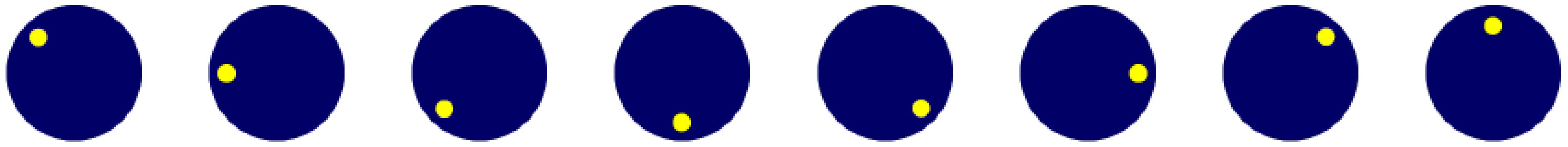
Graphics Animation

Introduction

- The android platform basically supports two types of animations: 'frame by frame animation' and 'tween animation'.

Frame By Frame Animation

- This is the simplest animation technique. Series of images that create the animation illusion.
- Sharing the same base name between all image files shall contribute to the clarity of our code (e.g. ball_1.jpg, ball_2.jpg, ball_3.jpg etc.).



Frame By Frame Animation

❖ We should now create an `ImageView` control in which the images will be displayed.

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

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<ImageView android:id="@+id/ImageView01"
android:layout_width="wrap_content"
android:layout_height="wrap_content"></ImageView>

<Button android:id="@+id/Button01" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:text="@+id/Press"></Button>

</LinearLayout>
```

Frame By Frame Animation

- Instantiating `AnimationDrawable` we shall get an object that represents an animation.
- We can instantiate that class by using the following XML element `<animation-list>.` The XML document should be saved within the the `drawable` folder. After all, it is a new drawable resource.

Frame By Frame Animation

- Object of the `AnimationDrawable` type holds a list of `Drawable` resources (e.g. images) and renders them at specified intervals.

Frame By Frame Animation

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

<animation-list
xmlns:android="http://schemas.android.com/apk/res/android"
android:oneshot="false">

        <item android:drawable="@drawable/ball_1" android:duration="50" />
        <item android:drawable="@drawable/ball_2" android:duration="50" />
        <item android:drawable="@drawable/ball_3" android:duration="50" />
        <item android:drawable="@drawable/ball_4" android:duration="50" />
        <item android:drawable="@drawable/ball_5" android:duration="50" />
        <item android:drawable="@drawable/ball_6" android:duration="50" />
        <item android:drawable="@drawable/ball_7" android:duration="50" />
        <item android:drawable="@drawable/ball_8" android:duration="50" />

</animation-list>
```

frames_animation.xml

Frame By Frame Animation

- The next step would be setting the `Drawable` object (the one represented by the animation-list XML element) as the background resource of the `ImageView` object.

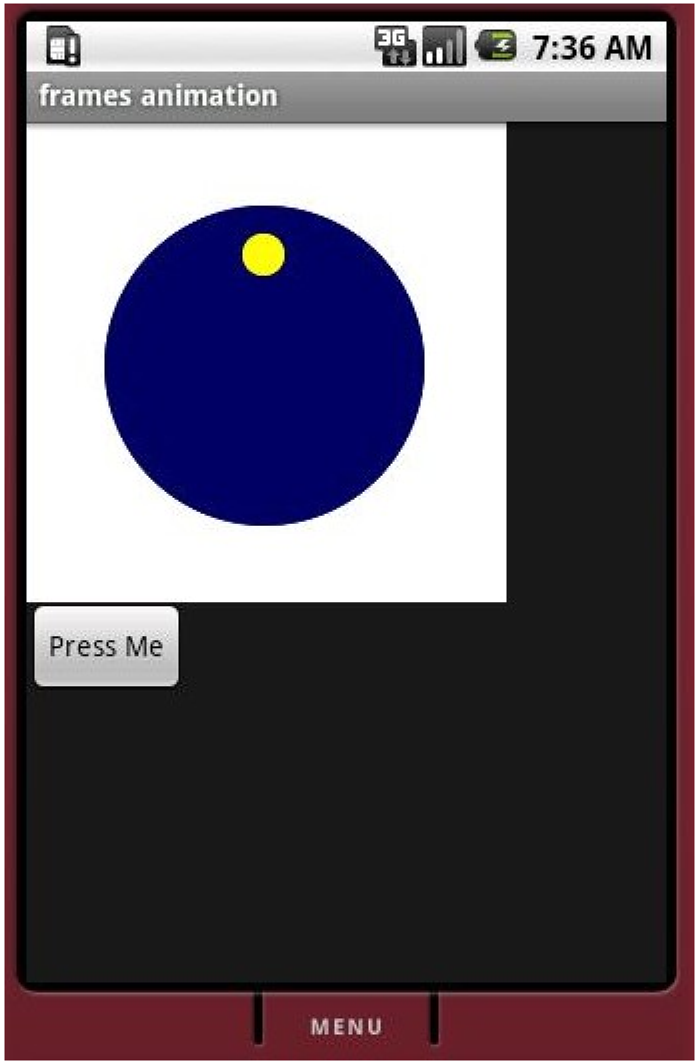
Frame By Frame Animation

```
package com.abelski.samples;

import android.os.Bundle;
import android.widget.Button;
import android.widget.ImageView;
import android.app.Activity;
import android.graphics.drawable.AnimationDrawable;
import android.view.View;
import android.view.View.OnClickListener;

public class FramesAnimationActivity extends Activity
{
    @Override
    public void onCreate(Bundle bundle)
    {
        super.onCreate(bundle);
        setContentView(R.layout.main);
        Button bt = (Button)this.findViewById(R.id.Button01);
        bt.setText("Press Me");
```

Frame By Frame Animation

```
            bt.setOnClickListener(new OnClickListener()
                {
                    public void onClick(View v)
                    {
                        ImageView imgView =
                            (ImageView)findViewById(R.id.ImageView01);
                        imgView.setVisibility(ImageView.VISIBLE);
                        imgView.setBackgroundResource(
                            R.drawable.frames_animation);
                        AnimationDrawable animation =
                            (AnimationDrawable)imgView.getBackground();
                            if(animation.isRunning())
                            {
                                animation.stop();
                            }
                            else
                            {
                                animation.stop();
                                animation.start();
                            }
                    }
                }                                                    );
        }
    }
```

Frame By Frame Animation



Tween Animation

- The tween animation technique is been used to add animation effects to views such as `GridView` and `ListView`.
- Each view has a matrix that maps its content to the screen. Changing that matrix can achieve various visual effects.
- We can add an XML document to the `res/anim` folder. That XML document will define the animation.

Tween Animation

- Each animation we create is represented as an object of a class.... these classes belong to the `android.view.animation` package.
- The API documentation lists the relevant available XML elements for each one of these classes.

Tween Animation

Lists can be populated with data from android data sources with cursors or with simple arrays. This code sample uses a simple array.

```
package com.abelski;

import android.app.ListActivity;
import android.os.Bundle;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.ArrayAdapter;
import android.widget.ListView;

public class ListViewActivityDemo extends ListActivity
{
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setListAdapter(new ArrayAdapter<String>(this,
                android.R.layout.simple_list_item_1 , strings));
```

The `android.R.layout.simple_list_item_1` resource is a predefined one. Please note that android.R is not the R class of your project.

Tween Animation

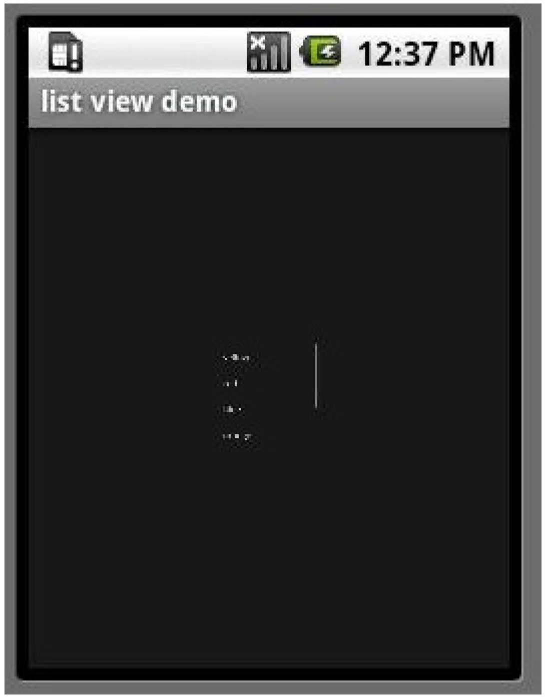
```
        ListView list = this.getListView();
        Animation animation =
            AnimationUtils.loadAnimation(this,com.abelski.R.anim.scale);
        animation.setRepeatCount(1);
        list.startAnimation(animation);
    }



    private String[] strings = {
            "yellow", "red", "blue", "orange", "black", "gray", "purple"
    };
}
```

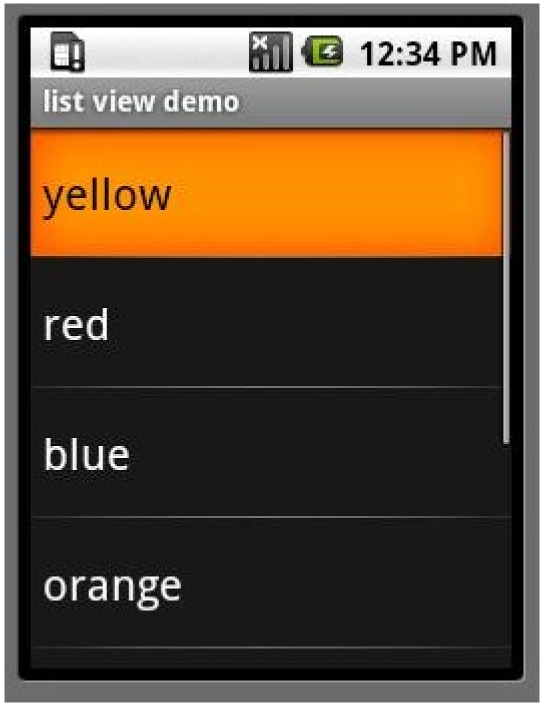
Tween Animation

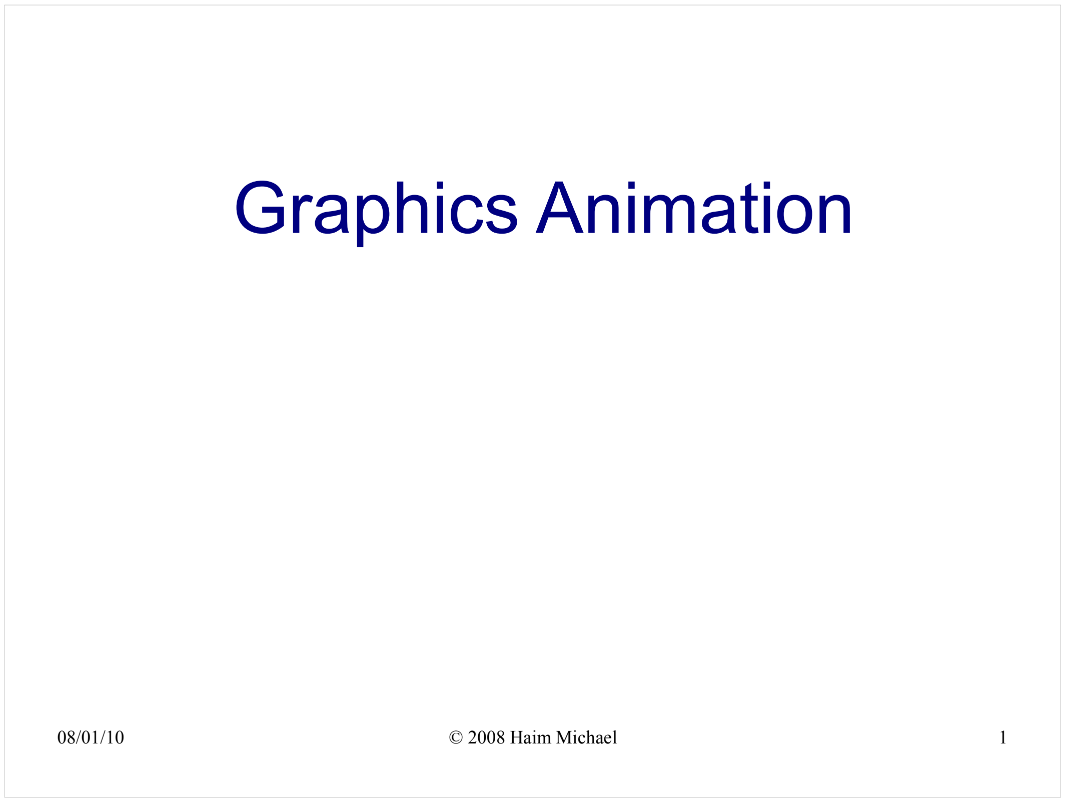
```
<?xml version="1.0" encoding="utf-8"?>
<set
xmlns:android="http://schemas.android.com/apk/res/android"

android:interpolator="@android:anim/accelerate_interpolator">

    <scale
            android:fromXScale="0.2"
            android:toXScale="1.0"
            android:fromYScale="0.2"
            android:toYScale="1.0"
            android:duration="600"
            android:pivotX="50%"
            android:pivotY="50%"
            android:startOffset="200"
            />

</set>
```

scale.xml should be saved within res/anim folder and will be treated as a new resource of type 'anim'. The ID of this new resource will be held within a static variable defined within 'anim' static inner class within R.

Tween Animation





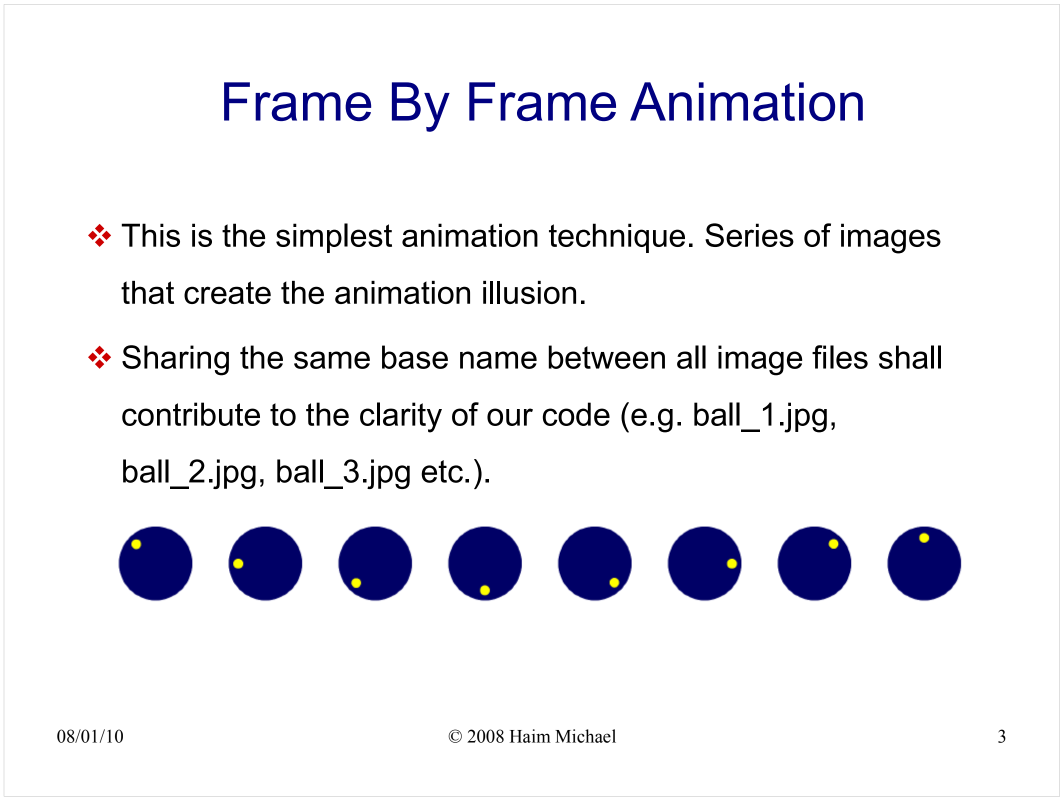


Graphics Animation

08/01/10 © 2008 Haim Michael 1

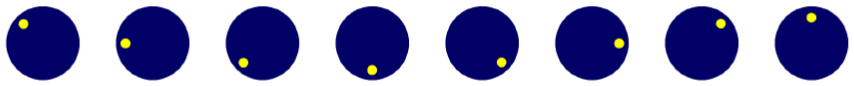
Introduction

- ❖ The android platform basically supports two types of animations: 'frame by frame animation' and 'tween animation'.



Frame By Frame Animation

- This is the simplest animation technique. Series of images that create the animation illusion.
- Sharing the same base name between all image files shall contribute to the clarity of our code (e.g. ball_1.jpg, ball_2.jpg, ball_3.jpg etc.).

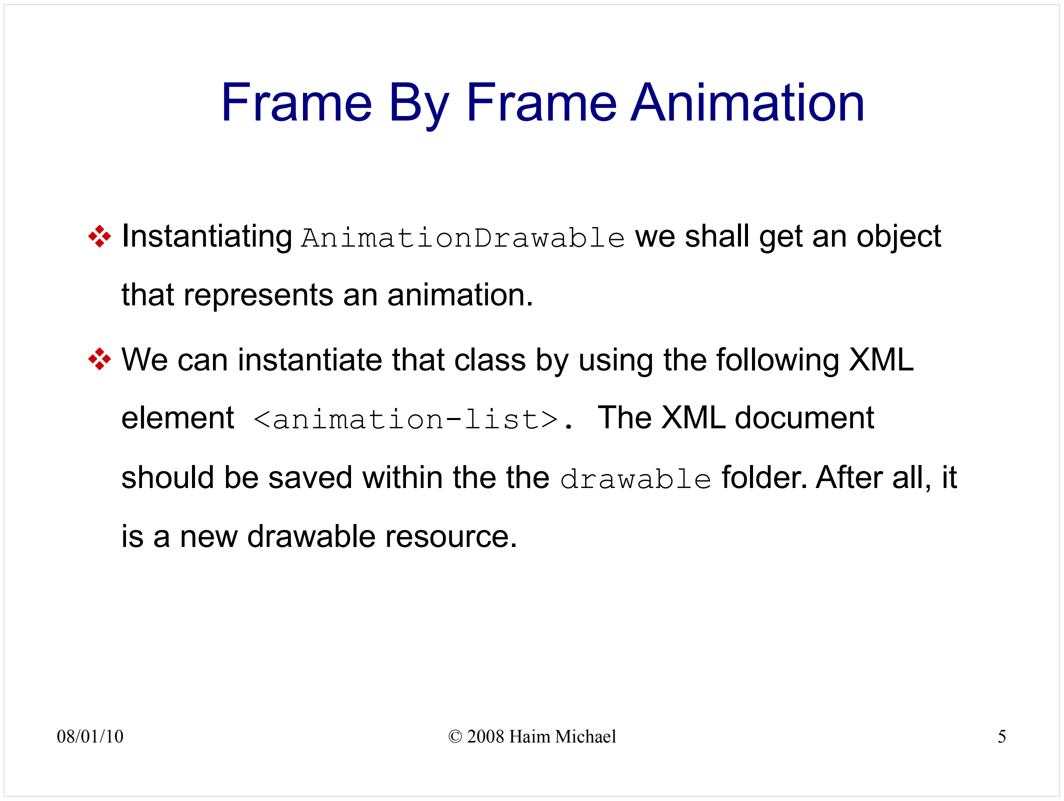


08/01/10 © 2008 Haim Michael 3

Frame By Frame Animation

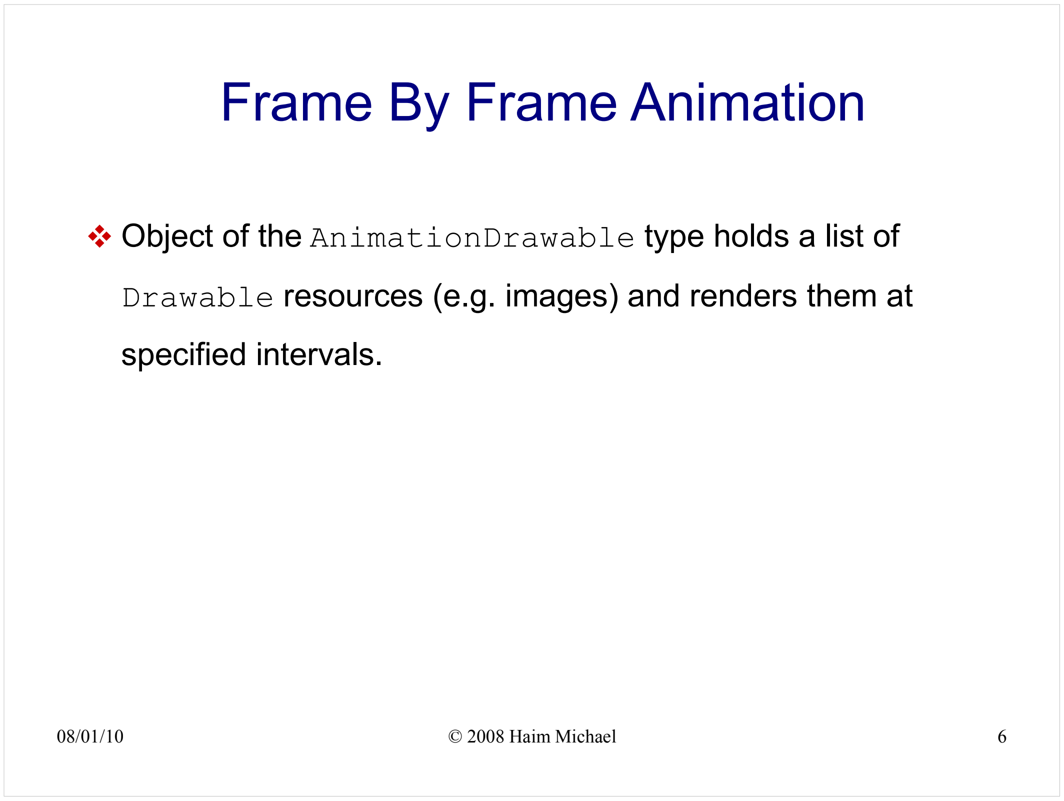
- We should now create an `ImageView` control in which the images will be displayed.

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

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<ImageView android:id="@+id/ImageView01"
android:layout_width="wrap_content"
android:layout_height="wrap_content"></ImageView>

<Button android:id="@+id/Button01" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:text="@+id/Press"></Button>

</LinearLayout>
```



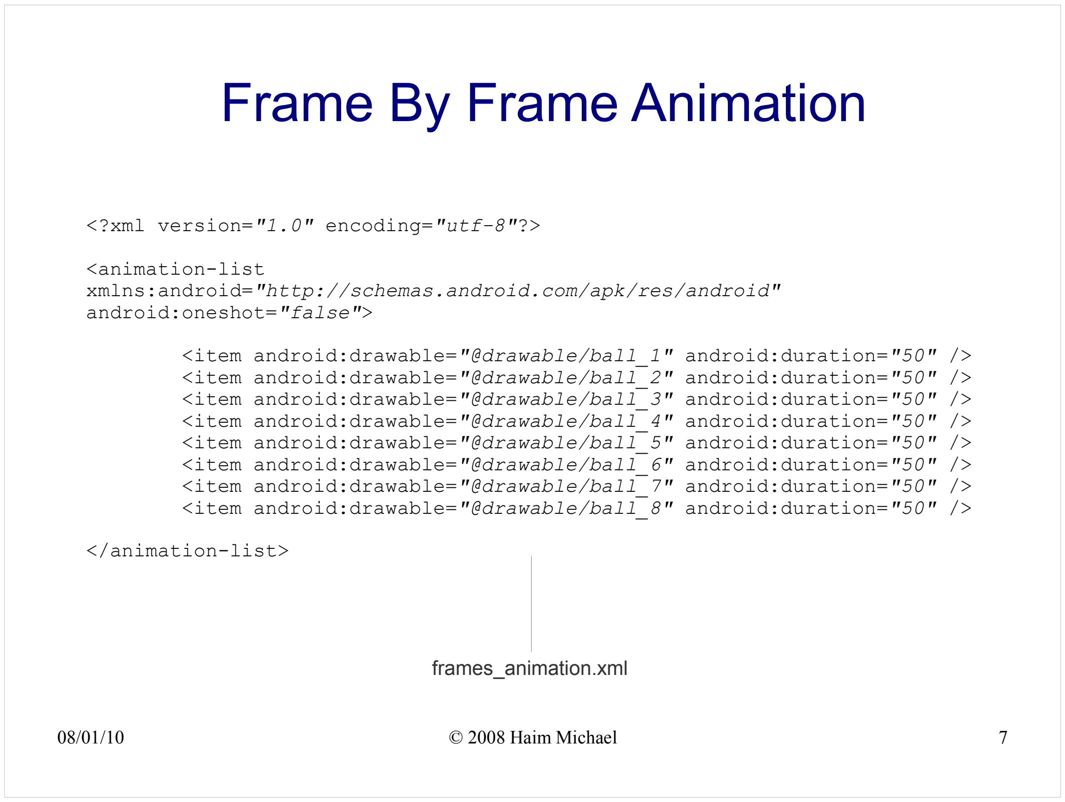
Frame By Frame Animation

- Instantiating `AnimationDrawable` we shall get an object that represents an animation.
- We can instantiate that class by using the following XML element `<animation-list>`. The XML document should be saved within the the `drawable` folder. After all, it is a new drawable resource.



Frame By Frame Animation

- Object of the `AnimationDrawable` type holds a list of `Drawable` resources (e.g. images) and renders them at specified intervals.



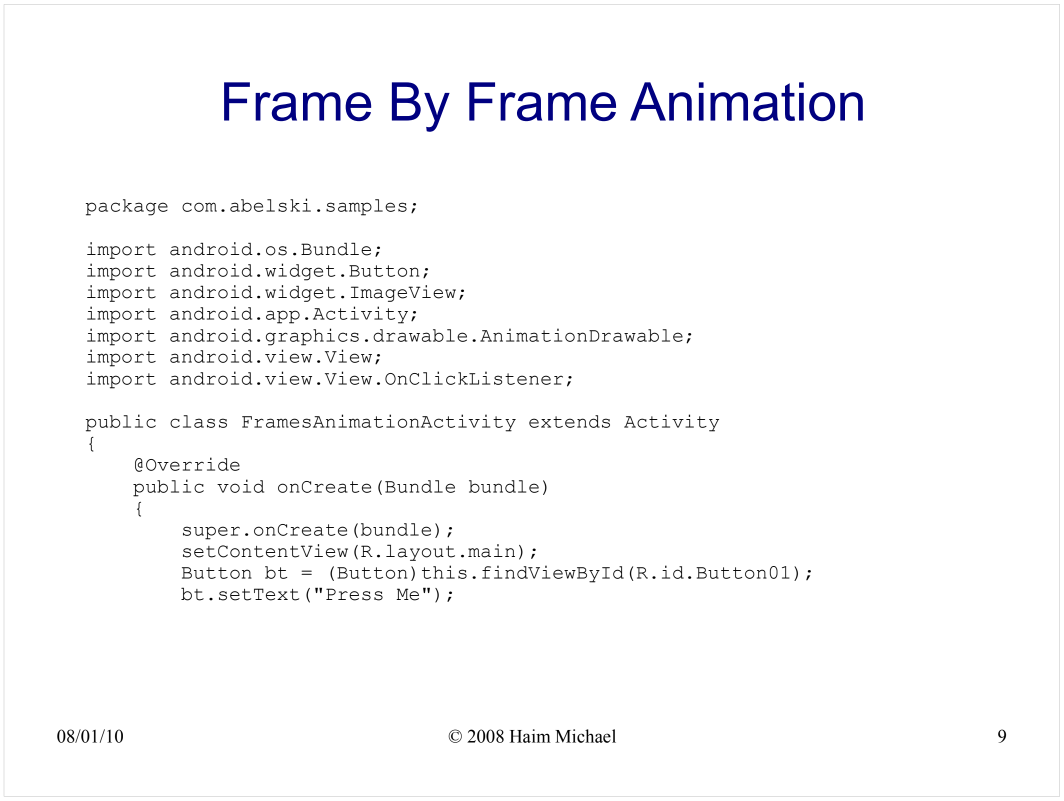
Frame By Frame Animation

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

<animation-list
xmlns:android="http://schemas.android.com/apk/res/android"
android:oneshot="false">

        <item android:drawable="@drawable/ball_1" android:duration="50" />
        <item android:drawable="@drawable/ball_2" android:duration="50" />
        <item android:drawable="@drawable/ball_3" android:duration="50" />
        <item android:drawable="@drawable/ball_4" android:duration="50" />
        <item android:drawable="@drawable/ball_5" android:duration="50" />
        <item android:drawable="@drawable/ball_6" android:duration="50" />
        <item android:drawable="@drawable/ball_7" android:duration="50" />
        <item android:drawable="@drawable/ball_8" android:duration="50" />

</animation-list>
```

frames_animation.xml

Frame By Frame Animation

- The next step would be setting the `Drawable` object (the one represented by the animation-list XML element) as the background resource of the `ImageView` object.



Frame By Frame Animation

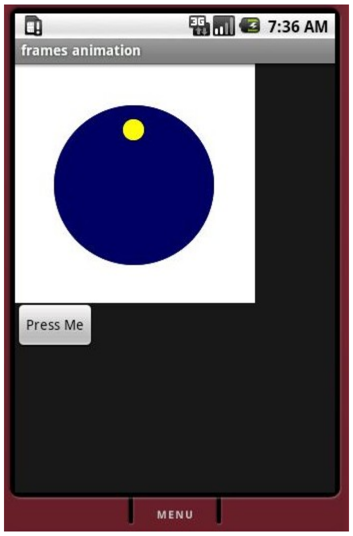
```
package com.abelski.samples;

import android.os.Bundle;
import android.widget.Button;
import android.widget.ImageView;
import android.app.Activity;
import android.graphics.drawable.AnimationDrawable;
import android.view.View;
import android.view.View.OnClickListener;

public class FramesAnimationActivity extends Activity
{
    @Override
    public void onCreate(Bundle bundle)
    {
        super.onCreate(bundle);
        setContentView(R.layout.main);
        Button bt = (Button)this.findViewById(R.id.Button01);
        bt.setText("Press Me");
```

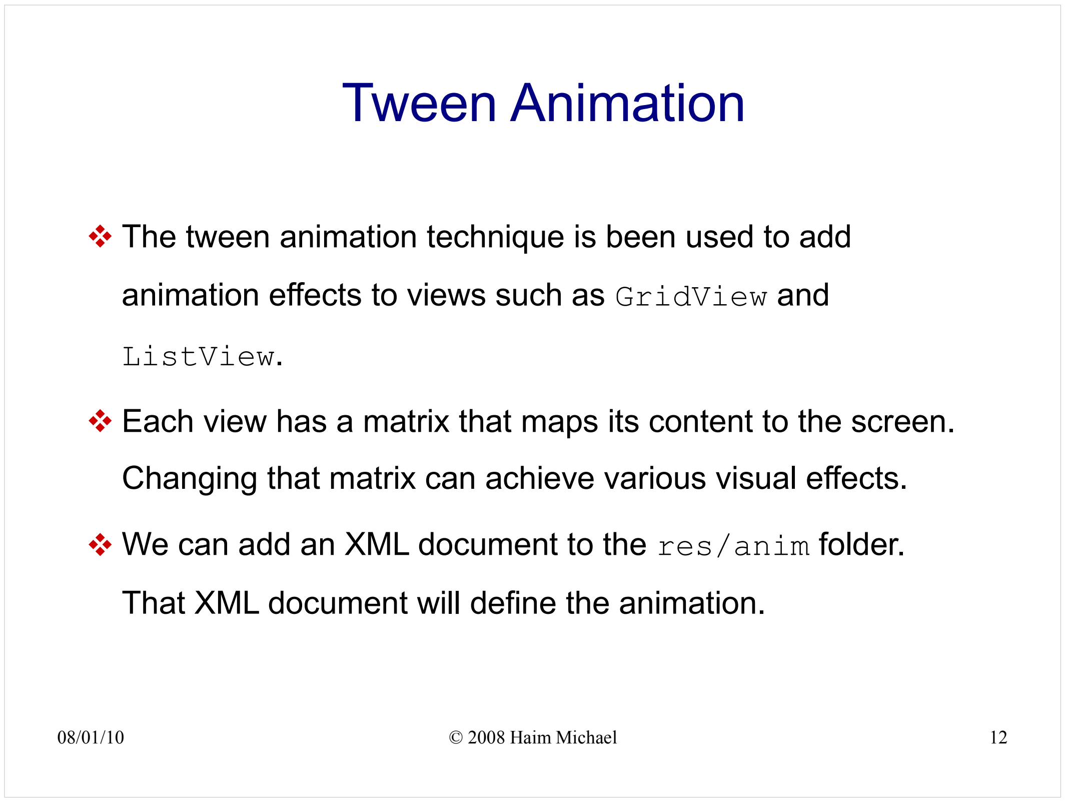
08/01/10 © 2008 Haim Michael 9

Frame By Frame Animation

```
        bt.setOnClickListener(new OnClickListener()
            {
                public void onClick(View v)
                {
                    ImageView imgView =
                        (ImageView)findViewById(R.id.ImageView01);
                    imgView.setVisibility(ImageView.VISIBLE);
                    imgView.setBackgroundResource(
                        R.drawable.frames_animation);
                    AnimationDrawable animation =
                        (AnimationDrawable)imgView.getBackground();
                        if(animation.isRunning())
                        {
                            animation.stop();
                        }
                        else
                        {
                            animation.stop();
                            animation.start();
                        }
                }
            }                                               );
    }
}
```

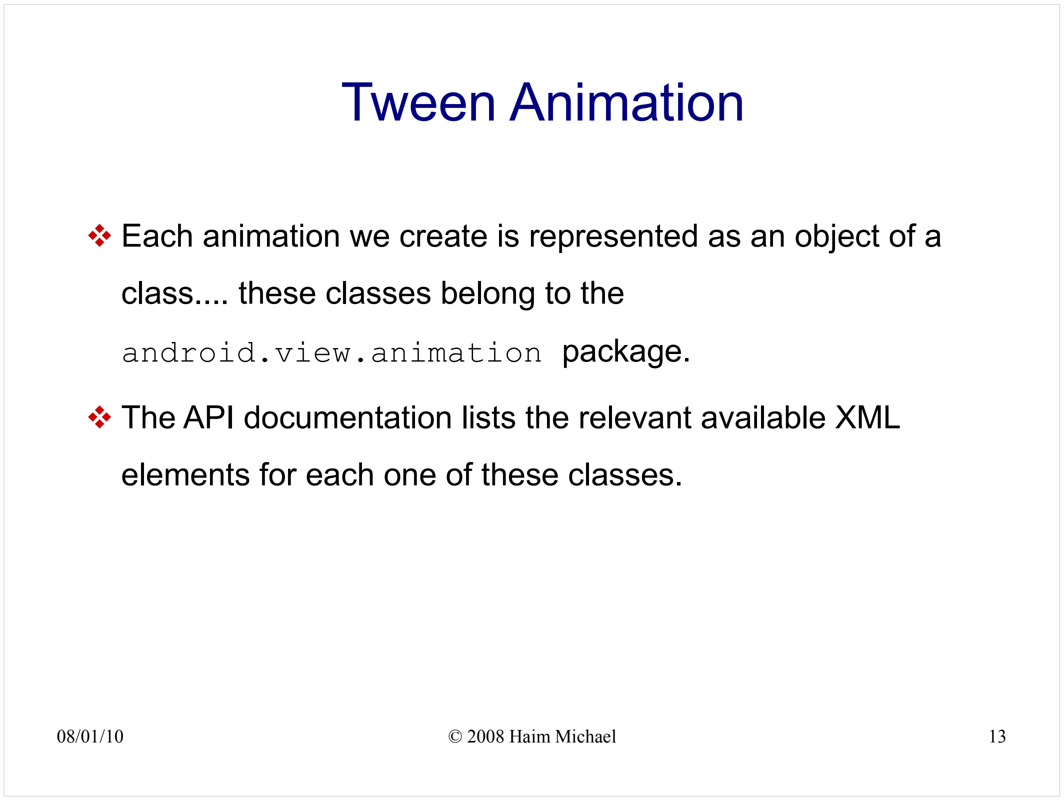
Frame By Frame Animation





Tween Animation

- The tween animation technique is been used to add animation effects to views such as `GridView` and `ListView`.
- Each view has a matrix that maps its content to the screen. Changing that matrix can achieve various visual effects.
- We can add an XML document to the `res/anim` folder. That XML document will define the animation.



Tween Animation

- Each animation we create is represented as an object of a class.... these classes belong to the `android.view.animation` package.
- The API documentation lists the relevant available XML elements for each one of these classes.

Tween Animation

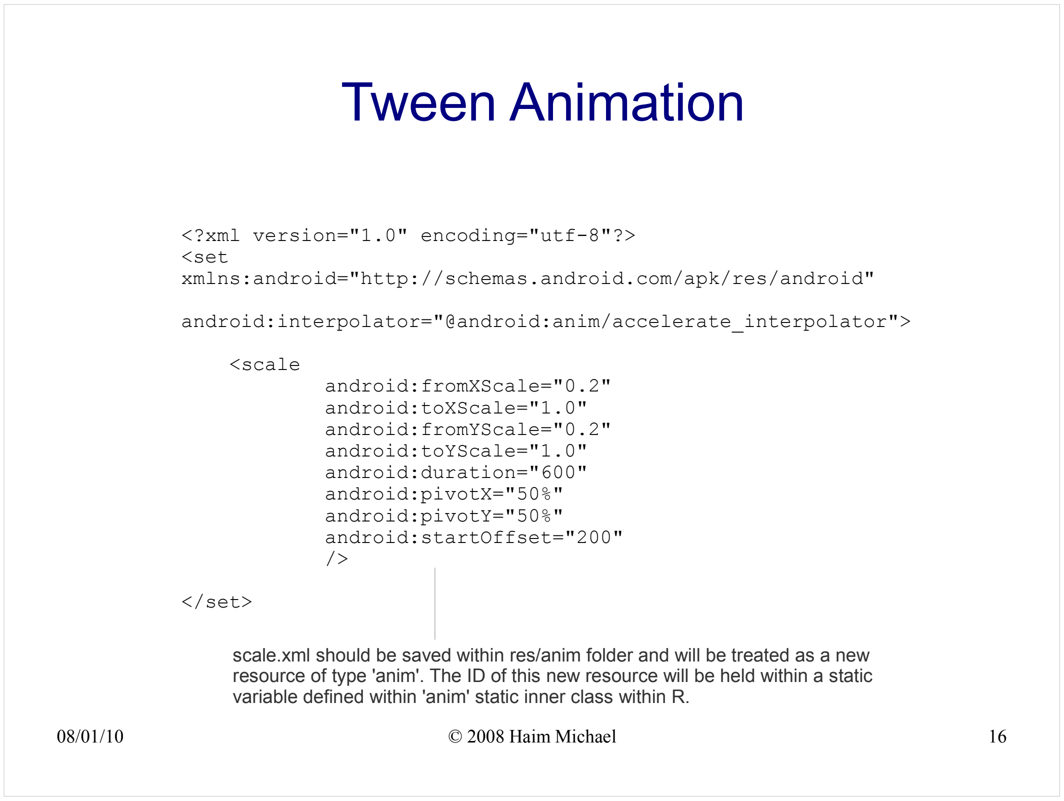
Lists can be populated with data from android data sources with cursors or with simple arrays. This code sample uses a simple array.

```
package com.abelski;

import android.app.ListActivity;
import android.os.Bundle;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.ArrayAdapter;
import android.widget.ListView;

public class ListViewActivityDemo extends ListActivity
{
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setListAdapter(new ArrayAdapter<String>(this,
                android.R.layout.simple_list_item_1 , strings));
```

The `android.R.layout.simple_list_item_1` resource is a predefined one. Please note that android.R is not the R class of your project.

Tween Animation

```
            ListView list = this.getListView();
            Animation animation =
                AnimationUtils.loadAnimation(this,com.abelski.R.anim.scale);
            animation.setRepeatCount(1);
            list.startAnimation(animation);
        }


        private String[] strings = {
                "yellow", "red", "blue", "orange", "black", "gray", "purple"
        };
    }
```

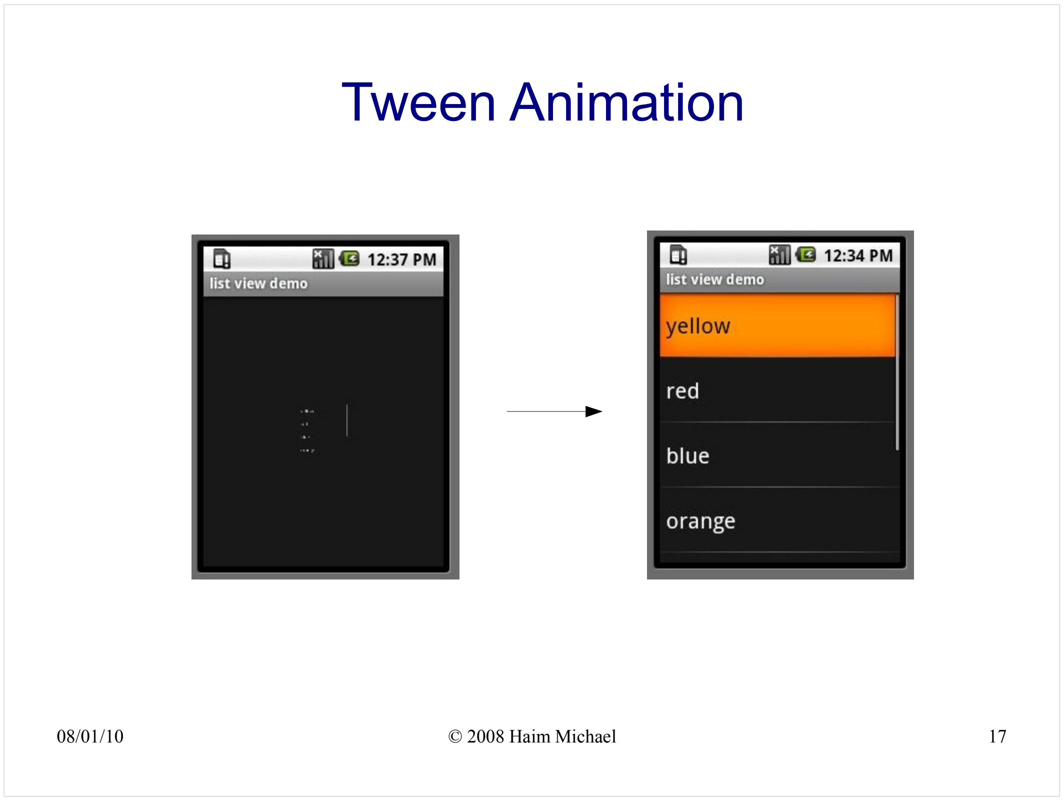


Tween Animation

```
<?xml version="1.0" encoding="utf-8"?>
<set
xmlns:android="http://schemas.android.com/apk/res/android"

android:interpolator="@android:anim/accelerate_interpolator">

    <scale
            android:fromXScale="0.2"
            android:toXScale="1.0"
            android:fromYScale="0.2"
            android:toYScale="1.0"
            android:duration="600"
            android:pivotX="50%"
            android:pivotY="50%"
            android:startOffset="200"
            />

</set>
```

scale.xml should be saved within res/anim folder and will be treated as a new resource of type 'anim'. The ID of this new resource will be held within a static variable defined within 'anim' static inner class within R.

08/01/10 © 2008 Haim Michael 16



Tween Animation