



2.9. Графические ресурсы. Стили. Адаптивные макеты приложения

Курс по программированию от
IT Академии Samsung

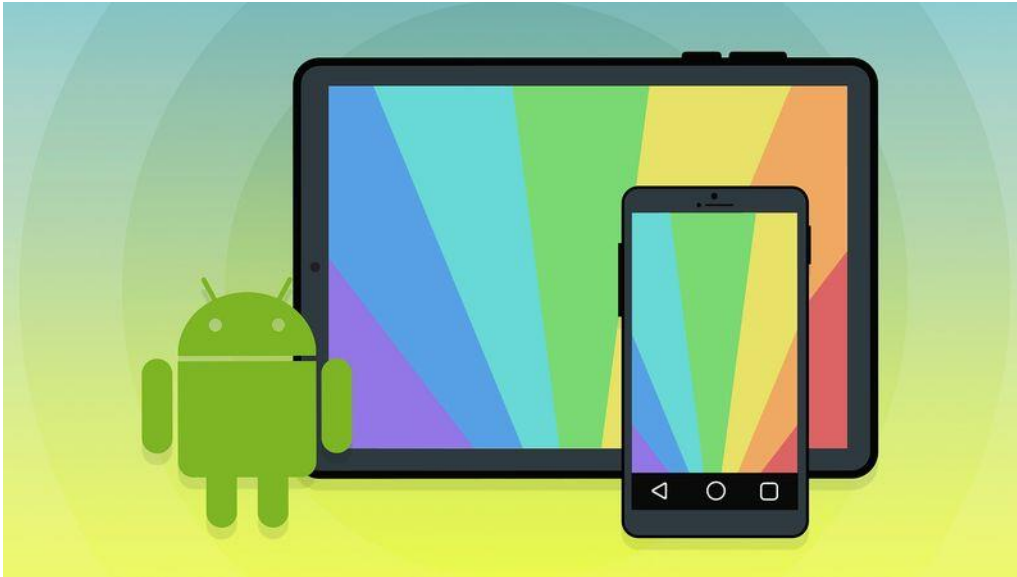


2.9.1. Графические ресурсы Android-приложений

SAMSUNG



Для статических изображений в приложениях можно задействовать класс **Drawable** и его подклассы для. **Drawable** - это общая абстракция для рисования. Различные его подклассы помогают в определенных ситуациях. Их можно расширить, чтобы задать собственные объекты рисования, которые ведут себя специальным образом.



Существует два способа создания экземпляра класса **Drawable** (кроме использования конструкторов):

- Обратиться к ресурсу изображения в проекте.
- Использовать ресурс XML, который определяет свойства для рисования.





2.9.1. Графические ресурсы Android-приложений

SAMSUNG



В Android-приложения возможно добавить графику в ваше приложение, ссылаясь на файл изображения из ресурсов проекта. При этом поддерживаются следующие типы:

- PNG (предпочтительно);
- JPG (приемлемо);
- GIF (не рекомендуется).

```
val myImage: Drawable =  
    ResourcesCompat.getDrawable(context.resources,  
        R.drawable.test, null)
```





2.9.1. Графические ресурсы Android-приложений

SAMSUNG



```
private lateinit var constraintLayout: ConstraintLayout
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    // Создание экземпляра ImageView и определение его свойств
    val i = ImageView(this).apply {
        setImageResource(R.drawable.test)
        contentDescription = resources.getString(R.string.test_desc)
        //устанавливаем границы ImageView
        adjustViewBounds = true
        layoutParams = ViewGroup.LayoutParams( ViewGroup.LayoutParams.WRAP_CONTENT,
                                                ViewGroup.LayoutParams.WRAP_CONTENT)
    }

    // Создаем ConstraintLayout, в который нужно добавить
    // ImageView constraintLayout = ConstraintLayout(this).apply {
    //     // Добавляем ImageView в разметку.
    //     addView(i)
    // } // Устанавливаем разметку
    setContentView(constraintLayout)
}
```





2.9.1. Графические ресурсы Android-приложений

SAMSUNG



<ImageView

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:src="@drawable/test"

android:contentDescription="@string/test_desc" />



- При использовании графических ресурсов следует убедиться, что изображения имеют подходящий размер в пикселях. Для избежания подобного рода проблем следует ознакомиться со [следующими материалами](#).
- Более подробную информацию об использовании графических ресурсов можно найти в [официальной документации](#). Кроме указанного выше примера в приложениях возможно рисунков из [XML-ресурсов](#), использование форм для рисунков ([Shape Drawables](#)), использование растровых изображений, которое вы можете использовать в качестве фона представления ([NinePatch drawables](#)) и др.



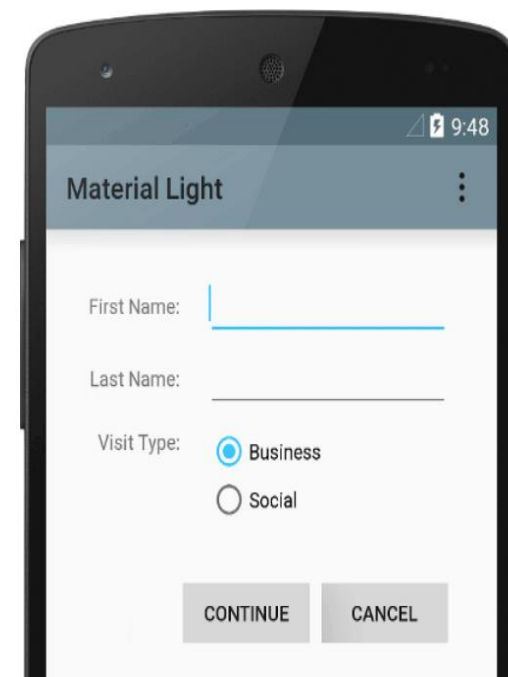
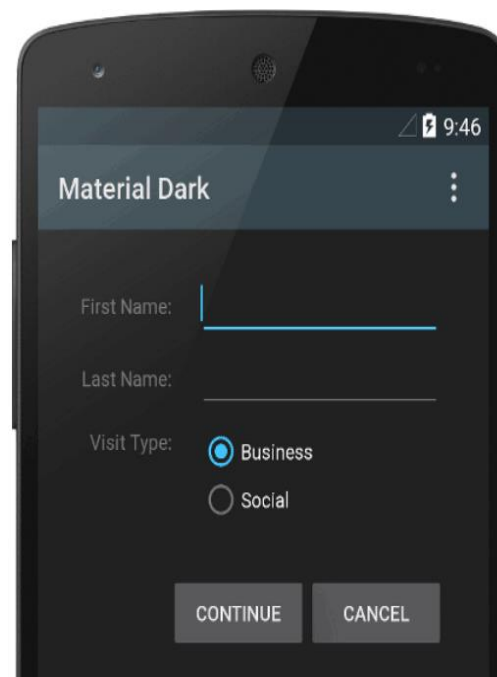


2.9.2. Стили и темы Android приложения



- **Стиль** - это набор атрибутов, которые определяют внешний вид представлений. Можно задавать такие атрибуты, как цвет шрифта, размер шрифта, цвет фона и многое другое.
- **Тема** приложения - это стиль, который применяется ко всему приложению, а не только к отдельному элементу пользовательского интерфейса.

Стили и темы объявляются в файле ресурсов стилей в *res/values /*, обычно называемом *styles.xml*.





2.9.2. Стили и темы Android приложения

SAMSUNG




Чтобы создать собственный стиль или тему, необходимо открыть файл *res/values/styles.xml*.

Далее следует выполнить следующие действия:

- добавьте элемент `<style>` с именем, однозначно определяющим стиль;
- добавьте элемент `<item>` для каждого атрибута стиля, который вы хотите определить.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <style name="RedText" parent="TextAppearance.AppCompat">
    <item name="android:textColor">#FF0000</item>
  </style>
</resources>
```



```
<Button
  style="@style/RedText"
  ...
/>
```





2.9.2. Стили и темы Android приложения

SAMSUNG



- При создании собственных стилей необходимо расширять существующие чтобы поддерживать совместимость со стилями пользовательского интерфейса платформы. Например, можно унаследовать внешний вид текста платформы Android по умолчанию и изменить его следующим образом:

```
<style name="RedText" parent="@android:style/TextAppearance">  
  <item name="android:textColor">#FF0000</item>  
</style>
```

- Ваши приложения могут наследовать стили из стандартной библиотеки Android. Это стало возможно начиная с версии Android 4.0 и выше. Например следующим образом.

```
<style name="RedText" parent="TextAppearance.AppCompat">  
  <item name="android:textColor">#FF0000</item>  
</style>
```

- Можно строить менять свои стили, строя из них своеобразную "цепочку".





2.9.2. Стили и темы Android приложения

SAMSUNG



В Android так же как и стили, можно создавать и темы. Отличие заключается в том, что тема будет применена не к отдельному представлению а к целому приложению или отдельно взятой Активности. Тогда, в случае если необходимо применить темную тему для всего приложения в файле AndroidManifest.xml. следует прописать следующее:

```
<manifest ... >  
  <application android:theme="@style/Theme.AppCompat" ... >  
  </application>  
</manifest>
```

Теперь каждое представление в приложении или Активности будет в едином стиле. Полную информацию по выбору стилей можно посмотреть на сайте developer.android.com/



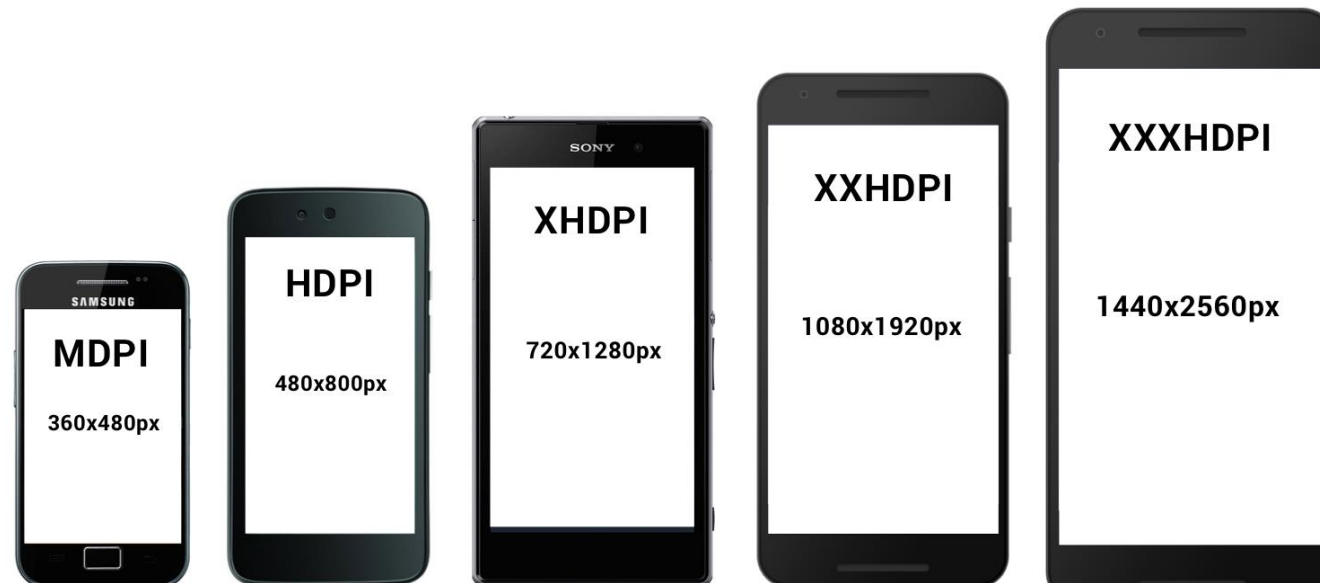


2.9.3. Ресурсы для адаптивных макетов

SAMSUNG



- Существует четыре обобщенных размера: маленький (small), нормальный (normal), большой (large), очень большой (xlarge) и четыре обобщенные плотности: низкая (ldpi), средняя (mdpi), высокая (hdpi), сверхвысокая (xhdpi).
- Для использования различных изображений для разных экранов необходимо поместить эти изображения в альтернативные ресурсы, которые располагаются в отдельных каталогах (подобно тому, как вы мы использовали различные строки для разной локализации приложения).



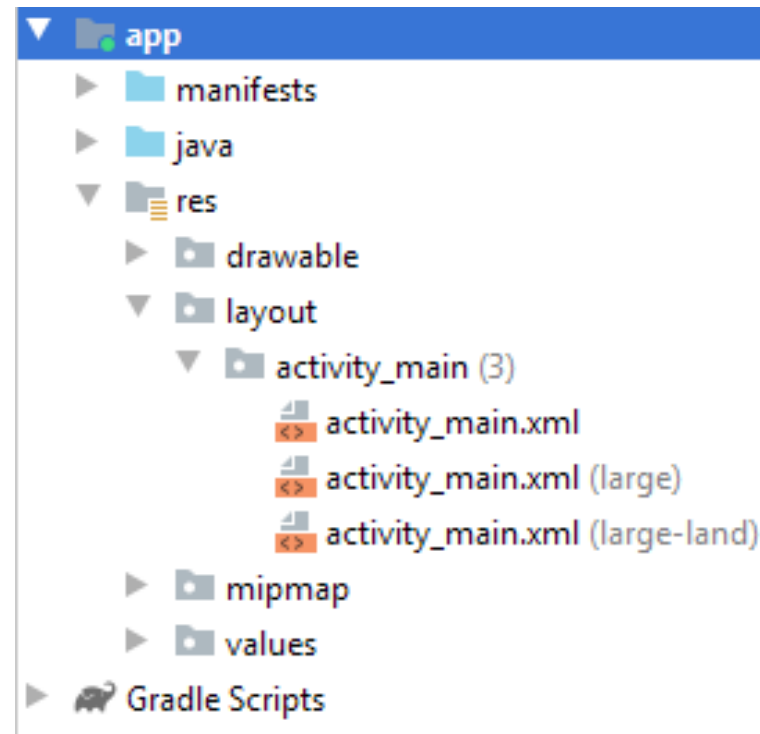
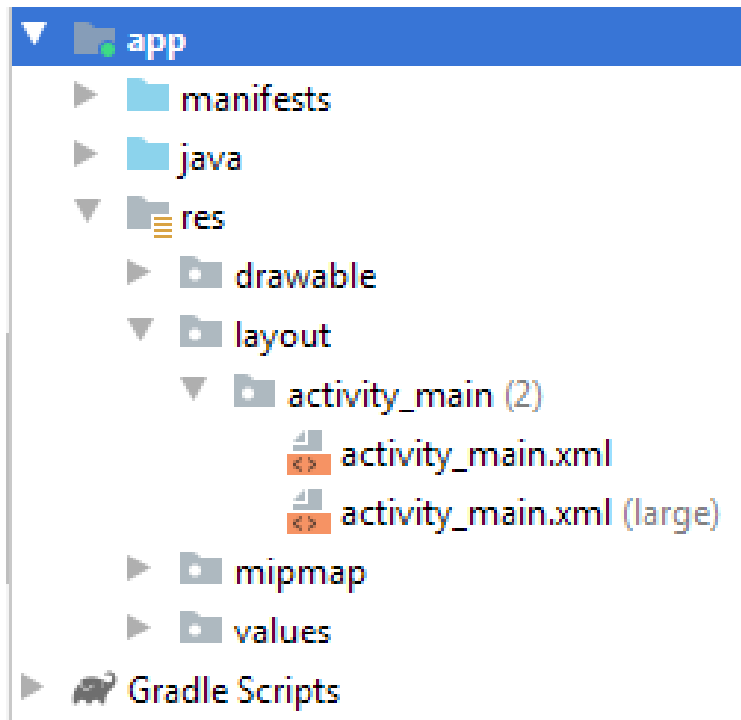


2.9.3. Ресурсы для адаптивных макетов

SAMSUNG



- Каждый макет должен быть сохранен в папке соответствующих ресурсов, названный с суффиксом размера экрана. Например, макет для больших экранов должен быть сохранен в папке **res/layout-large/**.

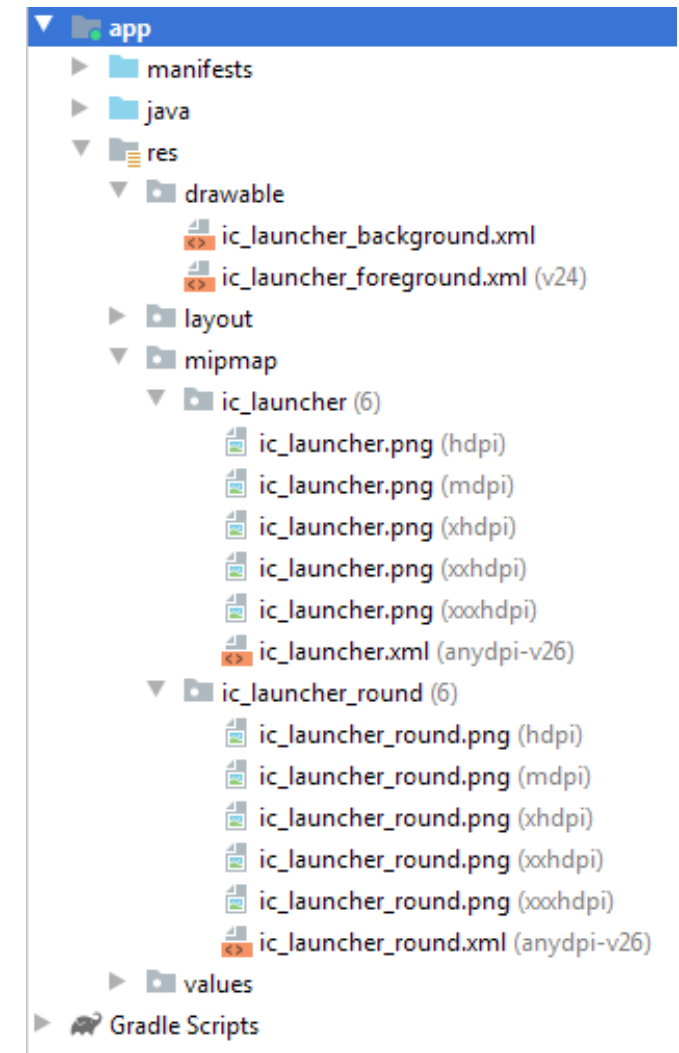




2.9.3. Ресурсы для адаптивных макетов

Вы всегда должны предоставлять растровые ресурсы, которые корректно масштабируются для каждой из обобщенных плотностей: низкой, средней, высокой и сверхвысокой плотности. Это поможет вам достичь хорошего графического качества и производительности на всех плотностях экрана. Для создания этих изображений, вы должны начать с ресурса в векторном формате и генерировать изображения для каждой плотности, используя следующую шкалу размеров:

- xhdpi: 2.0
- hdpi: 1.5
- mdpi: 1.0 (базовый)
- ldpi: 0.75





Упражнение 2.9

SAMSUNG



Разработаем приложение с адаптивной разметкой. В приложении будет присутствовать два файла `activity_main.xml` для портретной и ландшафтной ориентации. В каждом файле разметки есть `ImageView` с `ShapeDrawable`, заданной через xml (`ic_rounded_rectangle.xml`). Положение элементов на экране будет разным в зависимости от ориентации.

Сначала в папке `drawable` создадим файл `ic_rounded_rectangle.xml`. Его код будет выглядеть следующим образом.

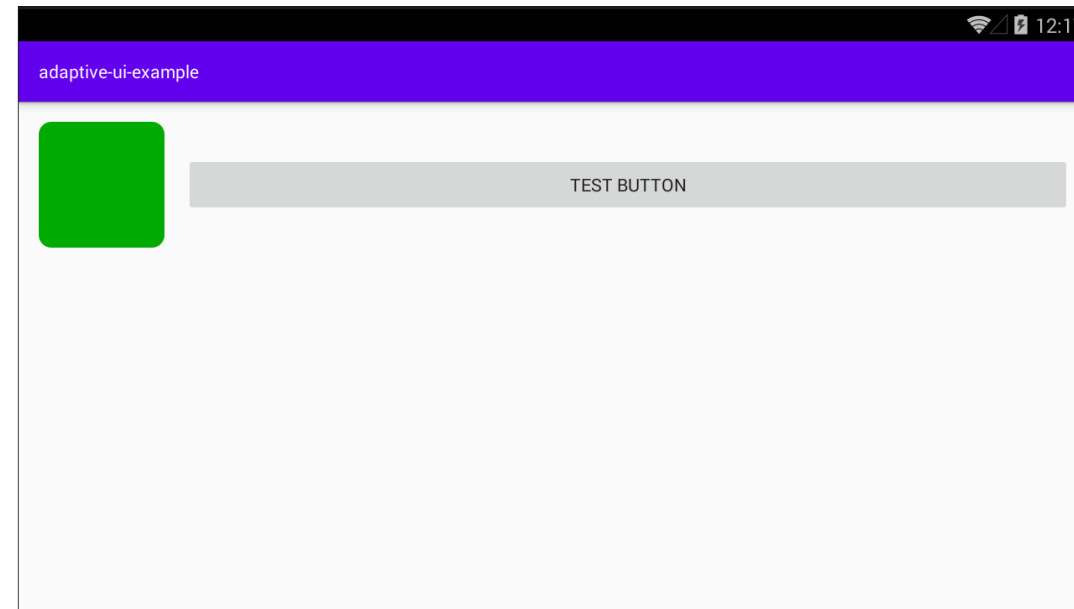
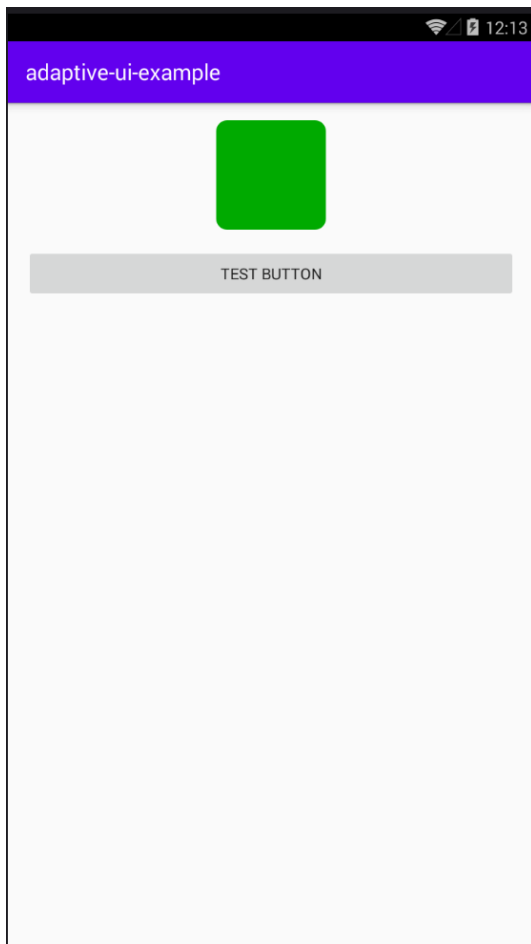
```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <size
        android:width="100dp"
        android:height="100dp" />
    <solid android:color="#00AA00"/>
    <corners android:radius="10dp"/>
</shape>
```





Упражнение 2.9

SAMSUNG



SAMSUNG



Спасибо за внимание!

Курс по программированию от
IT Академии Samsung