УНИВЕРСИТЕТ ИТМО

Ресурсы приложения и макеты экрана

Санкт-Петербург, 2016

Ресурс - это

- значение (например, заголовок экрана или телефон справочной вашей тех. поддержки)
- или файл (например, музыкальный файл, файл, описывающий компоновку экрана, или даже картинка)

Но, самое главное:
их можно модифицировать или предоставлять альтернативы без перекомпиляции исходного кода

Строковые ресурсы

app > src > main > res > values > strings.xml

strings.xml

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

    <string name="app_name">Resources Demo</string>
    <string name="hello_world">Hello world!</string>
    <string name="action_settings">Settings</string>

</resources>
```

app > src > main > res > values-ru > strings.xml

ru/strings.xml

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

    <string name="app_name">Пример использования ресурсов</string>
    <string name="hello_world">Здравствуй Мир!</string>
    <string name="action_settings">Настройки</string>

</resources>
```

Ресурсы компоновки

app > src > main > res > layout > activity_main.xml

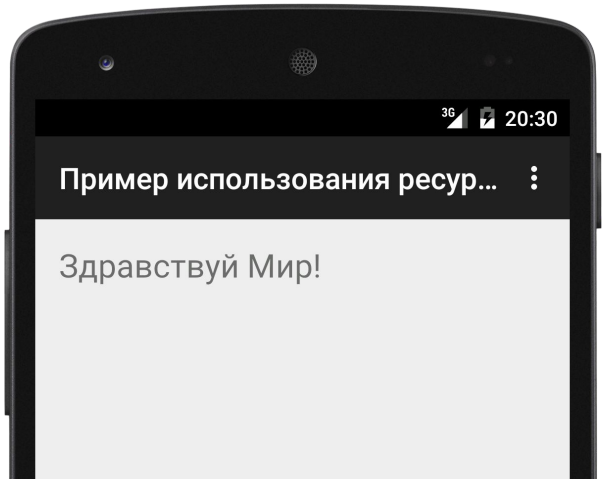
activity_main.xml

```
1  <RelativeLayout
2      xmlns:android="http://schemas.android.com/apk/res/android"
3      android:layout_width="match_parent"
4      android:layout_height="match_parent">
5
6      <TextView
7          android:text="@string/hello_world"
8          android:layout_width="wrap_content"
9          android:layout_height="wrap_content" />
10
11 </RelativeLayout>
```

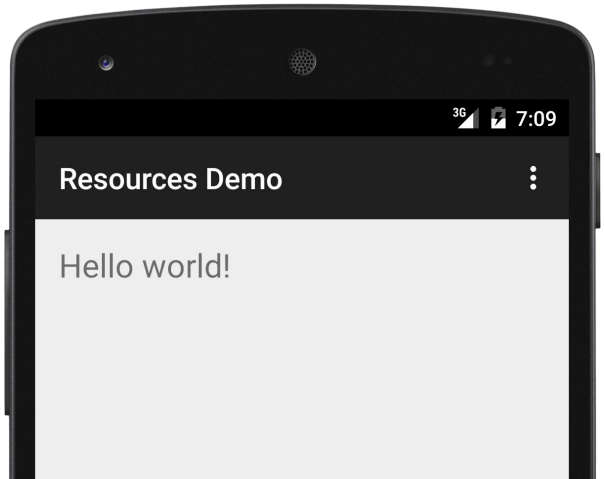
Активность .java

```java
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}
```

RU



Default



R.java

app > build > generated > source > r > debug > com > example > resapp > R.java

R.java

Generated source files should not be edited. The changes will be lost when sources are regenerated.

```
1  package com.example.resapp;
2
3  public final class R {
4
5      public static final class layout {
6          public static final int activity_main=0x7f030017;
7      }
8
9      public static final class string {
10         public static final int action_settings=0x7f0a0010;
11         public static final int app_name=0x7f0a0011;
12         public static final int hello_world=0x7f0a0012;
13     }
14
15     /* Тут были и другие элементы,
16        но мы их пока спрячем      */
17 }
```

Обращение к ресурсам приложения

Существует только два способа:

- Через XML

```
@string/hello_world
```

- Через код Java

```
R.string.hello_world
```

Синтаксис ссылок

Java

[package.]R.{type}.{name}

- R.drawable
- R.id
- R.layout
- R.string
- R.array

…

XML

@[package:]{type}\{name}

- drawable
- id
- layout
- string
- string-array

…

Обращение к ресурсам через XML

app > src > main > res > layout > activity_main.xml

activity_main.xml

```
1  <RelativeLayout
2      xmlns:android="http://schemas.android.com/apk/res/android"
3      android:layout_width="match_parent"
4      android:layout_height="match_parent">
5
6      <ImageView
7          android:src="@drawable/cat"
8          android:layout_width="wrap_content"
9          android:layout_height="wrap_content" />
10
11 </RelativeLayout>
```

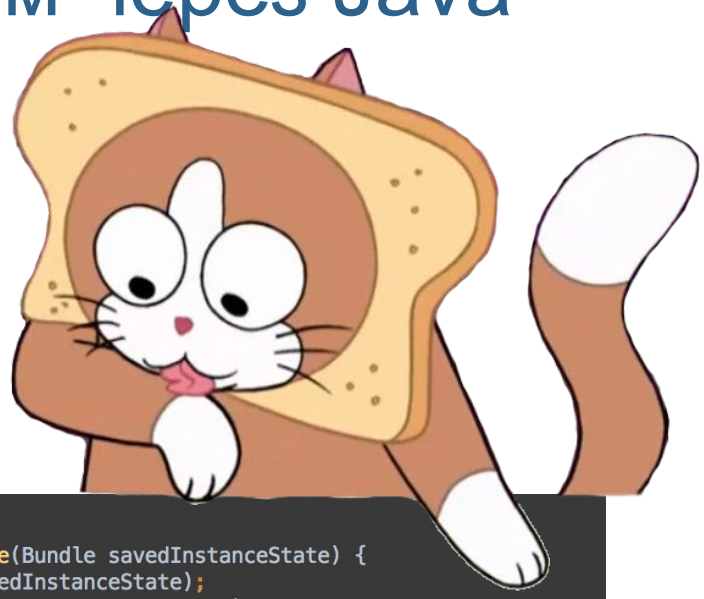
res/drawable/cat.png

Обращение к ресурсам через Java

app > src > main > res > layout > activity_main.xml

activity_main.xml

```
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:id="@+id/imageWithCat"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```



res/drawable/cat.png

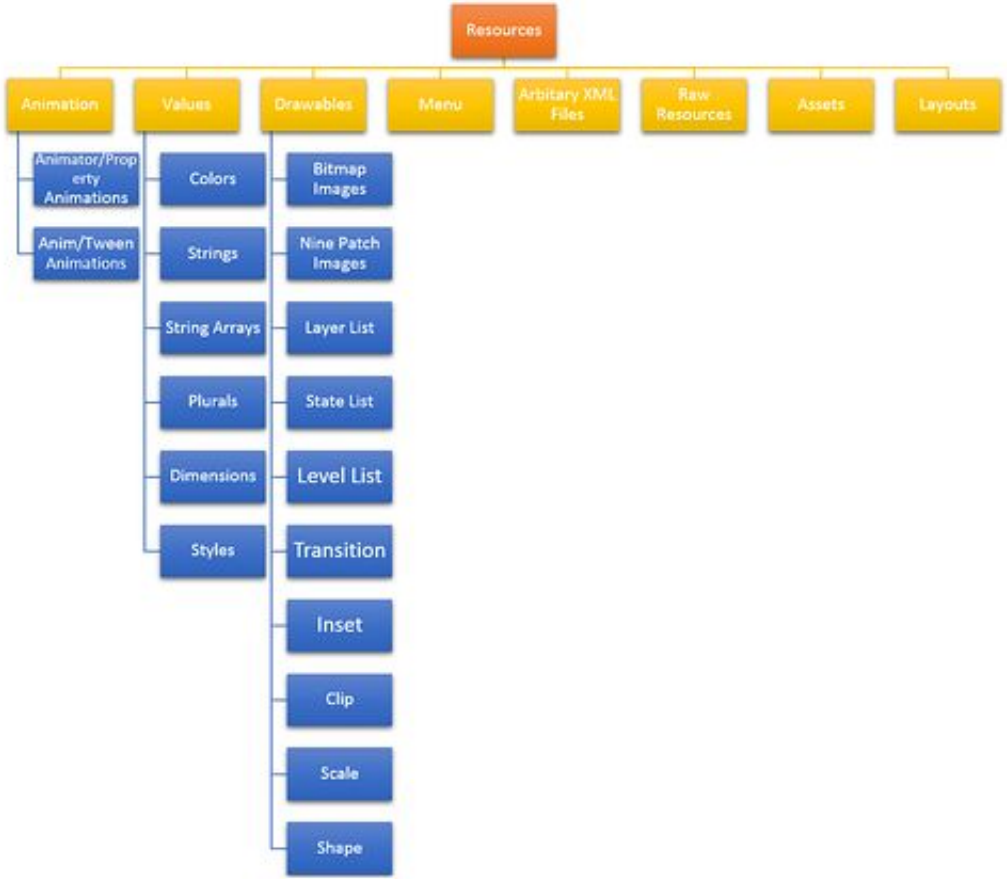
```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    ImageView cat = (ImageView) findViewById(R.id.imageWithCat);
    cat.setImageDrawable(getResources().getDrawable(R.drawable.cat));
}
```

Идентификаторы ресурсов

`@+id/your_best_id` – создает новый id

`@id/another_best_id` – использует уже созданный

Обзор структуры каталогов ресурсов




```
MyProject/
    java/com/example/MainActivity.java
    res/
        animator/
            *.xml
        anim/
            *.xml
        color/
            *.xml
        drawable/
            *.{png|9.png|jpg|gif}
            *.xml
        layout/
            *.xml
        menu/
            *.xml
        raw/
            *.*
        values/
            *.xml
        xml/
            *.xml
    assets/
        *.*/*.*
```

Основные типы ресурсов

```
MyProject/
    java/com/example/MainActivity.java
    res/
        drawable/
            myimage.png  - изображение
        layout/
            activity_main.xml  -  компоновка пользовательского интерфейса
        values/
            colors.xml   - цвета
            dimens.xml   - размерности
            strings.xml  - строки
            styles.xml   - стили
    assets/
        *.*/*.*  -  низкоуровневые ресурсы
```

Изображения

Расположение файла:
`res/drawable/{drawable_name}.{png|jpg|gif|9.png|xml}`
Ссылка в Java: `[package.]R.drawable.{drawable_name}`
Ссылка в XML: `@[package:]drawable/{drawable_name}`
XML:

```
<Button
    android:id="@+id/helloButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/hello_button"
    android:background="@drawable/btn_background" />
```

```
<ImageView
    android:src="@drawable/cat"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

Java
:

```
Drawable background = getResources().getDrawable(R.drawable.btn_background);

button.setBackground(background);
/* или проще */
button.setBackgroundResource(R.drawable.btn_background);
```

Компоновки

Расположение файла: `res/layout/{layout_name}.xml`

Ссылка в Java: `[package.]R.layout.{layout_name}`

Ссылка в XML: `@[package:]layout/{layout_name}`

Структура в XML:

```
<!--
    <ViewGroup
        [ViewGroup-specific attributes] >
        <View
            [View-specific attributes] />
    </ViewGroup >
-->
```

app > src > main > res > layout > activity_main.xml

activity_main.xml

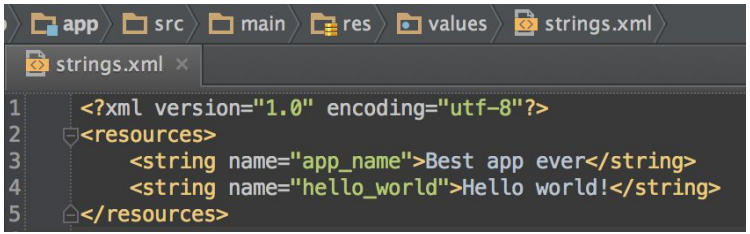
```
1  <RelativeLayout
2      xmlns:android="http://schemas.android.com/apk/res/android"
3      android:layout_width="match_parent"
4      android:layout_height="match_parent">
5
6      <ImageView
7          android:id="@+id/imageWithCat"
8          android:layout_width="wrap_content"
9          android:layout_height="wrap_content" />
10
11 </RelativeLayout>
```

Строки

Расположение файла:
`res/values/{strings}.xml`

Ссылка в Java:
`[package.]R.string.{string_name}`

Ссылка в XML:
`@[package:]string/{string_name}`



XML:

```xml
<TextView
    android:text="@string/hello_world"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

Java:

```java
String helloWorld = getString(R.string.hello_world);
```

Цвета

Расположение файла:
`res/values/{colors}.xml`

Ссылка в Java:
`[package.]R.color.{color_name}`

Ссылка в XML:
`@[package:]color/{color_name}`

```
<TextView
    android:text="@string/hello_world"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textColor="@color/opaque_red"/>
```

```
int red = getResources().getColor(R.color.opaque_red);
```

- #RGB
- #ARGB
- #RRGGBB
- #ARRGGBB

app > src > main > res > values > colors.xml

colors.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <resources>
3      <color name="opaque_red">#f00</color>
4      <color name="translucent_red">#80ff0000</color>
5  </resources>
```

Размерности

Расположение файла:
`res/values/{dimens}.xml`

Ссылка в Java:
`[package.]R.dimen.{dimension_name}`

Ссылка в XML:
`@[package:]dimen/{dimension_name}`

```
<TextView
    android:text="@string/hello_world"
    android:padding="@dimen/hello_padding"
    android:textSize="@dimen/hello_text_size"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

```
float textSize = getResources().getDimension(R.dimen.hello_text_size);
```

app > src > main > res > values > dimens.xml

dimens.xml

```
1  <resources>
2      <!-- Default screen margins, per the Android Design guidelines. -->
3      <dimen name="activity_horizontal_margin">16dp</dimen>
4      <dimen name="activity_vertical_margin">16dp</dimen>
5  </resources>
```

- **dp** – пиксели, не зависящие от разрешения;
- **sp** – пиксели, не зависящие от масштаба.

Стили

Расположение файла:
`/res/values/{styles}.xml`

Ссылка в XML:
`@[package:]style/{style_name}`

```
<TextView
    android:text="@string/hello_world"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    style="@style/BigRedText" />
```

app > src > main > res > values > styles.xml

styles.xml

```
1  <resources>
2
3      <!-- Base application theme. -->
4      <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
5          <!-- Customize your theme here. -->
6          <item name="android:windowBackground">@android:color/black</item>
7      </style>
8
9  </resources>
```

Контейнер

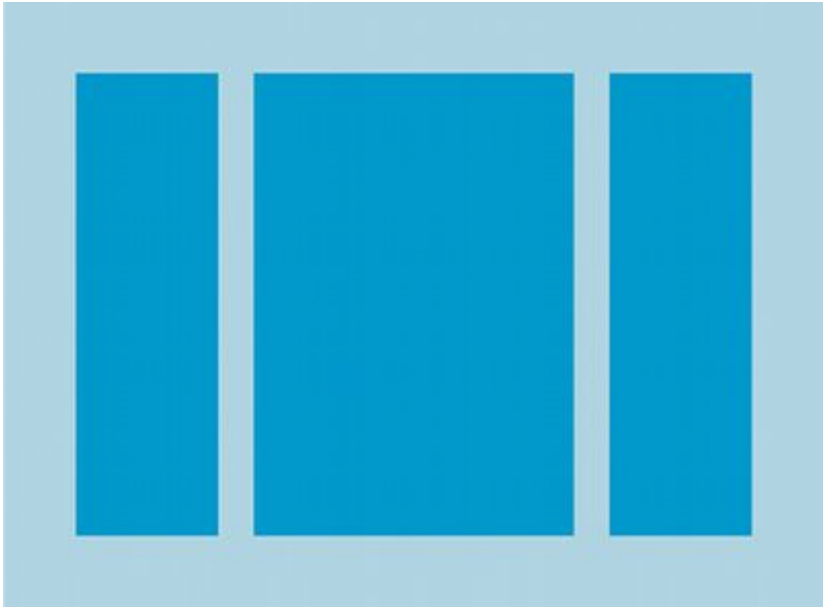
Контейнер определяет визуальную структуру пользовательского интерфейса, например, пользовательского интерфейса операцииКонтейнер определяет визуальную структуру пользовательского интерфейса, например, пользовательского интерфейса операции или виджета приложения.

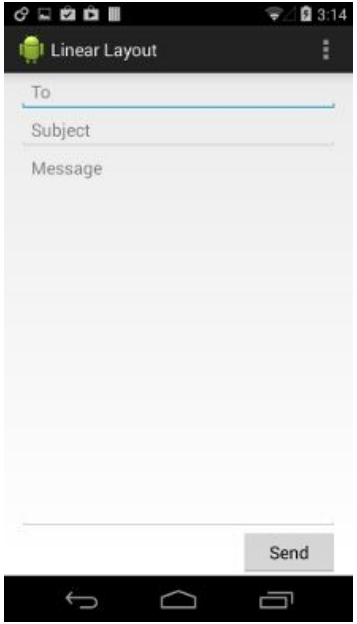
Существует два способа объявления:

- **Объявление элементов пользовательского интерфейса в XML**. В Android имеется удобный справочник XML-элементов для классов View и их подклассов, например таких, которые используются для виджетов и макетов.
- **Создание экземпляров элементов во время выполнения**. Ваше приложение может программным образом создавать объекты View и ViewGroup (а также

LinearLayout

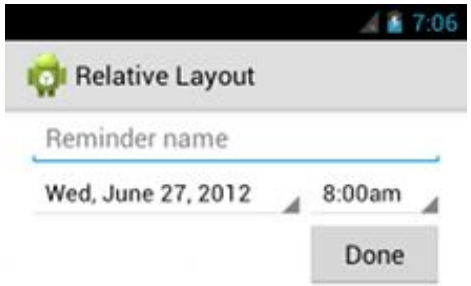
В линейной компоновке, как следует из названия, все элементы отображаются в одном направлении по горизонтали или по вертикали.





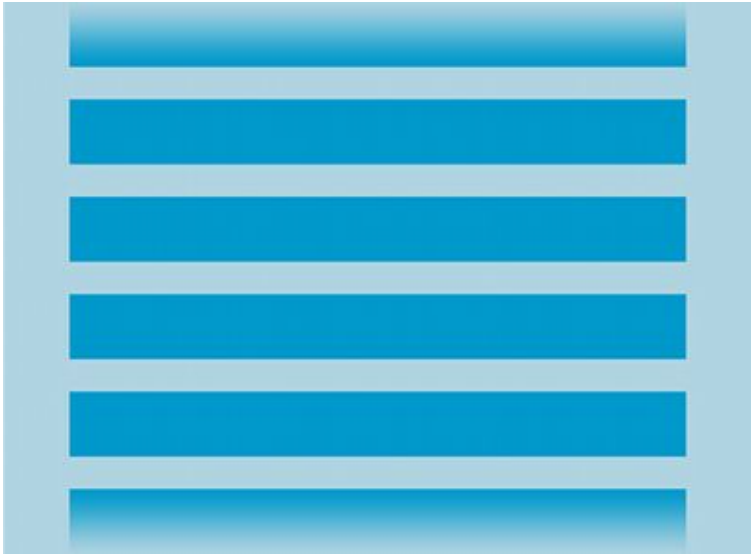
RelativeLayout

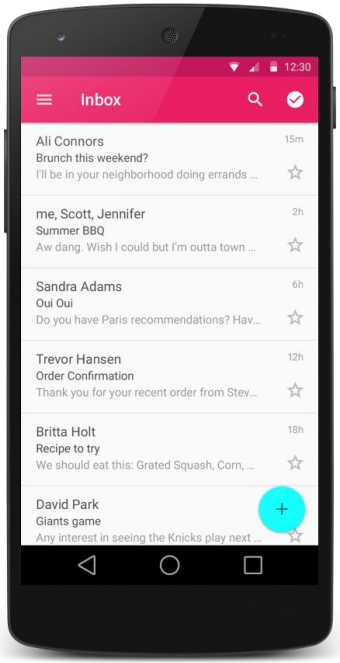
В относительной компоновке каждый элемент организует себя по отношению к другим элементам или родительского элемента.



List View

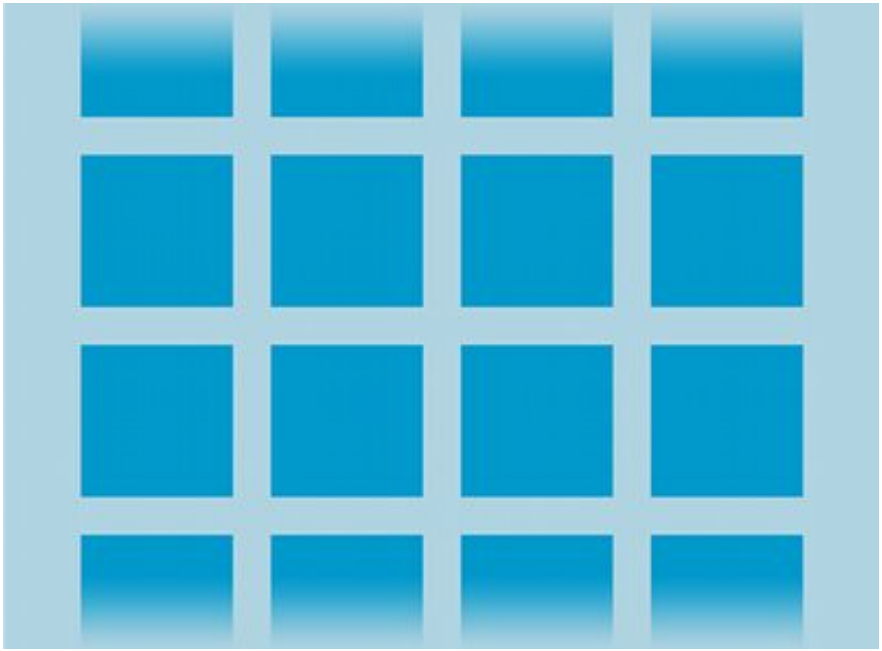
ListView представляет собой вид группы, которая отображает список пунктов. Элементы списка автоматически добавляются в список с помощью адаптера, который вытягивает содержимое из источника, такого как массив или база данных.





Grid View

Grid View является потомком ViewGroup, который отображает элементы в двумерной сетке, прокруткой. Элементы сетки автоматически вставляются в макет с помощью ListAdapter.



Материалы и ссылки

https://developer.android.com/guide/topics/resources/available-resources.html

https://developer.android.com/guide/topics/ui/declaring-layout.html#load

http://www.android-app-patterns.com