feminino";
            }
            toggleUm = tgbUm.getText().toString();

            Toast.makeText(getBaseContext(), "Nome: " + nome + "\n" + "Check Um: " +
checkUm + " e Check Dois: " + checkDois + "\n" + "Radio: " + sexo + "\n" + "Toggle: " +
toggleUm, Toast.LENGTH_SHORT).show();
      }
    }

    public void onCheckedChanged(RadioGroup group, int checkedId) {
      RadioButton radBotao = (RadioButton) this.findViewById(checkedId);
      if (radBotao.getText().equals("Sexo Feminino")) {
                  imgBotao.setImageResource(R.drawable.menina);
                  Toast.makeText(getBaseContext(), "Nome: ", Toast.LENGTH_SHORT).show();
            } else {
                  imgBotao.setImageResource(R.drawable.menino);
                  Toast.makeText(getBaseContext(), "Nome: ", Toast.LENGTH_SHORT).show();
            }
    }

}
```

Em todos os views (componentes) criados até aqui, sempre definimos uma propriedade android:id. Essa identificação é usada para recuperar este componente no código Java através do método findViewById().

Com posse da instância do Button podemos configurar um listener para o evento de clique, usando o método setOnClickListener e passando por parâmetro uma instância do próprio objeto que desejamos, no caso inicial vamos usar somente o Button. Vamos repetir essa operação também para o ImageButton. Veja o código a seguir:

```
btnExibir.setOnClickListener(this);
imgBotao.setOnClickListener(this);
```

Evento do Button e do ImageButton

Uma forma de tratar eventos é através da interface OnClickListener. Na classe Activity a linha da classe fica:

```
public class PrincipalActivity extends Activity  implements OnClickListener {
```

Atenção: localizada em: `import android.view.View.OnClickListener;`

Quando definimos a interface OnClickListener temos que codificar o método onClick. Com esse método definimos o que acontecerá quando o evento de clique do Button e do ImageButon forem acionados.

O texto que aparece na tela é criado através de um Toast. Usamos o método estático makeToast() que recebe os seguintes parâmetros: um contexto, a mensagem e um inteiro que define a duração. O contexto normalmente é o atual, isto é, é a tela que desejamos exibir a mensagem. Normalmente é a tela que já está em exibição. Por isso escrevemos: getBaseContext(). Para o tempo de exibição usamos a constante LENGHT_SHORT. Por fim, para exibir a mensagem na tela do Android usamos o método show().

Evento do RadioGroup para atender os RadioButtons

Vamos acrescentar mais outro evento ao nosso projeto. Agora utilizaremos também a interface OnCheckedChangeListener. A primeira linha da classe fica:

```
public class PrincipalActivity extends Activity  implements OnClickListener,
OnCheckedChangeListener {
```

Quando definimos a interface `OnCheckedChangeListener` temos que codificar o método `onCheckedChanged`. Com esse método definimos o que acontecerá quando o evento de clique dos RadioButtons forem acionados. Veja que colocamos os radios dentro de um RadioGroup.

A ideia é associar uma imagem dentro do ImageButton em função do evento de clique nos RadioButtons que indicam o sexo. Sendo assim, precisamos copiar duas imagens que estão na pasta Arquivos (na pasta da aula) e colar na pasta drawble-hdpi. As imagens são: menino.jpg e menina.jpg.

Também o método findViewById() recupera o componente no código Java. Com posse da instância do RadioGroup podemos configurar um listener para o evento de clique usando a codificação:

```
radGrupo.setOnCheckedChangeListener(this);
```

No método `setOnCheckedChangeListener` passamos o parâmetro com a instância do próprio view.

O método onCreate() agora completo fica:

```
   @Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    btnExibir = (Button) findViewById(R.id.buttonExibir);
    edtNome = (EditText) findViewById(R.id.editTextNome);
    chkUm = (CheckBox) findViewById(R.id.checkBox1);
    chkDois = (CheckBox) findViewById(R.id.checkBox2);
    radMasculino = (RadioButton) findViewById(R.id.radioButtonMasculino);
    radFeminino = (RadioButton) findViewById(R.id.radioButtonFeminino);
    tgbUm = (ToggleButton) findViewById(R.id.toggleButton1);
    imgBotao = (ImageButton) findViewById(R.id.imageButtonUm);
    btnExibir.setOnClickListener(this);
    imgBotao.setOnClickListener(this);

    radGrupo = (RadioGroup) findViewById(R.id.radioGroupUm);
    radGrupo.setOnCheckedChangeListener(this);

  }
```

Temos também que adicionar o método onCheckedChanged.

```
public void onCheckedChanged(RadioGroup group, int checkedId) {
  RadioButton radBotao = (RadioButton) this.findViewById(checkedId);
  if (radBotao.getText().equals("Sexo Feminino")) {
              imgBotao.setImageResource(R.drawable.menina);
              Toast.makeText(getBaseContext(), "Nome: ", Toast.LENGTH_SHORT).show();
        } else {
              imgBotao.setImageResource(R.drawable.menino);
              Toast.makeText(getBaseContext(), "Nome: ", Toast.LENGTH_SHORT).show();
        }
}
```