PROGRAMAÇÃO PARA DISPOSITIVOS MÓVEIS

Uso de Imagens no Android

Professor: Danilo Giacobo

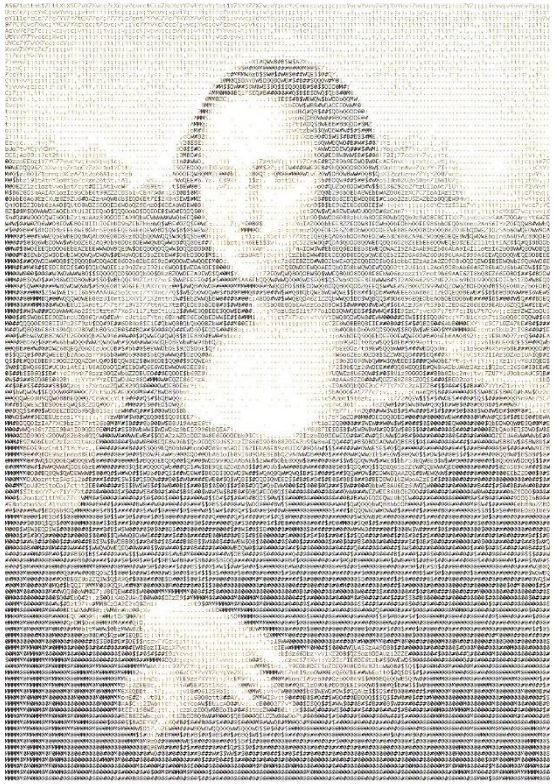
OBJETIVOS DA AULA

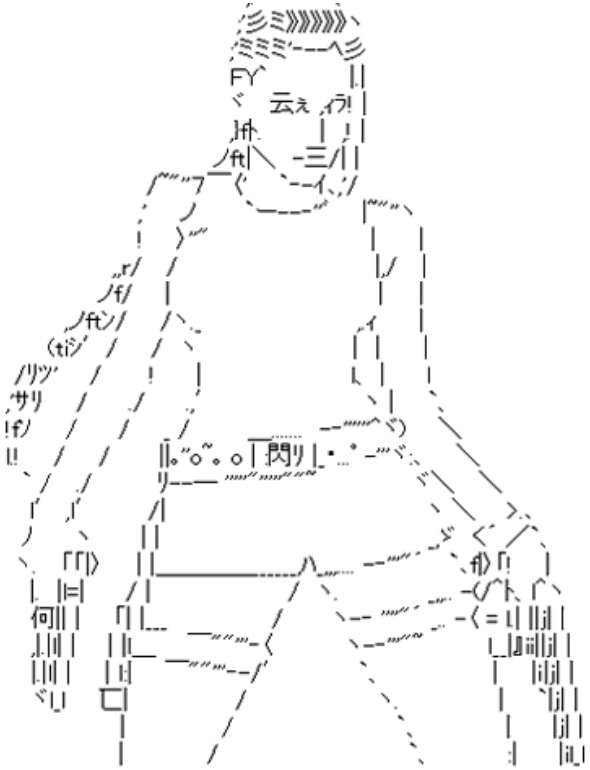
- Aprender a utilizar imagens em aplicações Android.
- Conhecer e utilizar o componente Gallery.
- Conhecer e utilizar o componente ImageView.
- Conhecer e utilizar o componente ImageSwitcher.
- Conhecer e utilizar o componente GridView.

INTRODUÇÃO

- Utilização de dados multimídia em programas computacionais é um dos recursos mais importantes ao longo da história da computação.
- Antigamente as animações e os vídeos eram desenvolvidos manualmente por meio de imagens ASCII.
- A reprodução de sons só era possível utilizando o *speaker* do computador.
- Atualmente, com a grande capacidade gráfica e computacional dos computadores pessoais, é possível usar e abusar dos recursos multimídias, adicionando facilmente imagens e vídeos nos programas de computador.

INTRODUÇÃO





INTRODUÇÃO

- As plataformas móveis, apesar de terem evoluído muito nos últimos anos, ainda possuem limitações que, quando não trabalhadas, podem inviabilizar o uso de imagens nos aplicativos móveis.
- Uma das limitações é o tamanho reduzido das telas dos dispositivos móveis.
- Outra limitação se refere ao formato da imagem compatível com os dispositivos móveis.
- Para trabalhar com imagens no Android, uma série de componentes foi criada, sendo apresentados nesta aula detalhes dos componentes Gallery, ImageView, ImageSwitcher e GridView.

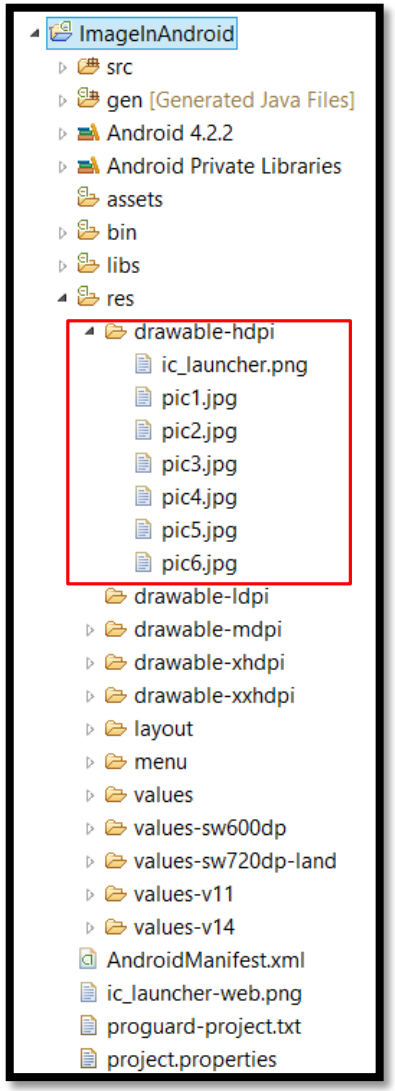
COMPONENTES GALLERY E IMAGEVIEW

- No nosso primeiro exemplo iremos criar um projeto com o nome **ImageInAndroid**. Posteriormente, iremos editá-lo para fazer uso de imagens. Após criar o projeto, o próximo passo é modificar o arquivo **activity_main.xml**, adicionando o componente Gallery, conforme apresentado abaixo:

```
1  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2      xmlns:tools="http://schemas.android.com/tools"
3      android:layout_width="fill_parent"
4      android:layout_height="fill_parent"
5      android:orientation="vertical"
6      android:paddingBottom="@dimen/activity_vertical_margin"
7      android:paddingLeft="@dimen/activity_horizontal_margin"
8      android:paddingRight="@dimen/activity_horizontal_margin"
9      android:paddingTop="@dimen/activity_vertical_margin"
10     tools:context=".MainActivity" >
11
12     <Gallery
13         android:id="@+id/gallery1"
14         android:layout_width="fill_parent"
15         android:layout_height="wrap_content" />
16
17 </LinearLayout>
```

COMPONENTES GALLERY E IMAGEVIEW

- O próximo passo é acessar a pasta onde o projeto foi salvo e na subpasta *res*, criar uma pasta chamada *drawable*, onde devem ser inseridas as imagens apresentadas no projeto. Após esses passos, a estrutura do projeto deve ficar parecida com o da figura ao lado.



COMPONENTES GALLERY E IMAGEVIEW

Dica: Adicionando imagens

Se a sua classe *R* não for atualizada com os novos recursos, deve-se clicar com o botão direito no projeto e escolher a opção **Limpar e Construir**. Esse comando fará a atualização dos arquivos de configuração do projeto, de forma que passem a reconhecer as imagens da pasta *drawable*.

Outra dica importante é não utilizar imagens muito grandes, com resoluções superiores a 500x500, pois isso dificultará o processo de apresentação da imagem.

O último passo para utilizar o componente Gallery é modificar o arquivo **MainActivity.java**, deixando-o conforme código apresentado no próximo slide.

COMPONENTES GALLERY E IMAGEVIEW

```
1  package pm25s.aula11.imageinandroid;
2
3  import android.app.Activity;
4  import android.content.Context;
5  import android.os.Bundle;
6  import android.view.View;
7  import android.view.ViewGroup;
8  import android.widget.AdapterView;
9  import android.widget.AdapterView.OnItemClickListener;
10 import android.widget.BaseAdapter;
11 import android.widget.Gallery;
12 import android.widget.ImageView;
13 import android.widget.Toast;
14
15 @SuppressWarnings("deprecation")
16 public class MainActivity extends Activity {
17
18     @Override
19     protected void onCreate(Bundle savedInstanceState) {
20         super.onCreate(savedInstanceState);
21         setContentView(R.layout.activity_main);
22
23         Gallery gallery1 = (Gallery) findViewById(R.id.gallery1);
24         gallery1.setAdapter(new ImageAdapter(this));
25
26         gallery1.setOnItemClickListener(new OnItemClickListener() {
27             public void onItemClick(AdapterView<?> parent, View v, int position, long id) {
28                 Toast.makeText(getBaseContext(), "Selecionou: " + (position + 1), Toast.LENGTH_SHORT).show();
29             }
30         });
31
32     }
```

COMPONENTES GALLERY E IMAGEVIEW

```
34    public class ImageAdapter extends BaseAdapter {
35        private Integer[] imageIDs = {
36                R.drawable.pic1,
37                R.drawable.pic2,
38                R.drawable.pic3,
39                R.drawable.pic4,
40                R.drawable.pic5,
41                R.drawable.pic6 };
42
43        private Context context;
44
45        public ImageAdapter(Context c) {
46            context = c;
47        }
48
49        public int getCount() {
50            return imageIDs.length;
51        }
52
53        public Object getItem(int position) {
54            return position;
55        }
56
57        public long getItemId(int id) {
58            return id;
59        }
60
61        public View getView(int position, View convertView, ViewGroup parent) {
62            ImageView imageView = new ImageView(context);
63            imageView.setImageResource(imageIDs[position]);
64            return imageView;
65        }
66    }
67 }
```

COMPONENTES GALLERY E IMAGEVIEW

- Este aplicativo, quando executado, apresentará uma imagem após a outra, no formato de galeria, conforme mostrado nas imagens ao lado.
- No exemplo apresentado, as imagens ainda estão com seu tamanho natural. Apesar se não ser visível na figura, ImageView possui uma transparência nas laterais e isso acontece no estilo padrão do componente.

COMPONENTES GALLERY E IMAGEVIEW

- Iremos alterar algumas propriedades de ImageView para melhorar a apresentação das imagens no componente Gallery. Adicione mais estas três linhas de código ao corpo de **getView** e veja no aplicativo as mudanças ocorridas.

```
public View getView(int position, View convertView, ViewGroup parent) {
    ImageView imageView = new ImageView(context);
    imageView.setImageResource(imageIDs[position]);
    imageView.setScaleType(ImageView.ScaleType.FIT_XY);
    imageView.setLayoutParams(new Gallery.LayoutParams(180, 180));
    imageView.setBackgroundResource(android.R.drawable.alert_light_frame);
    return imageView;
}
```

IMAGEM SELECIONADA NO CENTRO DA TELA

- Será desenvolvido agora um código para apresentar a imagem selecionada no componente Gallery no centro da tela, utilizando para isso, o componente ImageView., incluindo
- Como se trata da inclusão de um novo componente visual na tela do aplicativo, é necessário modificar o arquivo *activity_main.xml* conforme apresentado abaixo. Inclua as linhas abaixo logo após a declaração da *view* Gallery.

```
<ImageView
    android:id="@+id/image1"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:contentDescription="@string/mensagem" />
```

strings.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3
4     <string name="app_name">ImageInAndroid</string>
5     <string name="action_settings">Settings</string>
6     <string name="hello_world">Hello world!</string>
7     <string name="mensagem">Imagem selecionada</string>
8
9 </resources>
```

Dica: android:scaleType="fitXY"

IMAGEM SELECIONADA NO CENTRO DA TELA

- No arquivo *MainActivity.java*, é necessário alterar o código do método **onItemClick()**, conforme apresentado abaixo:

```
gallery1.setOnItemClickListener(new OnItemClickListener() {
    public void onItemClick(AdapterView<?> parent, View v, int position, long id) {
        //Toast.makeText(getBaseContext(), "Selecionou: " + (position + 1), Toast.LENGTH_SHORT).show();
        Integer [] imageIDs = {
                R.drawable.pic1,
                R.drawable.pic2,
                R.drawable.pic3,
                R.drawable.pic4,
                R.drawable.pic5,
                R.drawable.pic6
        };

        ImageView imageView = (ImageView) findViewById(R.id.image1);
        imageView.setImageResource(imageIDs[position]);
    }
});
```

O COMPONENTE IMAGESWITCHER

- A troca de imagens realizadas no exemplo anterior também é conseguida utilizando o componente ImageSwitcher, mas com uma grande vantagem: esse componente permite configurar efeitos 3D na troca de imagens.
- Para desenvolver a interface do aplicativo, modifique o arquivo *activity_main.xml*, conforme o conteúdo abaixo:

```
1  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
2      xmlns:tools="http://schemas.android.com/tools"
3      android:layout_width="fill_parent"
4      android:layout_height="fill_parent"
5      android:orientation="vertical"
6      tools:context=".ImageSwitcherActivity" >
7
8      <ImageSwitcher
9          android:id="@+id/switcher1"
10         android:layout_width="fill_parent"
11         android:layout_height="fill_parent" />
```

```
12
13     <Gallery
14         android:id="@+id/gallery1"
15         android:layout_width="fill_parent"
16         android:layout_height="wrap_content"
17         android:layout_alignParentRight="true"
18         android:layout_alignParentBottom="true" />
19
20 </RelativeLayout>
```

O COMPONENTE IMAGESWITCHER

- O próximo passo para desenvolver o aplicativo é codificar a classe *MainActivity.java*, a qual é apresentada neste e nos próximos slides.

```
1  package pm25s.aula11.imageinandroid;
2
3  import android.app.Activity;
4  import android.content.Context;
5  import android.os.Bundle;
6  import android.view.View;
7  import android.view.ViewGroup;
8  import android.view.ViewGroup.LayoutParams;
9  import android.view.animation.AnimationUtils;
10 import android.widget.AdapterView;
11 import android.widget.AdapterView.OnItemClickListener;
12 import android.widget.BaseAdapter;
13 import android.widget.Gallery;
14 import android.widget.ImageSwitcher;
15 import android.widget.ImageView;
16 import android.widget.ViewSwitcher.ViewFactory;
17
18 @SuppressWarnings("deprecation")
19 public class ImageSwitcherActivity extends Activity implements ViewFactory {
20
21     private ImageSwitcher switcher;
22
23     private Integer[] imageIDs = {
24             R.drawable.pic1,
25             R.drawable.pic2,
26             R.drawable.pic3,
27             R.drawable.pic4,
28             R.drawable.pic5,
29             R.drawable.pic6 };
```

O COMPONENTE IMAGESWITCHER

```
31      @Override
32      protected void onCreate(Bundle savedInstanceState) {
33          super.onCreate(savedInstanceState);
34          setContentView(R.layout.activity_image_switcher);
35
36          switcher = (ImageSwitcher) findViewById(R.id.switcher1);
37          switcher.setFactory(this);
38          switcher.setInAnimation(AnimationUtils.loadAnimation(this, android.R.anim.fade_in));
39          switcher.setOutAnimation(AnimationUtils.loadAnimation(this, android.R.anim.fade_out));
40
41          Gallery gallery1 = (Gallery) findViewById(R.id.gallery1);
42          gallery1.setAdapter(new ImageAdapter(this));
43
44          gallery1.setOnItemClickListener(new OnItemClickListener() {
45              public void onItemClick(AdapterView<?> parent, View v, int position, long id) {
46                  switcher.setImageResource(imageIDs[position]);
47              }
48          });
49      }
50
51      @Override
52      public View makeView() {
53          ImageView imageView = new ImageView(this);
54          imageView.setBackgroundColor(0xFF000000);
55          imageView.setScaleType(ImageView.ScaleType.FIT_XY);
56          imageView.setLayoutParams(new ImageSwitcher.LayoutParams(LayoutParams.FILL_PARENT, LayoutParams.FILL_PARENT));
57          return imageView;
58      }
```

O COMPONENTE IMAGESWITCHER

```
60    public class ImageAdapter extends BaseAdapter {
61        private Integer[] imageIDs = {
62                R.drawable.pic1,
63                R.drawable.pic2,
64                R.drawable.pic3,
65                R.drawable.pic4,
66                R.drawable.pic5,
67                R.drawable.pic6 };
68
69        private Context context;
70
71        public ImageAdapter(Context c) {
72            context = c;
73        }
74
75        public int getCount() {
76            return imageIDs.length;
77        }
78
79        public Object getItem(int position) {
80            return position;
81        }
82
83        public long getItemId(int id) {
84            return id;
85        }
86
87        public View getView(int position, View convertView, ViewGroup parent) {
88            ImageView imageView = new ImageView(context);
89            imageView.setImageResource(imageIDs[position]);
90            imageView.setScaleType(ImageView.ScaleType.FIT_XY);
91            imageView.setLayoutParams(new Gallery.LayoutParams(120, 90));
92            imageView.setBackgroundResource(android.R.drawable.alert_light_frame);
93            return imageView;
94        }
95    }
96 }
```

GRIDVIEW PARA A APRESENTAÇÃO DE IMAGENS

- Para finalizar a aula, será utilizado um componente GridView para a organização de várias imagens na tela do dispositivo móvel. Para sua utilização é necessário alterar o arquivo XML da interface gráfica conforme o conteúdo abaixo:

```
<GridView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/gridview"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:numColumns="auto_fit"
    android:verticalSpacing="10dp"
    android:horizontalSpacing="10dp"
    android:columnWidth="90dp"
    android:stretchMode="columnWidth"
    android:gravity="center"
    tools:context=".GridViewActivity" >

</GridView>
```

GRIDVIEW PARA A APRESENTAÇÃO DE IMAGENS

```java
package pm25s.aula11.imageinandroid;

import android.app.Activity;
import android.content.Context;
import android.os.Bundle;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.BaseAdapter;
import android.widget.GridView;
import android.widget.ImageView;
import android.widget.Toast;

public class GridViewActivity extends Activity {
    Integer[] imageIDs = {
            R.drawable.pic1,
            R.drawable.pic2,
            R.drawable.pic3,
            R.drawable.pic4,
            R.drawable.pic5,
            R.drawable.pic6 };

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_grid_view);

        GridView gridview = (GridView) findViewById(R.id.gridview);
        gridview.setAdapter(new ImageAdapter(this));

        gridview.setOnItemClickListener(new OnItemClickListener() {
            public void onItemClick(AdapterView<?> parent, View v, int position, long id) {
                Toast.makeText(getBaseContext(), "pic" + (position + 1) + " selected.", Toast.LENGTH_SHORT).show();
            }
        });
    }
```

```java
    public class ImageAdapter extends BaseAdapter {
        private Integer[] imageIDs = {
                R.drawable.pic1,
                R.drawable.pic2,
                R.drawable.pic3,
                R.drawable.pic4,
                R.drawable.pic5,
                R.drawable.pic6 };

        private Context context;

        public ImageAdapter(Context c) {
            context = c;
        }

        public int getCount() {
            return imageIDs.length;
        }

        public Object getItem(int position) {
            return position;
        }

        public long getItemId(int id) {
            return id;
        }
```

GRIDVIEW PARA A APRESENTAÇÃO DE IMAGENS

```
66        public View getView(int position, View convertView, ViewGroup parent) {
67            ImageView imageView;
68
69            if(convertView == null) {
70                imageView = new ImageView(context);
71                imageView.setLayoutParams(new GridView.LayoutParams(90, 90));
72                imageView.setScaleType(ImageView.ScaleType.CENTER_CROP);
73            } else {
74                imageView = (ImageView) convertView;
75            }
76
77            imageView.setImageResource(imageIDs[position]);
78
79            return imageView;
80        }
81    }
82 }
```