ImageView

Общая информация

Компонент **ImageView** предназначен для отображения изображений. Находится в разделе **Images**.

Для загрузки изображения в XML-файле используется атрибут **android:src**.

**ImageView** является базовым элементом-контейнером для использования графики. Можно загружать изображения из разных источников, например, из ресурсов программы, контент-провайдеров. В классе **ImageView** существует несколько методов для загрузки изображений:

* **setImageResource(int resId)** — загружает изображение по идентификатору ресурса
* **setImageBitmap(Bitmap bitmap)** — загружает растровое изображение
* **setImageDrawable(Drawable drawable)** - загружает готовое изображение
* **setImageURI(Uri uri)** — загружает изображение по его URI

Метод setImageResource()

Сначала нужно получить ссылку на **ImageView**, а затем используется идентификатор изображения из ресурсов:

ImageView imageView = (ImageView) findViewById(R.id.imageView);

imageView.setImageResource(R.drawable.cat);

Метод setImageBitmap()

Используется класс **BitmapFactory** для чтения ресурса изображения в объект **Bitmap**, а затем в **ImageView**указывается полученный **Bitmap**.

ImageView imageView = (ImageView) findViewById(R.id.imageView);

imageView.setImageBitmap(BitmapFactory.decodeResource(this.getResources(), R.drawable.cat));

Метод setImageDrawable()

Если у вас есть готовое изображение, например, на SD-карте, то его можно использовать в качестве объекта **Drawable**.

ImageView imageView = (ImageView) findViewById(R.id.imageView);

// плохой код. только для демонстрации

imageView.setImageDrawable(Drawable.createFromPath("/mnt/sdcard/cat.jpg"));

Метод setImageURI()

Берётся **URI** файла изображения и используется в качестве источника изображения. Этот способ годится для работы с локальными изображениями.

ImageView imageView = (ImageView) findViewById(R.id.imageView);

imageView.setImageURI(URI.parse("file://mnt/sdcard/cat.jpg"));

Другие методы

Также вам часто придется использовать методы, связанные с размерами и масштабированием: **setMaxHeight()**, **setMaxWidth()**, **getMinimunHeight()**, **getMinimunWidth()**, **getScaleType()**, **setScaleType()**.

Масштабирование через свойство Scale Type

Для масштабирования картинки в **ImageView** есть свойство **Scale Type** и соответствующий ему атрибут **android:scaleType** и перечисление **ImageView.ScaleType**.

Допустим, у нас есть простенькая разметка:

<?xml version="1.0" encoding="UTF-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:id="@+id/LinearLayout1"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:padding="5dp" >

<ImageView

android:id="@+id/icon"

android:layout\_width="match\_parent"

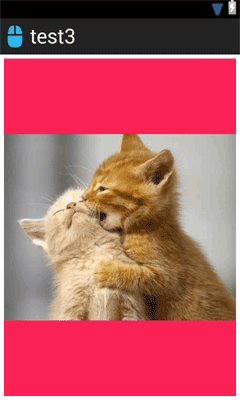
android:layout\_height="match\_parent"

android:background="#fa2255"

android:src="@drawable/catlove" />

</LinearLayout>

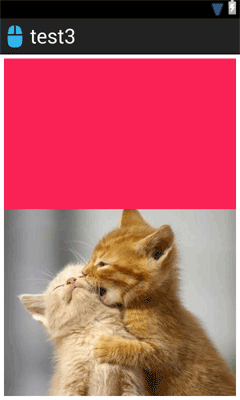
Для наглядности я задал красный цвет для фона **ImageView**. По умолчанию, у меня картинка установилась в режиме **android:scaleType="fitCenter"**.



Если выбрать режим **android:scaleType="fitStart"**, то картинка прижимается к левому верхнему углу и таким образом заполняет верхнюю половину ImageView.



Значение **android:scaleType="fitEnd"** сместит картинку в нижнюю часть контейнера.



Режим **android:scaleType="center"** выводит картинку в центре без растягивания. Аналогично в моём случае повела себя картинка и в режиме **android:scaleType="centerInside"**. Если у вас будет картинка большего размера, то она смаштабируется, чтобы поместиться в центре.



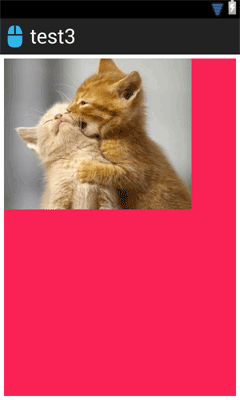
Режим **android:scaleType="centerCrop"** равномерно растягивает картинку, чтобы заполнить весь контейнер и обрезает лишнее.



**android:scaleType="fitXY"** растягивает/сжимает картинку, чтобы подогнать её к контейнеру.



Последний атрибут **android:scaleType="matrix"** вывел картинку без изменений в левом верхнем углу.



Атрибут android:adjustViewBounds="true"

При использовании атрибута **scaleType="fitCenter"** из предыдущего примера Android вычисляет размеры самой картинки, игнорируя размеры **ImageView**. В этом случае ваша разметка может "поехать". Атрибут **adjustViewBounds** заставляет картинку подчиниться размеру компонента-контейнера. В некоторых случаях это может не сработать, например, если у **ImageView** установлен атрибут **layout\_width="0dip"**. В таком случае поместите **ImageView** в **RelativeLayout** или **FrameLayout** и используйте значение **0dip** для этих контейнеров.

Загрузка изображения из галереи

Предположим, у вас есть на экране компонент **ImageView**, и вы хотите загрузить в него какое-нибудь изображение из галереи по нажатию кнопки:

static final int GALLERY\_REQUEST = 1;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

Button button = (Button)findViewById(R.id.button);

button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent photoPickerIntent = new Intent(Intent.ACTION\_PICK);

photoPickerIntent.setType("image/\*");

startActivityForResult(photoPickerIntent, GALLERY\_REQUEST);

}

});

}

@Override

protected void onActivityResult(int requestCode, int resultCode, Intent imageReturnedIntent) {

super.onActivityResult(requestCode, resultCode, imageReturnedIntent);

Bitmap bitmap = null;

ImageView imageView = (ImageView) findViewById(R.id.imageView);

switch(requestCode) {

case GALLERY\_REQUEST:

if(resultCode == RESULT\_OK){

Uri selectedImage = imageReturnedIntent.getData();

try {

bitmap = MediaStore.Images.Media.getBitmap(getContentResolver(), selectedImage);

} catch (IOException e) {

e.printStackTrace();

}

imageView.setImageBitmap(bitmap);

}

}

}

Намерение **ACTION\_PICK** вызывает отображение галереи всех изображений, хранящихся на телефоне, позволяя выбрать одно изображение. При этом возвращается адрес **URI**, определяющий местоположение выбранного изображения. Для его получения используется метод **getData()**. Далее для преобразования URI-адреса в соответствующий экземпляр класса **Bitmap** используется специальный метод **Media.getBitmap()**. И у нас появляется возможность установить изображение в **ImageView** при помощи **setImageBitmap()**.

На самом деле можно поступить ещё проще и использовать метод **setImageURI**.

Uri selectedImage = imageReturnedIntent.getData();

imageView.setImageURI(selectedImage);

Сравните с предыдущим примером - чувствуете разницу? Тем не менее, приходится часто наблюдать подобный избыточный код во многих проектах. Это связано с тем, что метод порой кэширует адрес и не происходит изменений. Рекомендуется использовать инструкцию **setImageURI(null)** для сброса кэша и повторный вызов метода с нужным **Uri**.

В последних версиях системных эмуляторов два примера не работают. Проверяйте на реальных устройствах.

Получить размеры ImageView - будьте осторожны

У элемента **ImageView** есть два метода **getWidth()** и **getHeight()**, позволяющие получить его ширину и высоту. Но если вы попробуете вызвать указанные методы сразу в методе **onCreate()**, то они возвратят нулевые значения. Можно добавить кнопку и вызвать данные методы через нажатие, тогда будут получены правильные результаты. Либо использовать другой метод активности, который наступает позже.

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

final ImageView imageView = (ImageView) findViewById(R.id.imageView);

final TextView infoTextView = (TextView) findViewById(R.id.textView);

// пробуем получить размеры ImageView сразу при загрузке. Вернёт нулевые значения

infoTextView.setText("Размеры ImageView: " + String.valueOf(imageView.getWidth())

+ " : " + String.valueOf(imageView.getHeight()));

Button button = (Button)findViewById(R.id.button);

button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

// этот же код вызывается при нажатии кнопки. Теперь всё правильно

infoTextView.setText("Размеры ImageView: " + String.valueOf(imageView.getWidth())

+ " : " + String.valueOf(imageView.getHeight()));

}

});

}

Копирование изображений между ImageView

Если вам надо скопировать изображение из одного **ImageView** в другой, то можно получить объект **Drawable**через метод **getDrawable()** и присвоить ему второму компоненту.

ImageView ivSource = (ImageView) findViewById(R.id.sourceImageView); // 1-й компонент с какой-то картинкой

ImageView ivTarget = (ImageView) findViewById(R.id.targetImageView); // 2-й компонент без картинки

Drawable drawable = ivSource.getDrawable(); // получим картинку у первого компонента

ivTarget.setImageDrawable(drawable); // присвоим второму