Image View

In Android, ImageView class is used to display an image file in application. Image file is easy to use but hard to master in Android, because of the various screen sizes in Android devices. An android is enriched with some of the best UI design widgets that allows us to build good looking and attractive UI based application.

ImageView class is used to display any kind of image resource in the android application either it can be **android.graphics.Bitmap** or **android.graphics.drawable.Drawable**

Attributes of ImageView

1. id: id is an attribute used to uniquely identify a image view in android. Below is the example code in which we set the id of a image view.

```
<ImageView
android:id="@+id/simpleImageView"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
/>
```

2. src: src is an attribute used to set a source file or you can say image in your imageview to make your layout attractive.

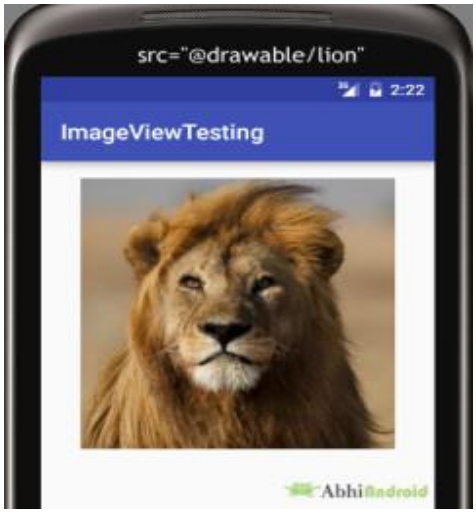
Below is the example code in which we set the source of a imageview lion which is saved in drawable folder.

```
<ImageView
android:id="@+id/simpleImageView"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:src="@drawable/lion" /><!--set the source of an image view-->
```

In Java:

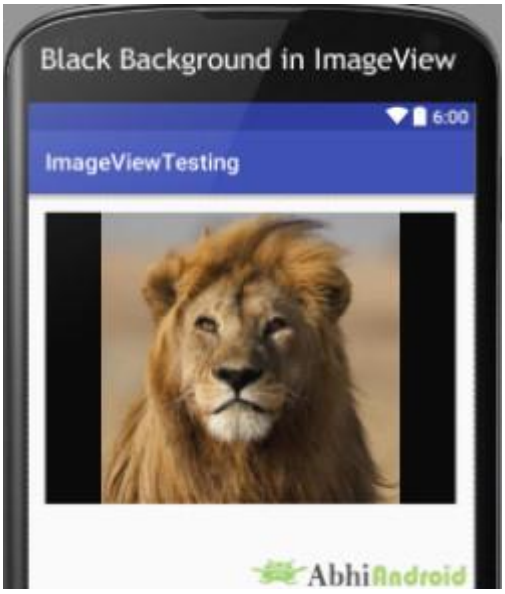
We can also set the source image at run time programmatically in java class. For that we use setImageResource() method as shown in below example code.

```
/*Add in Oncreate() funtion after setContentView()*/
ImageView simpleImageView=(ImageView) findViewById(R.id.simpleImageView);
simpleImageView.setImageResource(R.drawable.lion);//set the source in java class
```



3. background: background attribute is used to set the background of a ImageView. We can set a color or a drawable in the background of a ImageView.
Below is the example code in which we set the black color in the background and an image in the src attribute of image view.

```
<ImageView
    android:id="@+id/simpleImageView"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:src="@drawable/lion"
    android:background="#000"/><!--black color in background of a image view-->
```



In Java:
We can also set the background at run time programmatically in java class. In below example code we set the black color in the background of a image view.

```
/*Add in Oncreate() funtion after setContentView()*/
```

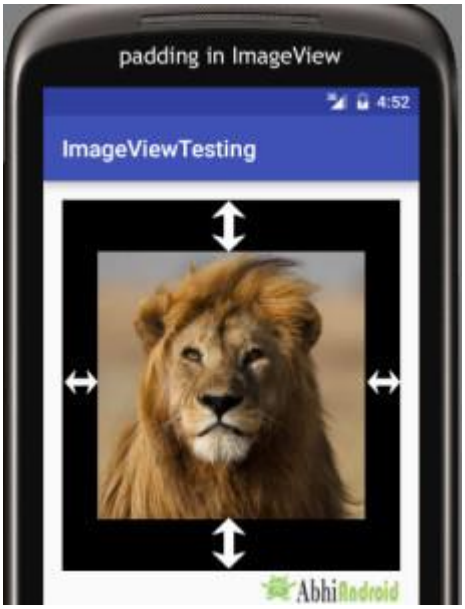
```
ImageView simpleImageView=(ImageView) findViewById(R.id.simpleImageView);

simpleImageView.setBackgroundColor(Color.BLACK);//set black color in background of a image view in java class
```

4. padding: padding attribute is used to set the padding from left, right, top or bottom of the Imageview.

- **paddingRight:** set the padding from the right side of the image view.
- **paddingLeft:** set the padding from the left side of the image view.
- **paddingTop:** set the padding from the top side of the image view.
- **paddingBottom:** set the padding from the bottom side of the image view.
- **padding:** set the padding from the all side's of the image view.

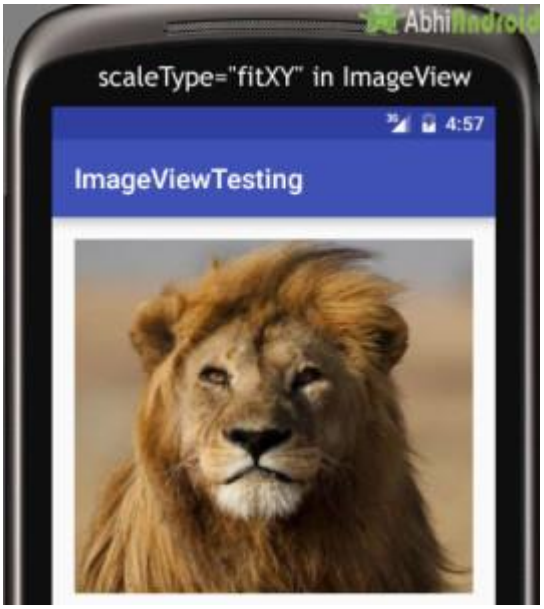
Below is the example code of padding attribute in which we set the 30dp padding from all the side's of a image view.

```
<ImageView
    android:id="@+id/simpleImageView"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:background="#000"
    android:src="@drawable/lion"
    android:layout_marginStart="14dp"
    android:layout_marginTop="153dp"
    android:layout_marginEnd="14dp"
    android:padding="30dp"/><!--set 30dp padding from all the sides-->
```



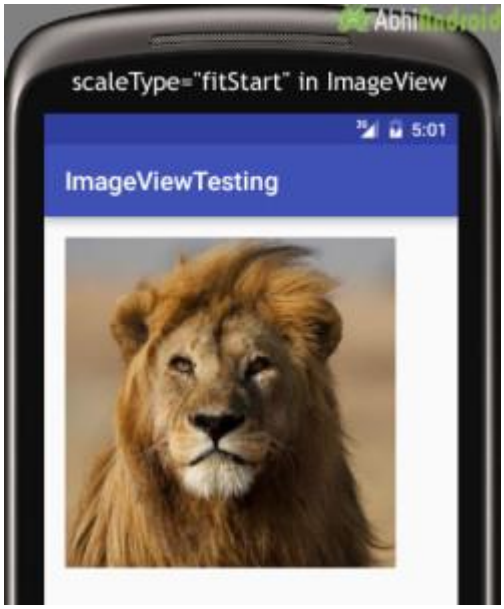
5. scaleType: scaleType is an attribute used to control how the image should be re-sized or moved to match the size of this image view. **The value for scale type attribute can be fit_xy, center_crop, fitStart etc.**
Below is the example code of scale type in which we set the scale type of image view to fit_xy.

```
<ImageView
android:id="@+id/simpleImageView"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:src="@drawable/lion"
android:scaleType="fitXY"/><!--set scale type fit xy-->
```



Let's we take an another example of scale type to understand the actual working of scale type in a image view.
In below example code we set the value for scale type "fitStart" which is used to fit the image in the start of the image view as shown below:

```
<ImageView
    android:id="@+id/simpleImageView"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:src="@drawable/lion"
    android:scaleType="fitStart"/><!--set scale type fit start of image view-->
```



All `scaleType` in ImageView

Here we have specified all the possible values for the attribute `scaleType` which can be used in your app for ImageView:

- **CENTER**: Places the image in center, but does not scale it.
- **CENTER_CROP**: Scales the image uniformly.
- **CENTER_INSIDE**: This will place the image inside the container and the edges of the image will not overlap with that of the container, the image will be inside it.
- **FIT_CENTER**: Scale the image from the center.
- **FIT_END**: Scale the image from the end of the container, i.e from the right hand side.
- **FIT_START**: Scale the image from the start of the container, i.e from the left hand side.

- **FIT_XY**: This will fill the complete container with the image. This generally distorts the image by stretching/sqeezing it in disproportionate ratios.
- **MATRIX**: Used to scale the image using the image matrix when drawing.

Example

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity" >

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_margin="20dp"
        android:layout_marginTop="12dp"
        android:layout_weight="0"
        app:srcCompat="@drawable/abc" />
</LinearLayout>
```

```
package com.example.demo;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ImageView iv=findViewById(R.id.imageView);
        iv.setOnClickListener(new View.OnClickListener() {
            int i=0;
            @Override
            public void onClick(View view) {
                i++;
                        if(i%2==0)
                iv.setImageResource(R.drawable.a1);
```

```
            else
                iv.setImageResource(R.drawable.abc);
            }
        });

    }
}
```