# Второй курс, осенний семестр 2016/17
# Конспект семинаров по программированию под Android

Собрано 17 октября 2016 г. в 11:19

---

## Содержание

# Глава #1
# Android Studio

3 октября

## 1.1. Установка

Всё просто: заходим на сайт, скачиваем, устанавливаем.

## 1.2. Создание проекта

File → New → New Project

1) Название

   Application name — название приложения, например, **My Application**.

   Company Domain — например, **me.spbau.ru**

   Из Application name и Company Domain складывается название пакета: **ru.spbau.me.myapplication**.

2) Target Android Devices

   Выбираем **Phone and Tablet** и минимальную версию Андроида, которая будет поддерживаться приложением. Кнопка **Help me choose** покажет, сколько процентов пользователей смогут пользоваться приложением, если оно будет поддерживать эту версию, и какие возможности есть у разных версий.

3) Add an Activity to Mobile

   Выбираем **Empty**.

   Активити — это, грубо говоря, окно приложения (подробнее будет дальше).

4) Customize the Activity

   Activity Name — имя Джава-класса, соответствующего этому активити.

   Layout Name — имя XML-файла, соответствующего этому активити.

## 1.3. SDK

SDK — software development kit. Помимо Джава-пакетов, которые нужны для разработки приложений, SDK содержит также образ операционной системы, на которой приложение будет эмулироваться.

file → Settings → Appearance & Behaivour → System Settings → Android SDK

Здесь все необходимые SDK можно легко скачать, просто поставив галочки и нажав **Apply**.

## 1.4. Эмулятор

Тестировать приложение можно на эмуляторе.

Tools → Android → AVD Manager

Здесь можно создавать виртуальные устройства: **Create Virtual Device**.

Можно отключать акселерометры и другие ненужные сенсоры: **New hardware profile**.
Лучше выбирать эмулятор на x86, а не на ARM, потому что быстрее.

## 1.5. Реальное устройство

Сначала нужно стать разработчиком. Обычно для этого можно на устройстве зайти в About и нажать семь раз на Build number.

После этого нужно включить на устройстве USB debugging (находится в Developer options).

Теперь подключаем устройство по USB и запускаем проект:

**Run** (Shift + f10)

Появляется окно, в котором нужно выбрать устройство. Реальные устройства отображаются в Connected Devices.

## 1.6. Структура проекта

- `manifest/AndroidManifest.xml`

  Мета-информация о проекте (имя приложения, иконка и т. п.)

  Сюда же мы можем добавить, например, такую строчку:

```
1 <uses-permission android:name="android.permission.CAMERA"></uses-permission>
```

  и это будет означать, что приложению для работы требуется камера.

  Также здесь перечисляются все Activity.

- `java/*`

  Собственно, джава-проект. Здесь есть классы, соответствующие каким-то активити, тесты.

- `res/*`

  Здесь то, что не является кодом: картинки (например, иконка), XML-файлы и др.

# Глава #2
# Layout, XML

3 октября

## 2.1. View

В приложении есть графические элементы: кнопки, чекбоксы и т. д. Все эти элементы являются наследниками класса View, а не Object (как в Java).

Некоторые наследники: ImageView, ProgressBar, TextView, ViewGroup.

## 2.2. ViewGroup

Часто графические элементы можно объединять в группы, например, в калькуляторе кнопки образуют таблицу. Для этого предназначен наследник класса View — класс ViewGro up. Его наследники, которые, собственно, и занимаются группировкой элементов, называются лейаутами. (На самом деле, layout — более общая вещь, но об этом позже). Лейауты отличаются друг от друга способом группировки элементов. Примеры лейаутов: AbsoluteLayout, LinearLayout, TableLayout.

## 2.3. LayoutParams

У лейаутов есть специальный внутренний класс, описывающий их параметры — LayoutParams. Популярные поля: layout_height, layout_width.

По понятным причинам, не стоит указывать значения в пикселях. Вместо этого лучше использовать dp – Density-independent Pixels.

Для значений ширины и высоты часто используются константы: match_parent и wrap_content. match_parent означает растянуть до размера родительского лейаута, wrap_content означает минимальный размер, нужный, чтобы в этот лейаут поместилось всё, что в нём лежит.

## 2.4. R

Activity - экран приложения. В своем корне содрежит один layout, который содержит все другие элементы. Можно в коде редактировать его, но лучше в XML, который потом собирается в класс R. Таким образом, можно работать в рантайме с теми графическими элементами, которые мы создали в XML. Чтобы можно было обратиться к нужному View, существует поле id. Чтобы получить, например, кнопку button, нужно сделать следующее:

```
1 Button b = (Button) findViewById(R.id.button);
```

## 2.5. strings.xml

Хорошим тоном считается сохранять строковые константы в отдельном файле `res/values/strings.xml`. Это позволяет разделить логику и данные. Для горизонтальной ориентации экрана можно задать отдельные layout, хотя андроид с этим и сам может справиться. При переходе в горизонтальную ориентацию телефон сам загрузит layout.

# Глава #3
# Виды Layout

3 октября

## 3.1. FrameLayout

Простейший вид разметки. Предполагается, что содержит всего один элемент, поэтому если лепим другие, то они накладываются один на другого. Элементы можно "прятать". Чтобы указать, какие видны в данный момент, используем:

```
android:visibility = "invisible" (visible, gone);
```

Gone элементы – те элементы, которые невидимы и "не занимают место". То есть если создать три кнопки подряд, сделать среднюю invisible, то у нас будет просто промежуток между третьей и первой. А если сделать ее еще и gone, то первая и третья теперь будут идти подряд, не обращая внимание не теоретическую кнопку между ними. Но мы все еще можем учитывать размер gone элементов при полном подсчете размера layout:

```
android:measureAllChildren = "true";
```

Можем указывать положение элементов относительно layout, параметры перечисляются через pipe:

```
android:layout_gravity = "top|right" (center, center_horizontal, etc);
```

Ну и задавать фон тоже бывает полезно. Для этого есть два атрибута. android:foreground принимает объект вида Drawable и отображает его, как изображение переднего фона, то есть выводит поверх всего содержимого. android:foregroundGravity выровнивает изображение переднего фона (сместить его с центра вправо вниз).

```
android:foreground = "#rgb" (@drawable/nameOfDrawable)
android:foregroundGravity = "top" (center, fill, etc);
```

## 3.2. AbsoluteLayout

Для позиционирования элементов указываются только координаты x и y. Лучше этим типом не пользоваться, так как могут с ним возникнуть всякие проблемы (в стиле того, что у разных экранов разное количество пикселей и тд).

```
android:layout_x
android:layout_y
```

## 3.3. LinearLayout

Выстраивает все элементы в ряд горизонтально или вертикально, мы это задаем сами. Задаем так:

```
android:orientation = "horizontal" (vertical);
```

Элементам можно сказать, как они разделят между собой пространство, для этого нужен вес. Веса это как бы соотношения, исходя из которых поделится пространство (даже если высота объекта 0, то из-за веса он может быть большим). А можно указать общий вес всего лэйаута, вместо того чтобы указывать его для ВСЕХ параметров.

```
android:layout_weight = "1";
android:weightSum = "100";
```

Есть разделители элементов (параметр лэйаута, все элементы разделит этим разделителем). Например, если в нашем лэйауте три кнопки, а разделитель – горизонтальный прямоугольник в цвет фона, то мы увидим три кнопки с промежутками между ними.

```
android:devider = "@drawable/blackDevider";
```

Сам файл blackDevider - это .xml файл с описанием разделителя.

## 3.4. TableLayout

Элементы организуются в строки и столбцы. Количество столбцов – максимальное количество элементов в одной из строк. Если в какой-то строке элементов меньше, чем максимум, то в конце будет просто пустота. Ширина столбца – ширина самого широкого элемента в нем. Каждая строка таблицы – элемент TableRow, в котором и перечисляются все элементы. Несколько ячеек подряд можно превратить в одну длинную:

```
android:layout_span = "4";
```

Атрибут stretchColumns указывается у самого TableLayout, если в строке останется незанятое место, то в этом параметре перечисляются ячейки, которые поделят это место между друг другом. А * ставим, чтобы выполнялось для всех (нумерация ячеек как в матрице).

```
android:stretchColumns = "0,2,3" (*);
```

Ну и еще парочка атрибутов. shrinkColumns – индексы столбцов, где элементы будут сжиматься, чтобы не стать шире размера стобца (если длинное название, то не хотим ему давать кушать место, указываем параметр, содержимое кнопки начнет переноситься). collapseColumns – каким стобцам стоит пропасть. В таблице они останутся, но видны не будут. Вроде потом им можно сказать, чтобы они опять стали видны.

```
android:shrinkColumns = "1,2";
android:collapseColumns = "0,1,3,6";
```

## 3.5. RelativeLayout

Указываем положение элементов относительно лэйаута и друг друга. Пишется это у самого элементов. Позиционирование относительно layout:

```
android:layout_allignParentTop = "true";
android:layout_centerHorizontal = "true";
android:layout_centerInParent = "true";
```

Позиционирование относительно других элементов (id – айди элемента, относительно которого все делается):

```
android:layout_above = "@id/name";
android:layout_toLeftOf = "@id/name";
```

Выравнивание относительно других элементов:

```
android:layout_allignLeft = "@id/name";
android:layout_allignBaseline = "@id/name";
```

# Глава #4
# Материалы

3 октября

## 4.1. Полезные ссылки

На вики-странице, лежат все материалы с первого "семинара".