

Министерство науки и высшего образования РФ  
Южно-Российский государственный политехнический  
университет (НПИ) имени М.И. Платова

## **РАЗРАБОТКА МОБИЛЬНЫХ ПРИЛОЖЕНИЙ**

*Методические указания  
к выполнению практических занятий  
и лабораторных работ*

Новочеркасск  
ЮРГПУ (НПИ)

2023

## ОГЛАВЛЕНИЕ

|  |    |
|--|----|
| ЛАБОРАТОРНАЯ РАБОТА № 1 ИНТЕРФЕЙС ANDROID STUDIO.....                | 1  |
| ЛАБОРАТОРНАЯ РАБОТА № 2 ПОЛЯ ВВОДА.....                              | 2  |
| ЛАБОРАТОРНАЯ РАБОТА № 3 ОБРАБОТЧИКИ СОБЫТИЙ.....                     | 3  |
| ЛАБОРАТОРНАЯ РАБОТА № 4 СТРОКОВЫЕ И ЦВЕТОВЫЕ<br>РЕСУРСЫ.....         | 4  |
| ЛАБОРАТОРНАЯ РАБОТА № 5 ПЕРЕКЛЮЧЕНИЕ МЕЖДУ<br>СТРАНИЦАМИ.....        | 5  |
| ЛАБОРАТОРНАЯ РАБОТА № 6 ПЕРЕДАЧА УПРАВЛЕНИЯ МЕЖДУ<br>СТРАНИЦАМИ..... | 6  |
| ЛАБОРАТОРНАЯ РАБОТА № 7 ОБМЕН ДАННЫХ МЕЖДУ<br>СТРАНИЦАМИ.....        | 7  |
| ЛАБОРАТОРНАЯ РАБОТА № 8 РАБОТА С ИЗОБРАЖЕНИЯМИ.....                  | 8  |
| ЛАБОРАТОРНАЯ РАБОТА № 9 АУДИО ПЛЕЕР.....                             | 9  |
| ЛАБОРАТОРНАЯ РАБОТА № 10 ВИДЕО ПЛЕЕР.....                            | 10 |
| ЛАБОРАТОРНАЯ РАБОТА № 11 КАДРОВАЯ АНИМАЦИЯ.....                      | 11 |
| ЛАБОРАТОРНАЯ РАБОТА № 12 АНИМАЦИЯ СВОЙСТВ.....                       | 12 |
| ЛАБОРАТОРНАЯ РАБОТА № 13 АНИМАЦИЯ ГЕОМЕТРИЧЕСКИХ<br>ОБЪЕКТОВ.....    | 13 |
| ЛАБОРАТОРНАЯ РАБОТА № 14 ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ.....                 | 14 |
| БИБЛИОГРАФИЧЕСКИЙ СПИСОК.....  | 15 |

### Список лабораторных (практических) работ

1. Знакомство с интерфейсом Android Studio, содержанием проекта и основными элементами управления.
2. Создание приложения с использованием элементов ImageButton, TextView, EditText и обработчика нажатия на кнопку.
3. Изучение способов обработки нажатия на кнопку и создание приложения для подсчёта количества нажатий на кнопку.
4. Изучение правил создания строковых и цветовых ресурсов проекта, создание приложения с использованием этих ресурсов.
5. Создание приложения с переключением на другой экран.
6. Создание приложения с переключением на другой экран с передачей данных на вторую страницу.
7. Создание приложения с переключением на другой экран с передачей данных на вторую страницу и получением данных из второй страницы.
8. Создание приложения с выводом и масштабированием изображения.
9. Создание приложения с аудио-плеером.
10. Создание приложения с видео-плеером.
11. Создание приложения с кадровой анимацией.
12. Создание приложения с анимацией свойств.
13. Разработка приложения с созданием и анимацией геометрических объектов.
14. Описание индивидуального задания.

# ЛАБОРАТОРНАЯ РАБОТА № 1

## ИНТЕРФЕЙС ANDROID STUDIO

*Знакомство с интерфейсом Android Studio, содержанием проекта и основными элементами управления*

**Цель занятия:** знакомство с инструментами разработки Android-приложений.

### *Методические указания*

**Android** - операционная система, которая широко используется в мобильных устройствах: смартфонах и планшетных компьютерах. Это бесплатная операционная система, основанная на Linux с интерфейсом программирования Java. Приложение для Android пишется на языке Java или Kotlin. Наиболее популярной средой разработки в настоящее время является Android Studio [1-3].

**Android Studio** - среда разработки под Android, основанная на IntelliJ IDEA. Она предоставляет интегрированные инструменты для разработки и отладки. Для отладки приложений используется эмулятор телефона - виртуальная машина, на которой будет запускаться созданное приложение.

Для того, чтобы установить Android Studio на Windows, следует выполнить следующие действия:

- Загрузите актуальную версию Android Studio с официального сайта <https://android-studio.ru/> и затем запустите сохраненный файл.
- Установите Android Studio и все необходимые инструменты SDK по инструкциям мастера установки.

Отдельные инструменты и другие пакеты SDK сохраняются вне каталога приложений в Android Studio. Например: \Users\<user>\sdk\

При настройке Android SDK следует установить пакеты соответствующего уровня, который зависит от версии OS Android, для которой предполагается разрабатывать приложения. Следует учитывать, что при установке каждого из них потребуется значительный объем памяти. Например, для API 29 – примерно 260 Мб.

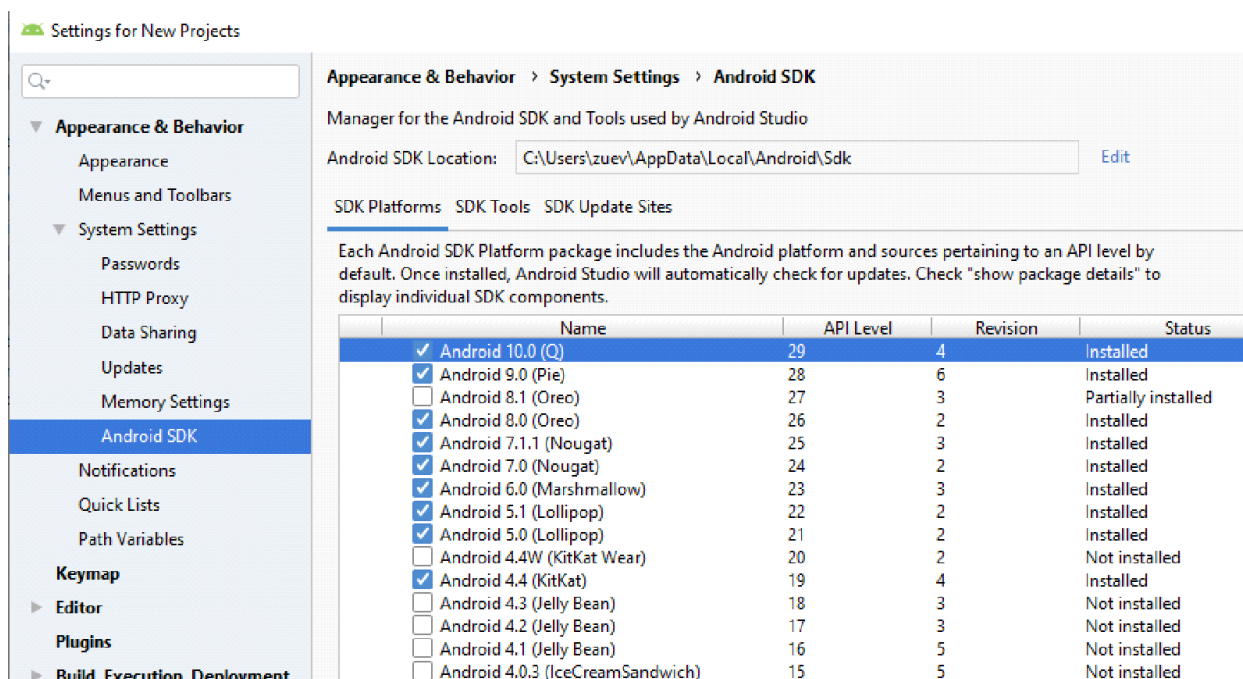


Рисунок 1.1 Менеджер установки Android SDK

При создании проекта в нем создается по меньшей мере один модуль. В модуле содержится описание на языке xml изображения на экране приложения и код на языке Java. Каждый модуль по сути является приложением, а проект - контейнер для модуля. Если в проекте содержится несколько модулей, то следует указать, какой из них надо запустить.

При выполнении студентом комплекса лабораторных работ целесообразно создать один проект, где каждый из модулей будет соответствовать приложению для очередной темы лабораторной работы.

Рассмотрим этапы создания нового проекта. После запуска Android Studio выбираем ***Start a New Android Studio Project***, появится диалоговое окно мастера. Предлагается несколько шаблонов проекта, где выбираем ***Empty Activity***.

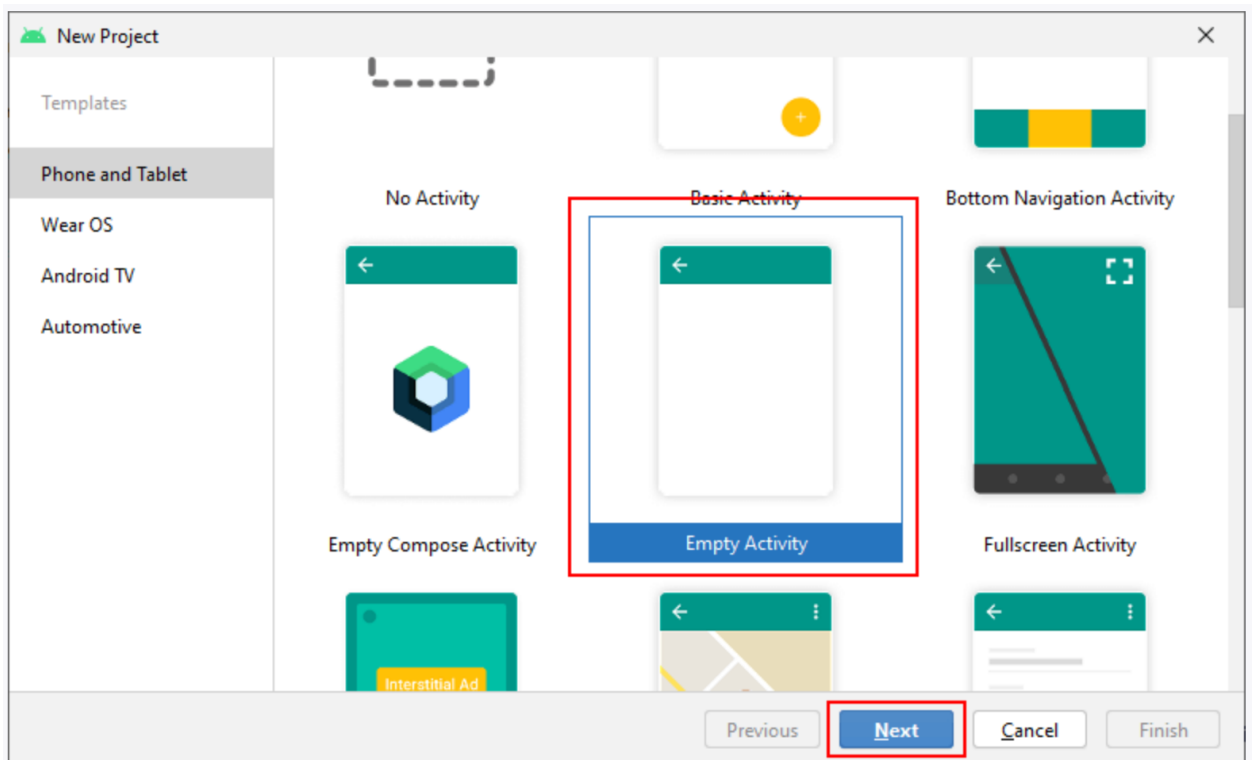


Рисунок 1.2 Менеджер создания проекта Android Studio

Далее предлагается ввести следующую информацию:

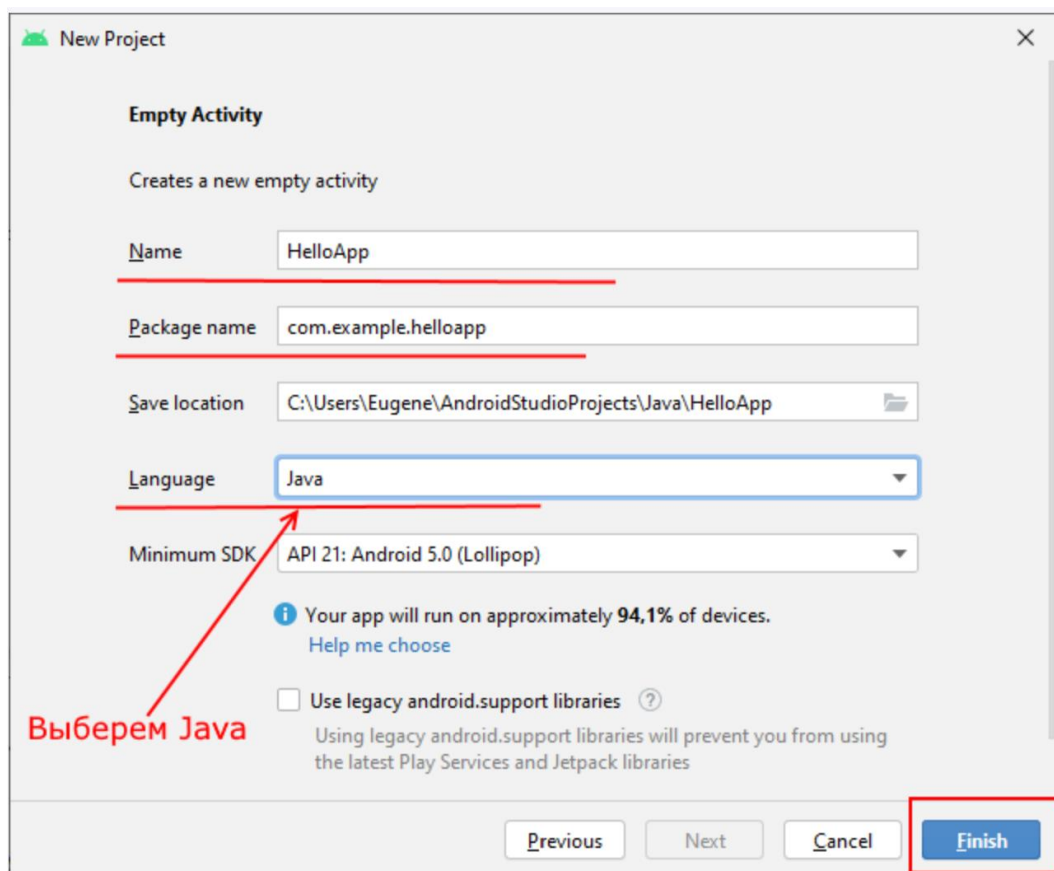


Рисунок 1.3 Настройка проекта Android Studio

- Поле **Name** - имя, которое будет отображаться в заголовке приложения (по умолчанию MyApplication).

- Поле **Package name** формирует специальный java-пакет на основе вашего имени из предыдущего поля.

- Поле **Save location** позволяет выбрать место на диске для создаваемого проекта.

После нажатия на кнопку **Next** переходим к следующему окну. Здесь выбираем типы устройств, под которые будем разрабатывать приложение. Выбираем смартфоны. Далее необходимо выбрать внешний вид экрана приложения. Выберем, например, вариант **Blank Activity**. После нажатия кнопки **Finish** формируется проект и создаётся необходимая структура из различных файлов и папок.

В левой части среды разработки на вкладке Android появится иерархический список из папок, которые относятся к проекту. Вкладка Android содержит две основные папки: app и Gradle Scripts. Первая папка app является отдельным модулем для приложения и содержит все необходимые файлы приложения - код, ресурсы картинок и т.п. Вторая папка служит для различных настроек для управления проектом. Раскрываем папку app. В ней находятся три папки: manifest, java, res.

Папка manifest содержит единственный файл манифеста AndroidManifest.xml. В этом файле должны быть объявлены все активности, службы, приёмники и контент-провайдеры приложения. Также он должен содержать требуемые приложению разрешения. «AndroidManifest.xml» можно рассматривать, как описание для развертывания Android-приложения.

Папка java содержит три подпапки - рабочую и для тестов. Рабочая папка имеет название пакета и содержит файлы классов. Сейчас там один класс MainActivity.

Папка res содержит файлы ресурсов, разбитых на отдельные подпапки.

Запуск программы осуществляется выбором команды Run в пункте меню Run. После этого в эмуляторе на экране появится окно приложения с надписью «Hello World!» и заголовком программы.

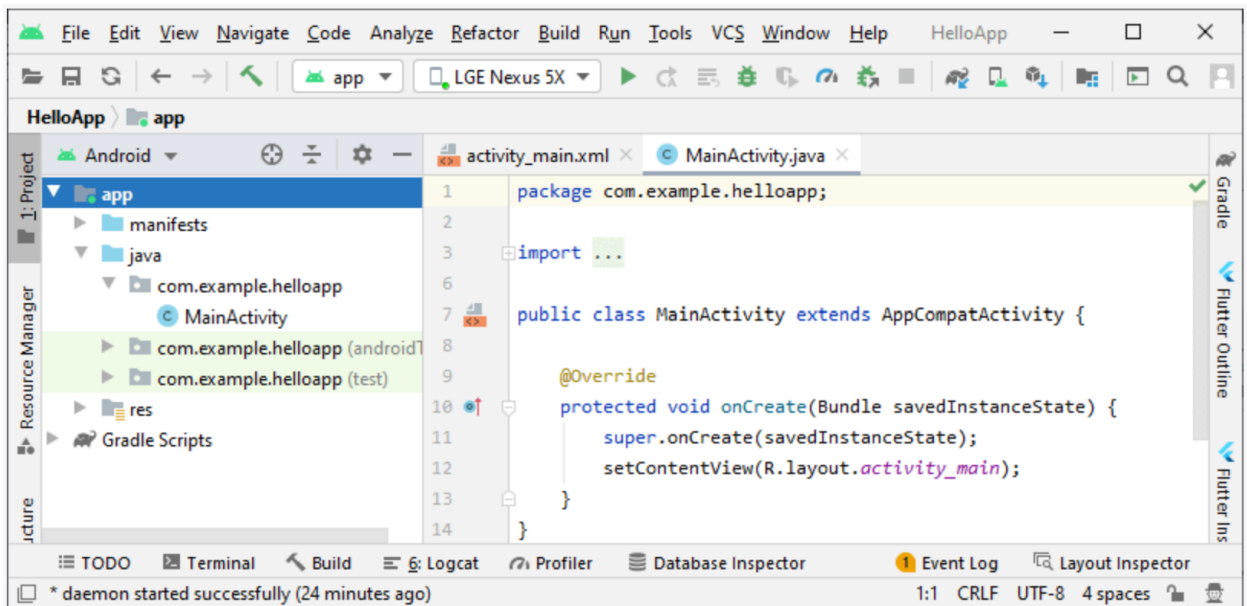


Рисунок 1.4 Окно Android Studio с новым проектом

Итак, проект создан. Теперь создадим в проекте свой модуль (приложение) для этого текущего урока. Эта процедура будет частично похожа на создание проекта, но с небольшими отличиями. Чтобы создать модуль – в меню выбираем **File -> New -> New module**. Тип модуля выбираем **Phone and Tablet Application** и жмем **Next**.

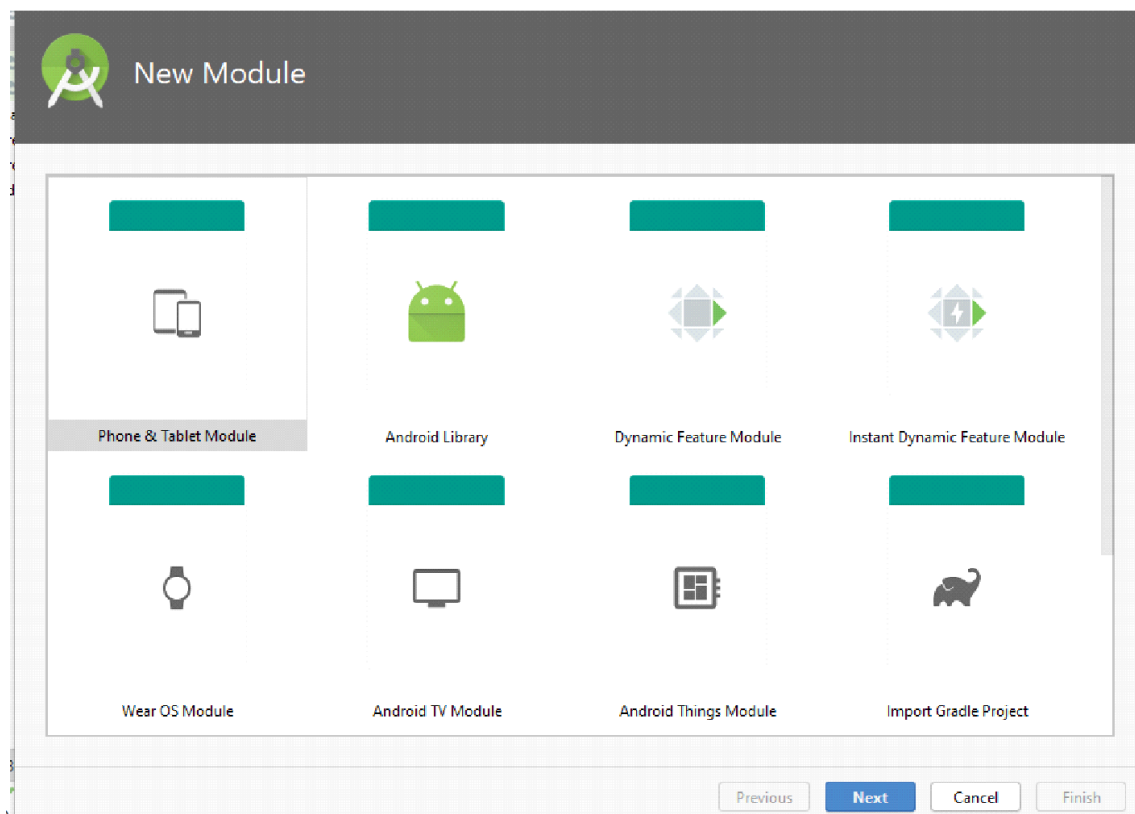


Рисунок 1.5 Окно Android Studio с выбором модуля



Далее надо заполнить поля:

**Application/Library name** – непосредственно имя приложения, которое будет отображаться в списке приложений в смартфоне.

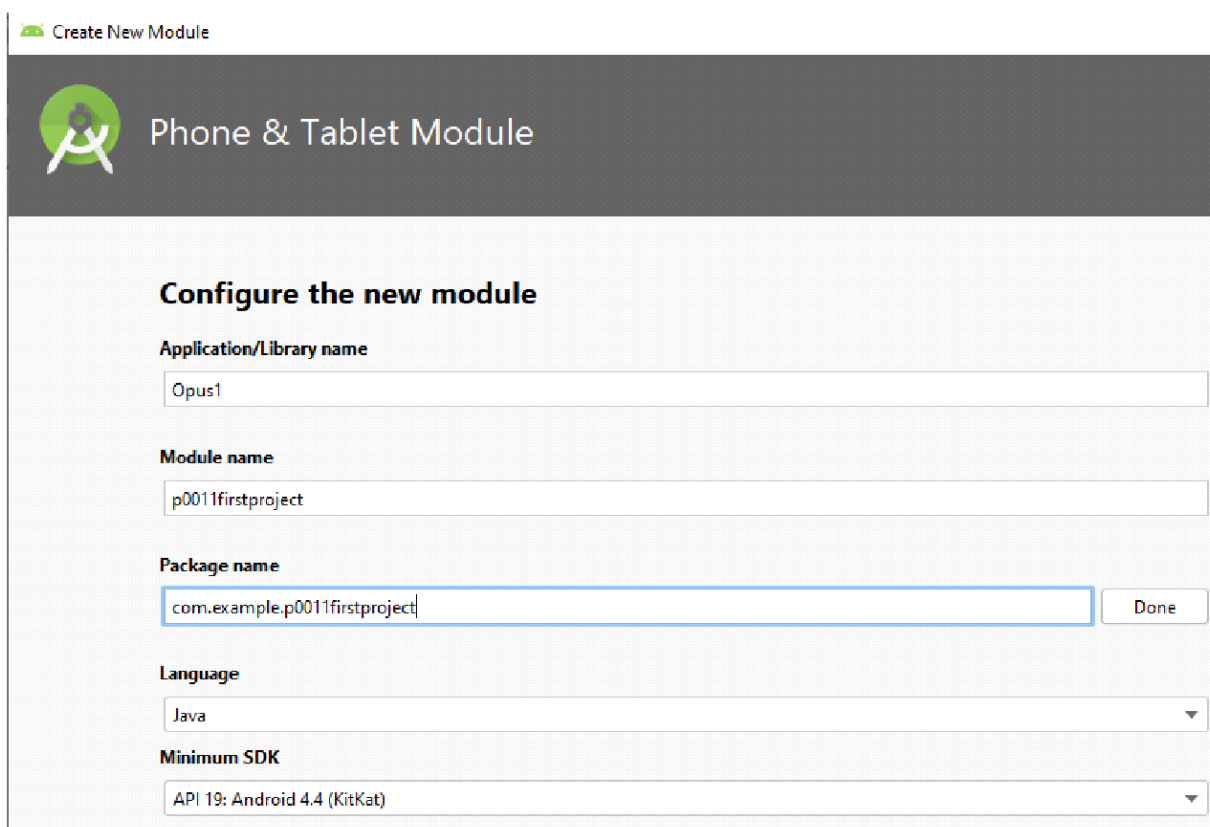
**Module name** – это название модуля. Это название будет отображаться слева в списке модулей, там, где сейчас есть app. Предлагается такой шаблон для названия модулей:

p<номер занятия><номер проекта в этом занятии>.

На номер занятия выделим три цифры, а на номер проекта – одну. Также, будем добавлять название приложения, например, - firstproject. Все это пишем маленькими буквами и без пробелов. Получится такое имя модуля: *p0011firstproject*.

**Package name** – имя пакета отредактируем вручную, нажав *edit* справа. Оставим там ru. siurgtu и добавим точку и имя модуля.

**Minimum SDK** оставляйте без изменений и нажимайте кнопку *Next*.



Create New Module

Phone & Tablet Module

**Configure the new module**

Application/Library name  
Opus1

Module name  
p0011firstproject

Package name  
com.example.p0011firstproject Done

Language  
Java

Minimum SDK  
API 19: Android 4.4 (KitKat)

Рисунок 1.6 Создание нового модуля

Далее выберите *Empty Activity* и нажимайте *Next*. В следующей форме ничего менять не нужно; нажимайте кнопку *Finish*. В итоге модуль (*Package*)

будет создан, мы увидим его в списке слева. На скриншоте, который показан ниже, это *p0031firstproject* - значение, которое было указано в поле *Module name*.

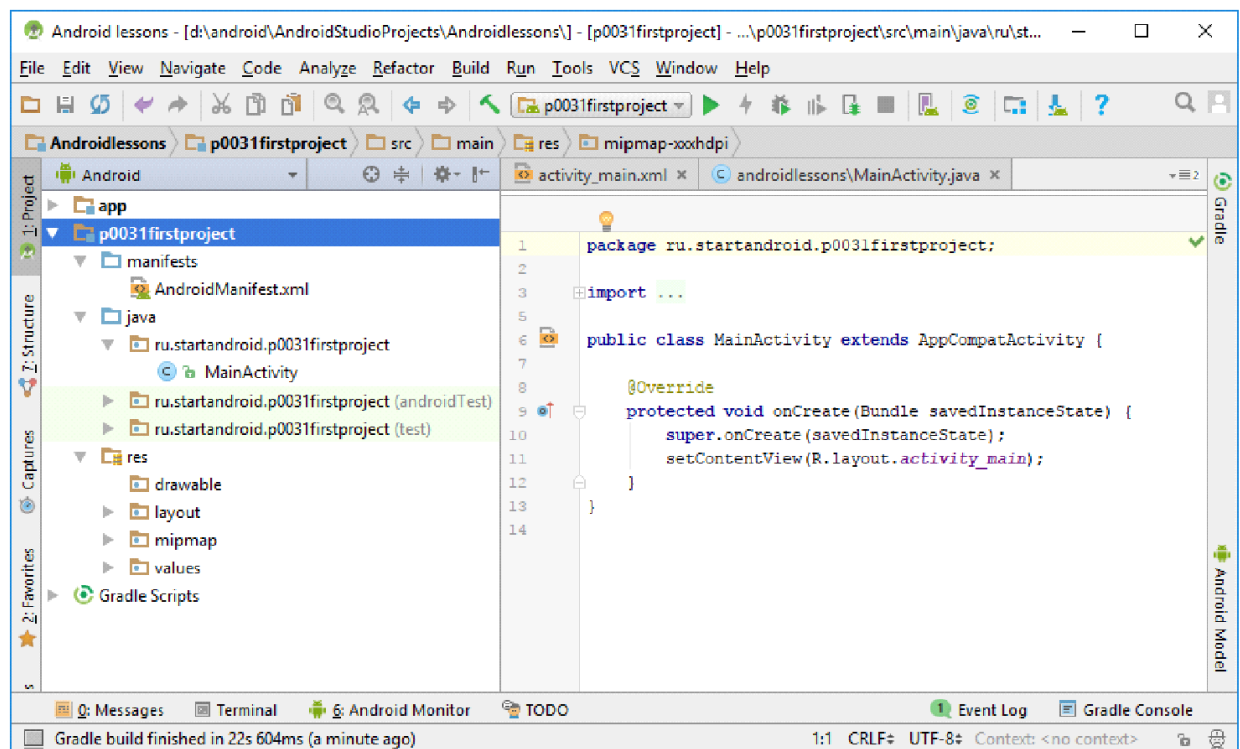


Рисунок 1.7 Подготовка к запуску

Запустим наше первое приложение! Для этого надо выбрать соответствующий ему модуль в выпадающем списке сверху.

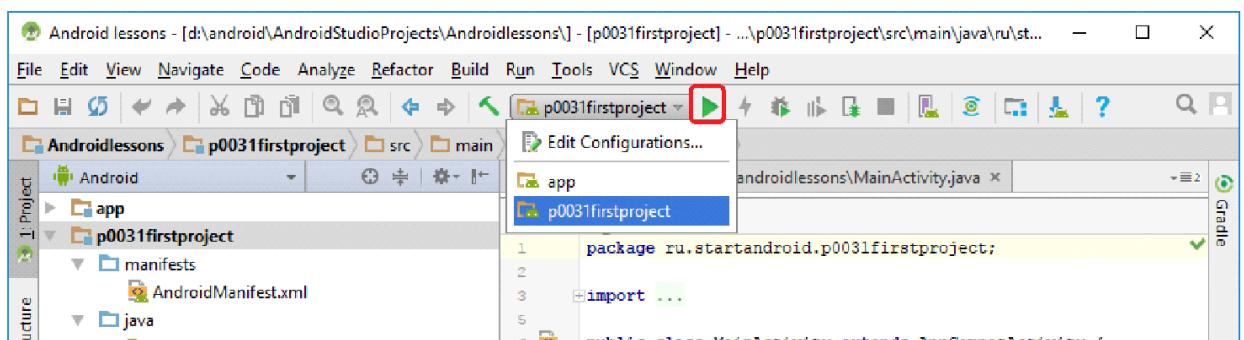


Рисунок 1.8 Запуск приложения

Чтобы запустить приложение, нужно реальное Android-устройство или эмулятор. Если через шнур подключен смартфон или планшет, то приложение будет запущено на нем.

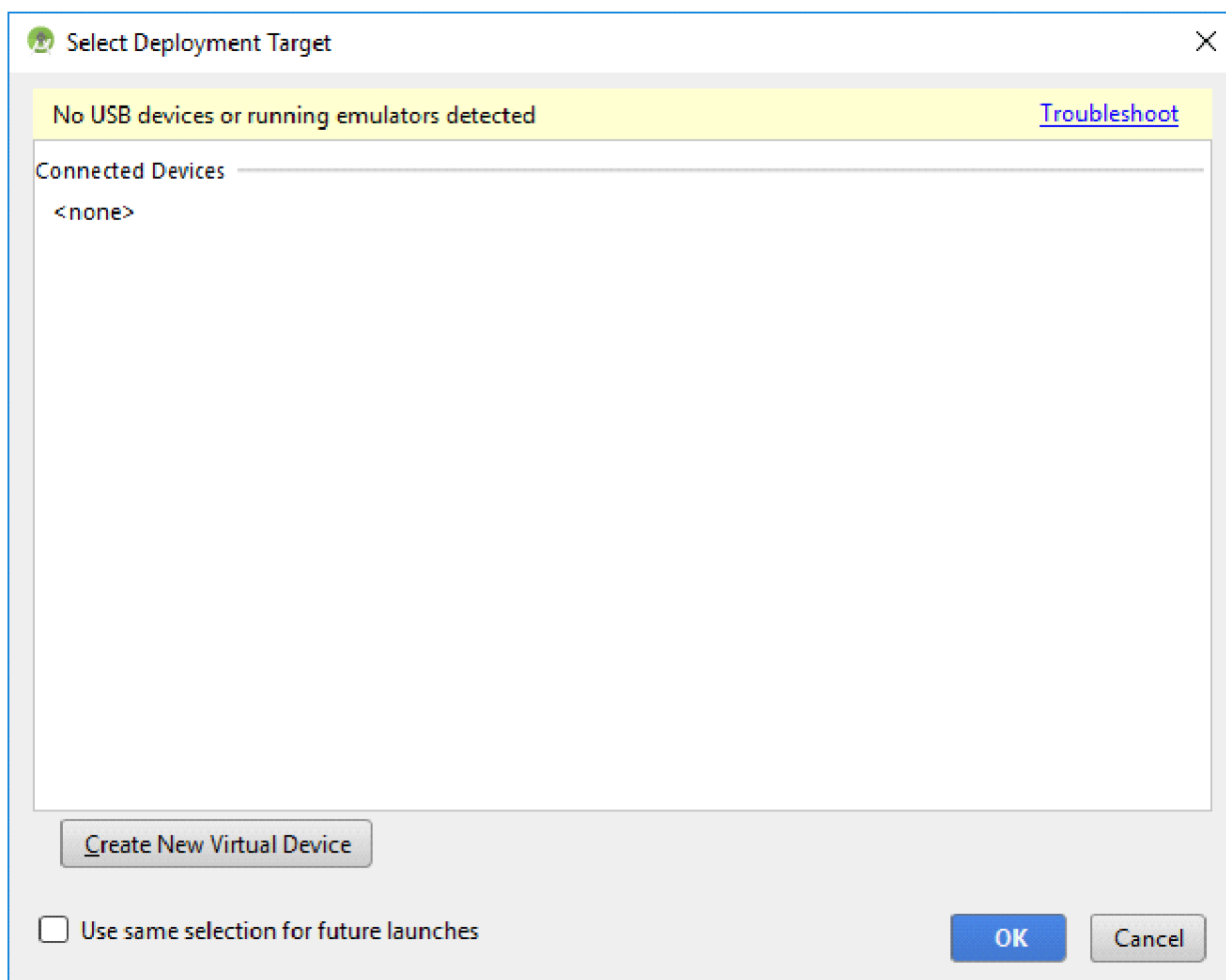


Рисунок 1.9 Предупреждение об отсутствии подключенного Android-устройства

Для выбора эмулятора следует нажать кнопку Create New Virtual Device.

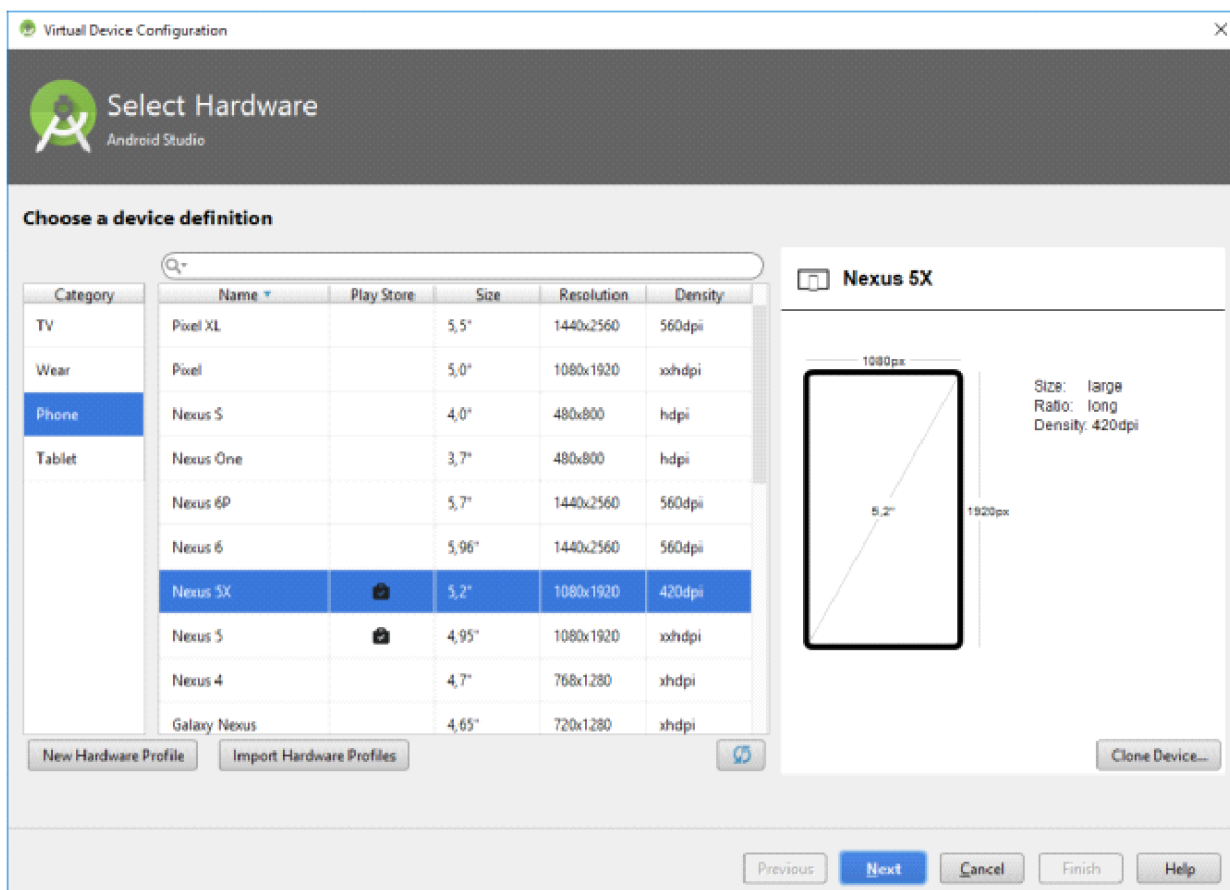


Рисунок 1.10 Настройка эмулятора Android-устройства

Далее переходите на вкладку x86 Images, где выбирайте образ, в названии которого нет слова Download. То есть он уже загружен, мы можем его использовать. Затем следует указать название эмулятора и поменять его настройки.

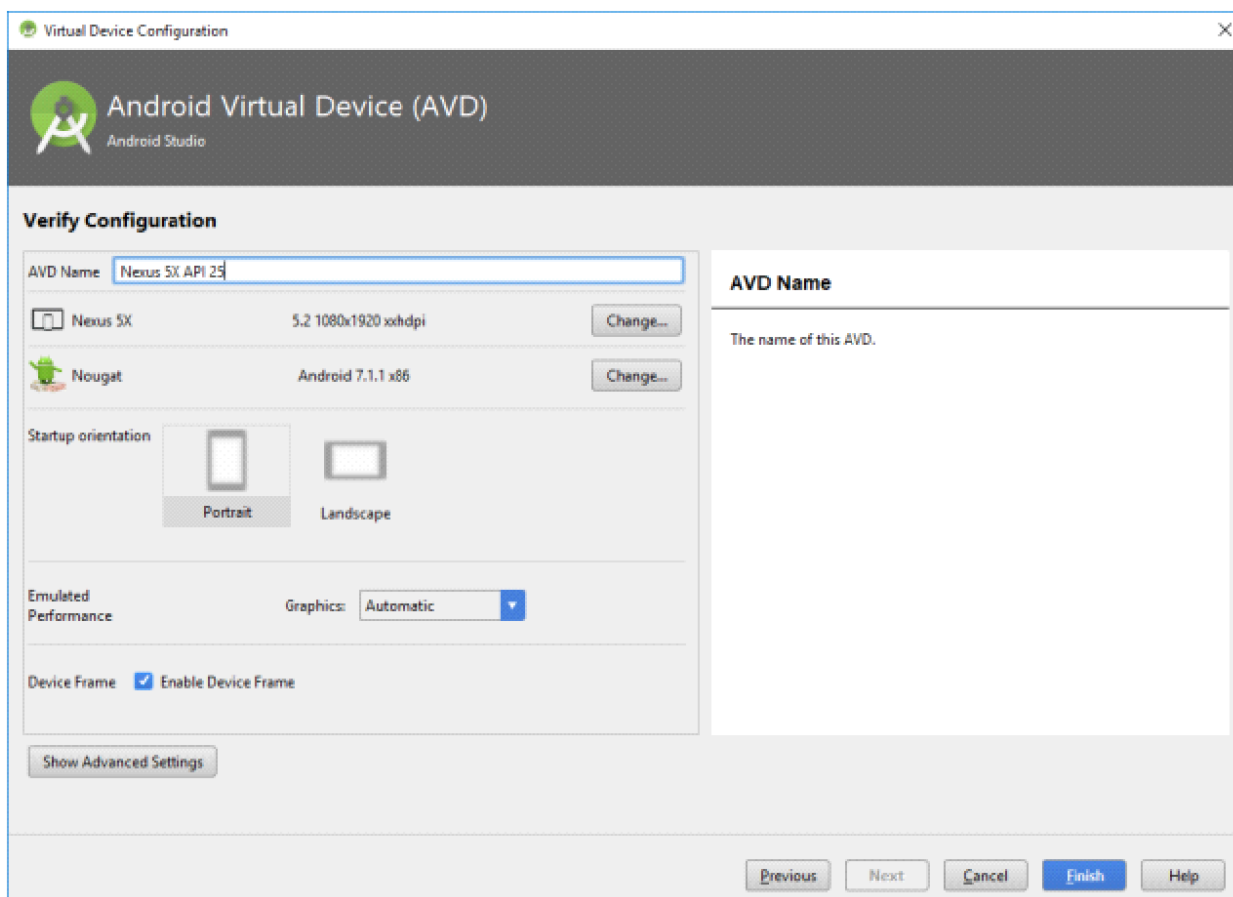


Рисунок 1.11 Настройка эмулятора Android-устройства

В итоге, в списке устройств появляется только что созданный эмулятор. Теперь его можно использовать для запуска приложения.

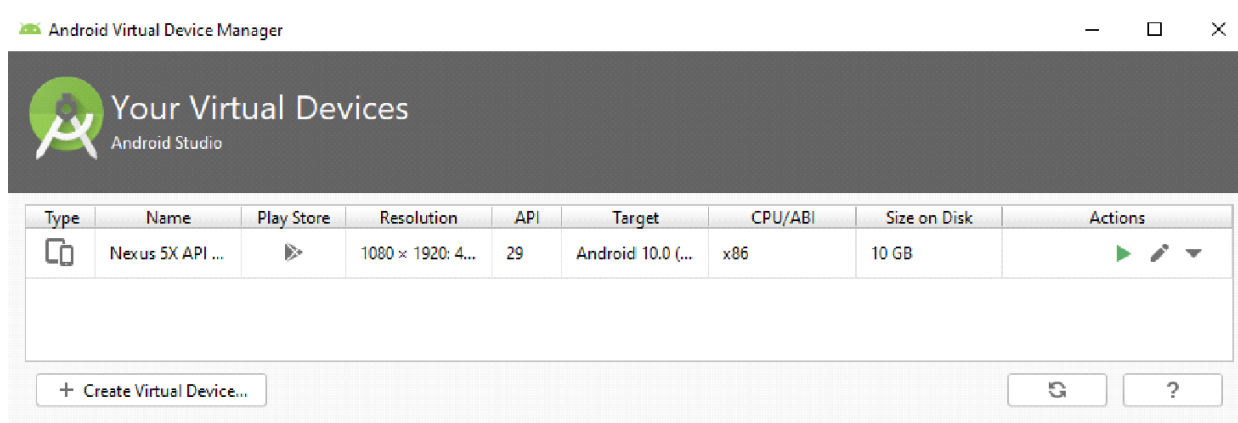


Рисунок 1.12 Настройка эмулятора Android-устройства

В результате, на экране появится эмулятор выбранного мобильного устройства, в котором будет запущено приложение. Чтобы создать эмулятор телефона, можно выбрать в меню **Android Studio** пункты **Tools / AVD Manager / Create Virtual Device**.

## **Оформление отчета**

**Цель работы:** изучить технологию создания приложений в среде разработки Android Studio

### **Программа выполнения работы:**

1. Изучить методические указания к практическому занятию.
2. Запустить на выполнение Android Studio.
3. Создать в среде Android Studio проект, разработанный на практическом занятии.
4. Создать в соответствующих директориях файлы проекта.
5. Запустить созданное приложение в эмуляторе Android и наблюдать за появлением этого приложения и результатов его работы в окне приложений эмулятора.

### **Контрольные вопросы**

1. Какие возможности имеет среда разработки приложений Android Studio?
2. Этапы создания нового проекта?
3. В каких папках хранятся файлы проекта?
4. Что содержится в манифесте проекта?
5. Что нужно сделать для создания модуля?

### **Содержание отчета:**

1. Название лабораторной работы и её номер
2. Цель работы.
3. Ход работы, включающий в себя экранные копии (скриншоты) для созданного приложения в Android Studio для каждого из ключевых пунктов создания приложения. Результаты работы приложения в эмуляторе мобильного устройства или на устройстве в виде скриншотов.
4. Вывод в форме: “В результате выполнения лабораторной работы было/были изучено/изучены... проведено ознакомление с ...”.

**\*Примечание** Для данной лабораторной работы вывод должен иметь следующую формулировку: “В результаты выполнения лабораторной работы было произведено ознакомление с технологией создания приложений в среде разработки Android Studio”. Таким образом, в вывод должен содержать ту же формулировку, которая содержится в пункте лабораторной “Цель работы”. **Вывод должен быть обезличен** и не содержать формулировок: “Мы / я / он / они...”.

### **Оформление**

**Важно!** Лабораторные работы должны быть выполнены шрифтом Times New Roman 14 кегль, отступ 1.5. Каждый рисунок должен быть подписан в соответствии с номером лабораторной и номером рисунка в ней. *Например*, второй рисунок в пятой лабораторной должен иметь номер 5.2, а седьмой рисунок во второй лабораторной 2.7 и т.п. Помимо номера рисунок должен содержать подпись, кратко передающая суть изображения. Каждая страница в итоговом отчете должна быть пронумерована!

## ЛАБОРАТОРНАЯ РАБОТА № 2

### ПОЛЯ ВВОДА

*Создание приложения с использованием элементов `ImageButton`, `TextView`, `EditText` и обработчика нажатия на кнопку.*

**Цель занятия:** Знакомство с основными элементами в редакторе XML ANDROID STUDIO.

#### *Методические указания*

В данной лабораторной работе нам пригодится файл ***activity\_main.xml***, в котором мы создадим поля для взаимодействия с программой. Файл имеет расширение **XML**, *eXtensible Markup Language* — «расширяемый язык разметки» и представляет собой текстовый файл, в котором хранится представление основного (main) окна проекта.

**XML** поддерживает обмен информацией между компьютерными системами, такими как веб-сайты, базы данных и сторонние приложения. Предопределенные правила упрощают передачу данных в виде XML-файлов по любой сети, поскольку получатель может использовать эти правила для точного и эффективного чтения данных.

Если проводить аналогию с Windows, то ANDROID приложение состоит из окон, называемых Activity. В конкретный момент времени обычно отображается одно Activity и занимает весь экран, а приложение переключается между ними. В качестве примера можно рассмотреть почтовое приложение. В нем одно Activity – список писем, другое – просмотр письма, третье – настройки ящика. При работе вы перемещаетесь по ним.

Содержимое Activity формируется из различных компонентов, называемых View. Самые распространенные View - это кнопка, поле ввода, чекбокс и т.д.



Примерно это можно изобразить так:

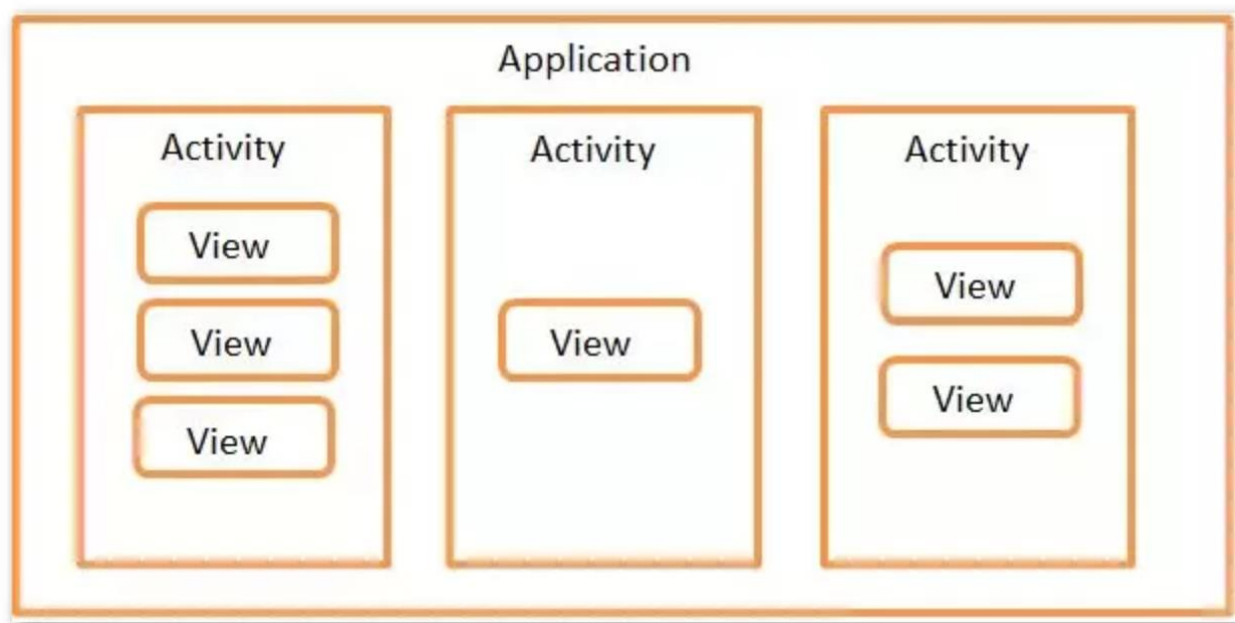


Рис 2.1 Структура графической части ANDROID – проекта

Вложенность компонентов view может быть сколь угодно глубокой. Обычно View размещают в контейнер ViewGroup. Самый распространенный пример ViewGroup – это Layout. Layout бывает различных типов и отвечает за позиционирование элементов, как будут расположены его дочерние View на экране (таблицей, строкой, столбцом и т.д.).

Итак, создадим приложение из одного окна (activity) в котором будут элементы *ImageButton*, *TextView*, *EditText*, кнопка и обработчик нажатий на кнопку. Задача состоит в том, чтобы по нажатию на кнопку на экране появлялся текст, введенный в поле ввода, и картинка.

Создадим новый проект на **java**, выбрав **Empty view activity**:

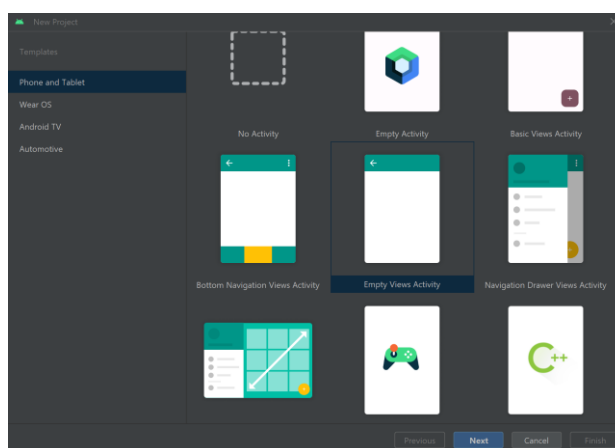


Рис 2.2 Empty views activity

Выберем файл *activity\_main.xml*. Его можно редактировать, как в редакторе кода, во вкладке code, так и графически, во вкладке design.

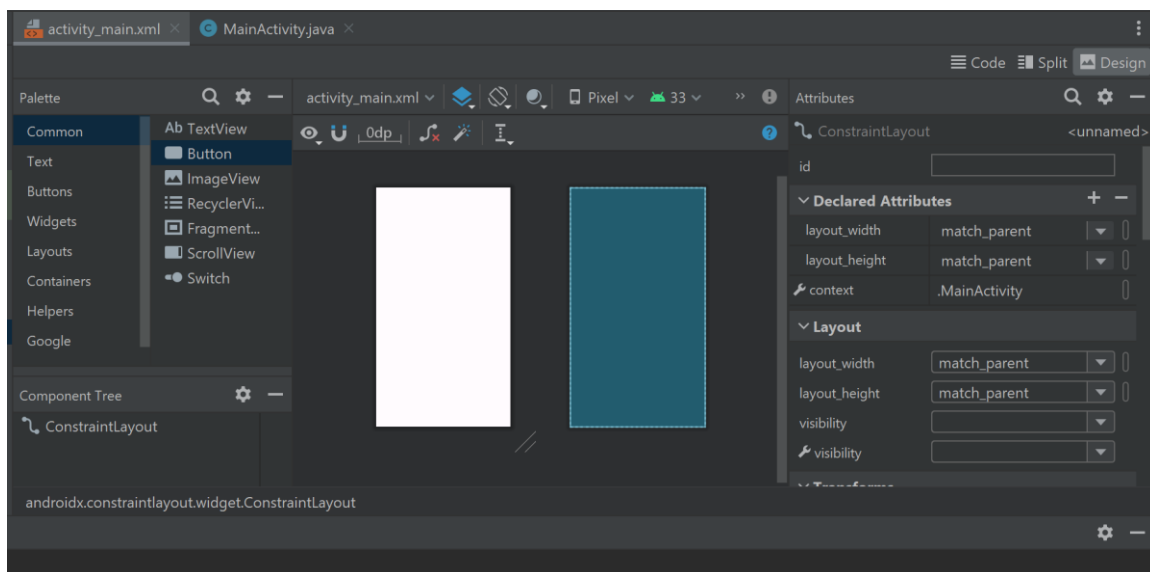


Рис 2.3 Пустой activity\_main.xml

Вытащим элементы *ImageButton*, *TextView* и *Button* в окно редактора, как показано на рисунке 2.3.

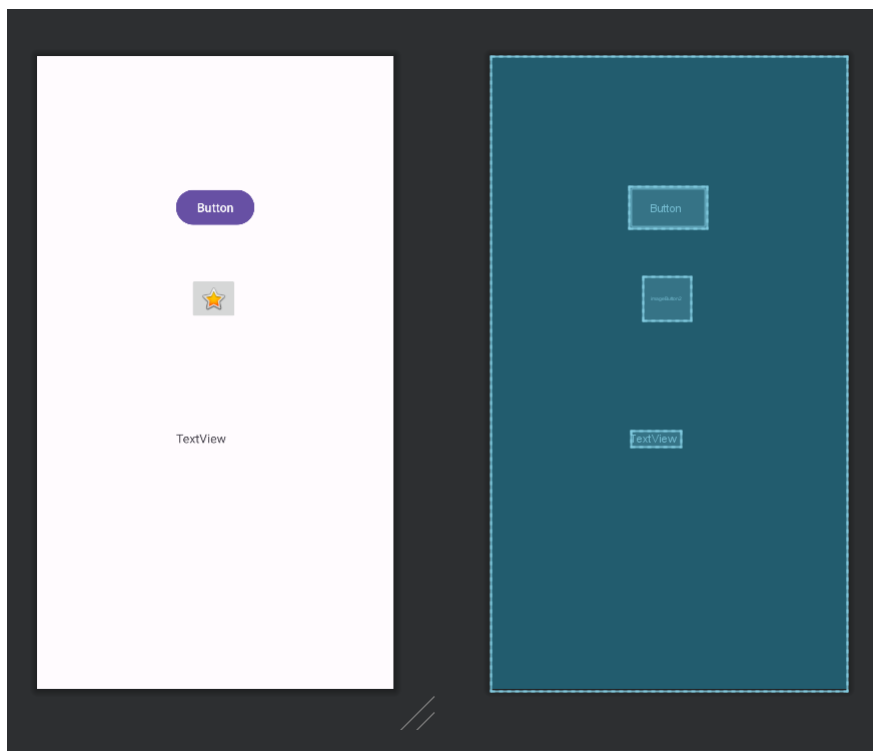


Рис 2.4 activity\_main.xml с тремя view компонентами

Запустим проект и увидим, что все компоненты находятся вовсе не там, где мы их оставили при редактировании.

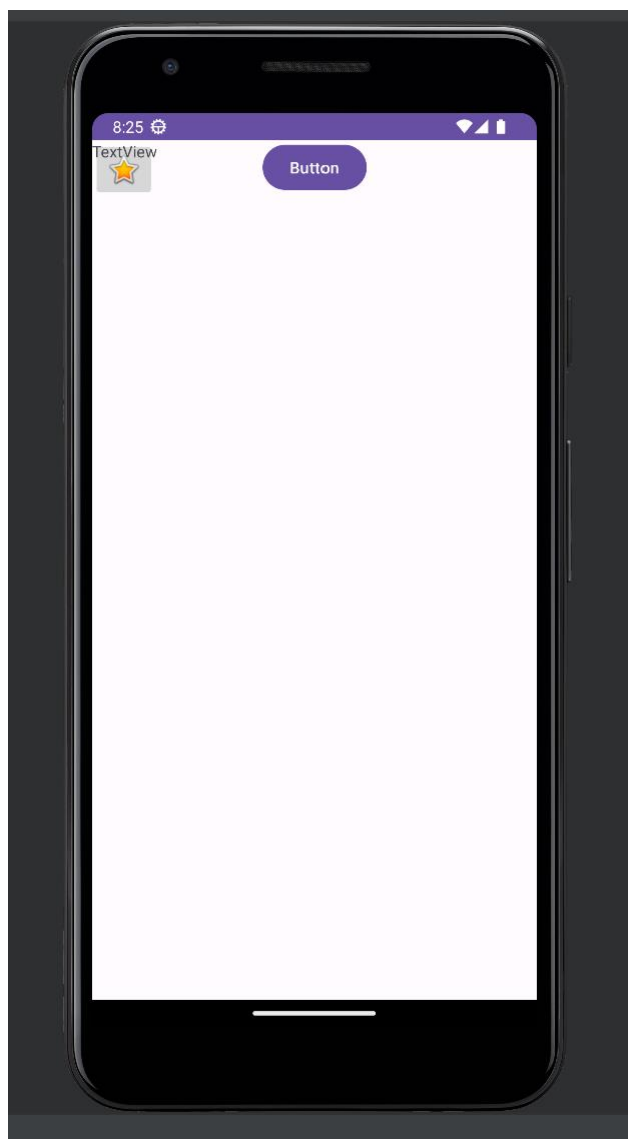


Рис 2.5 Окно эмулятора с тремя непривязанными view компонентами

Для того, чтобы компоненты оставались в той позиции, в которой мы их оставили при редактировании, необходимо в дизайнере сцены выделить все элементы и нажать кнопку “infer constraints”. Она “привяжет” все элементы друг к другу, а какие-то из элементов сделает “якорями” – за них будут цепляться все остальные, а сам якорь, в свою очередь, за сцену(окно) activity или за контейнер, если он есть. В нашем случае, элементы не вложены в контейнер.

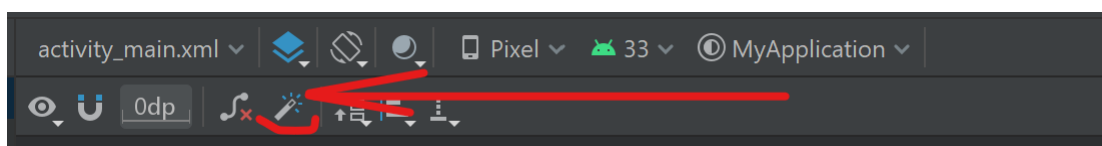


Рис 2.6 Окно дизайнера с кнопкой привязки компонентов

В данном случае, *Button* связана с *ImageButton* и с окном, а *TextView* связан с окном отдельно.

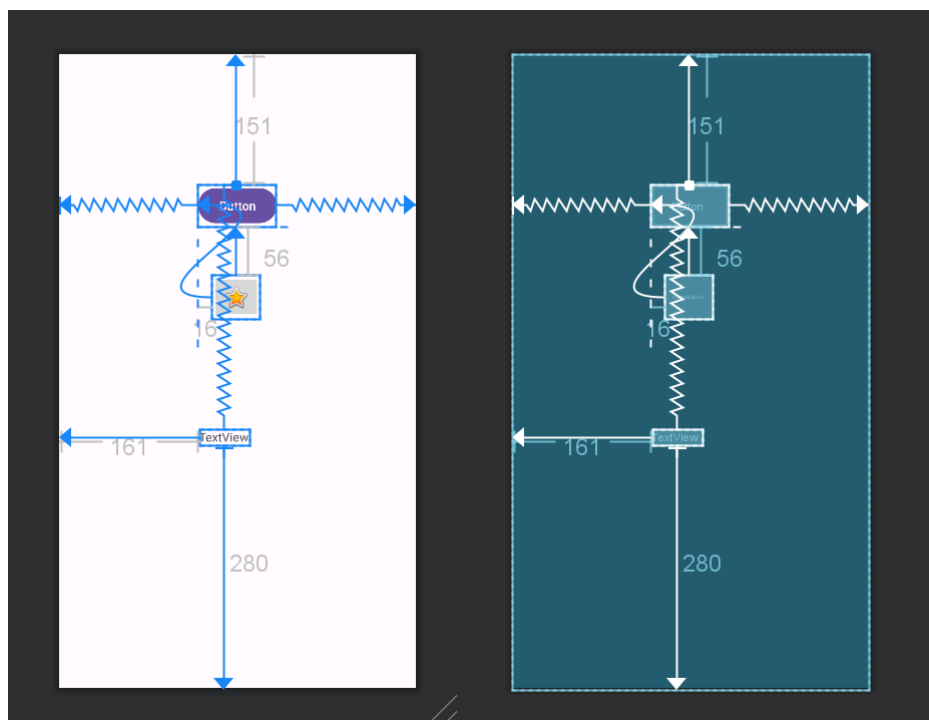


Рис 2.7 Окно дизайнера с привязанными компонентами

Обратите внимание, что привязанные друг к другу компоненты двигаются нераздельно. Привязку можно удалить, как отдельно для каждого компонента, так и удалить сразу все привязки.

Теперь, добавим в скрипт *MainActivity.java* следующий код:

Для начала, создадим три переменные ссылочного типа внутри класса *MainActivity*:

```
TextView myTextView;
```

```
Button myButton;
```

```
ImageButton imageButton;
```

Переменные ссылочного типа хранят ссылку на объекты, так как сами объекты могут быть очень больших размеров и их размер может меняться во время выполнения программы. Объект – сложная переменная, состоящая из более простых типов данных и созданная по шаблону, называемому классом.

В эти переменные мы запишем ссылки на объекты из файла *activity\_main.xml*. Ссылку на объект можно получить через функцию *findViewById()*.

По умолчанию в скрипте есть функция(метод) **onCreate** – она выполняется один раз во время старта программы. В ней присутствует следующий код:

```
super.onCreate(savedInstanceState); // обращается к методу onCreate
родительского класса AppCompatActivity
```

```
setContentView(R.layout.activity_main); // функция setContentView
связывается с окном(активностью) activity_main.xml
```

Дополним метод следующим кодом:

```
myTextView = findViewById(R.id.textView2);

myButton = findViewById(R.id.button3);

imageButton = findViewById(R.id.imageButton2);
```

**Важно!** Вместо выделенных красным шрифтом названий нужно вписать свои поля из файла activity\_main.xml. Обращение к полям происходит по id(identifier) – уникальный номер или строка, например: android:id="@+id/**imageButton2**" – жирным выделена часть, которую нужно скопировать.

Теперь запишем в файл activity\_main.xml в ImageButton следующую строку android:visibility="invisible". Это сделает данное поле невидимым при старте программы. Запустим программу, чтобы убедиться в этом.

Наконец, допишем скрипт MainActivity.java так, чтобы по нажатию кнопки в текстовом поле изменялся текст, а поле *imageButton* становилось видимым.

Для этого создадим функцию onClick() внутри следующей конструкции:

```
View.OnClickListener oMyButton = new View.OnClickListener() {

@Override // ключевое слово для переопределения метода

public void onClick(View v) {

}

};
```

Метод `OnClickListener` класса `View` запустится и будет работать все время, пока запущена программа.

Теперь, внутри функции `onCreate` допишем следующую строку:

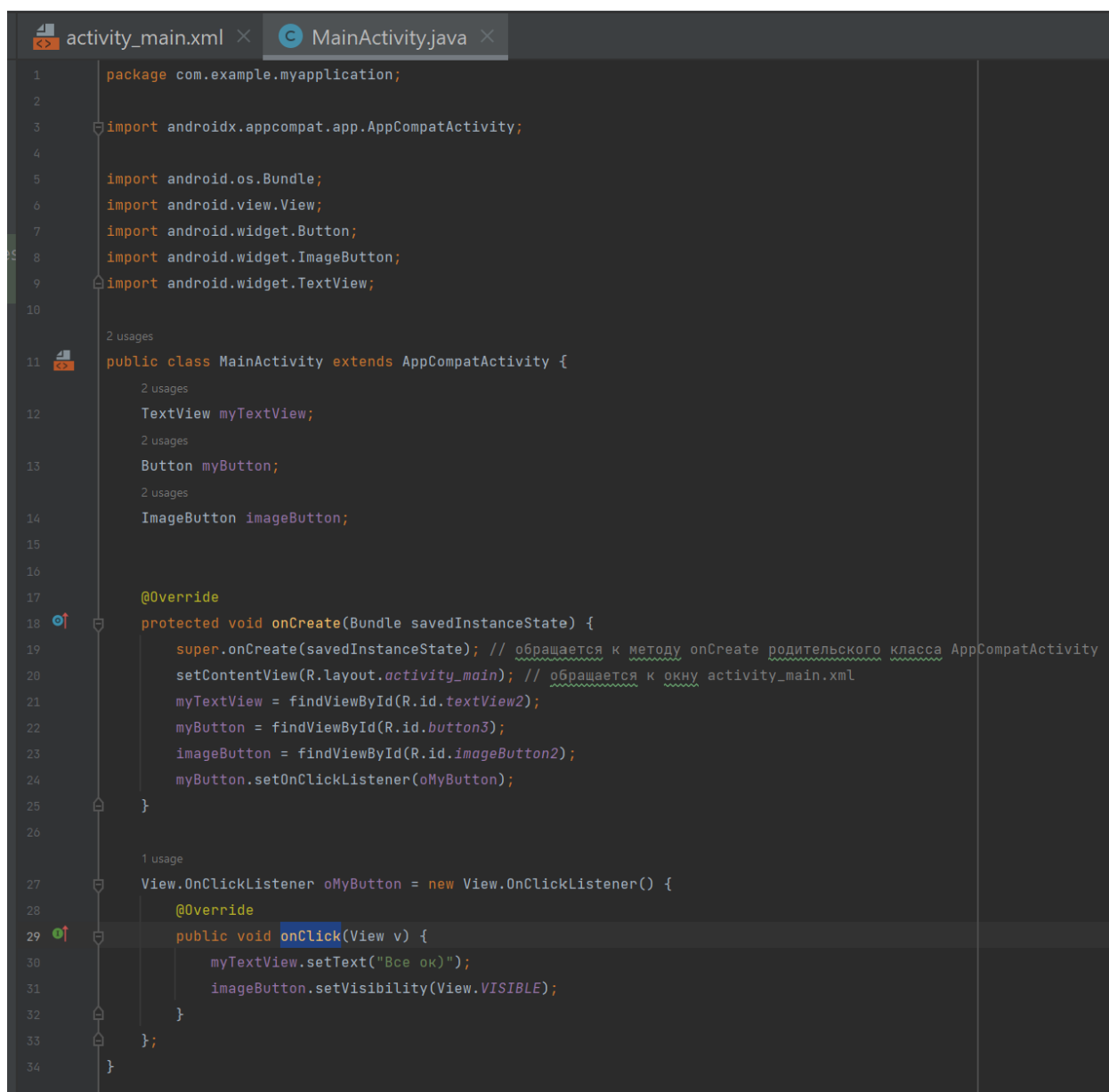
`myButton.setOnClickListener(oMyButton);` // свяжет кнопку `myButton` с объектом `oMyButton` класса `OnClickListener`

Допишем внутри метода `onClick()` следующие строки кода:

`myTextView.setText("Все ок");`

`imageButton.setVisibility(View.VISIBLE);`

В итоге, должно получиться следующее:



```
1 package com.example.myapplication;
2
3 import androidx.appcompat.app.AppCompatActivity;
4
5 import android.os.Bundle;
6 import android.view.View;
7 import android.widget.Button;
8 import android.widget.ImageButton;
9 import android.widget.TextView;
10
11 public class MainActivity extends AppCompatActivity {
12     TextView myTextView;
13     Button myButton;
14     ImageButton imageButton;
15
16     @Override
17     protected void onCreate(Bundle savedInstanceState) {
18         super.onCreate(savedInstanceState); // обращается к методу onCreate родительского класса AppCompatActivity
19         setContentView(R.layout.activity_main); // обращается к окну activity_main.xml
20         myTextView = findViewById(R.id.textView2);
21         myButton = findViewById(R.id.button3);
22         imageButton = findViewById(R.id.imageButton2);
23         myButton.setOnClickListener(oMyButton);
24     }
25
26     1 usage
27     View.OnClickListener oMyButton = new View.OnClickListener() {
28         @Override
29         public void onClick(View v) {
30             myTextView.setText("Все ок");
31             imageButton.setVisibility(View.VISIBLE);
32         }
33     };
34 }
```

Рис 2.8 Итоговый скрипт лабораторной №2

## ЛАБОРАТОРНАЯ РАБОТА № 3

### ОБРАБОТЧИКИ СОБЫТИЙ

Изучение способов обработки нажатия на кнопку и создание приложения для подсчёта количества нажатий на кнопку.

**Цель занятия:** Изучить основные способы обработки активации кнопки. Написать скрипт для подсчета количества нажатий на кнопку.

#### *Методические указания*

Существует несколько способов обработки нажатия кнопки:

- 1) с помощью атрибута `onClick`;
- 2) с помощью метода `setOnClickListener()`;
- 3) с помощью интерфейса `OnClickListener`

Одним из них является использование интерфейса `OnClickListener` и его метода `onClick`.

Программирование приложений в ANDROID studio основано на языке java, в котором есть классы и интерфейсы. Классом называю шаблон, содержащий в себе функции(их еще называют методы или свойства) и переменные(также, их называют поля). Интерфейс же, не имеет конкретной реализации функций, а только их названия. А сами функции(методы, свойства) интерфейса описываются в классах, применяющих эти интерфейсы. Например, интерфейсы есть у большинства компьютерных программ и игр. В широком смысле интерфейс — некий «пульт», который связывает две взаимодействующие друг с другом стороны.

Простой пример интерфейса из повседневной жизни — пульт от телевизора. Он связывает два объекта, человека и телевизор, и выполняет разные задачи: прибавить или убавить звук, переключить каналы, включить или выключить телевизор.

Одной стороне (человеку) нужно обратиться к интерфейсу (нажать на кнопку пульта), чтобы вторая сторона выполнила действие. Например, чтобы телевизор переключил канал на следующий. При этом пользователю не обязательно знать устройство телевизора и то, как внутри него реализован процесс смены канала.

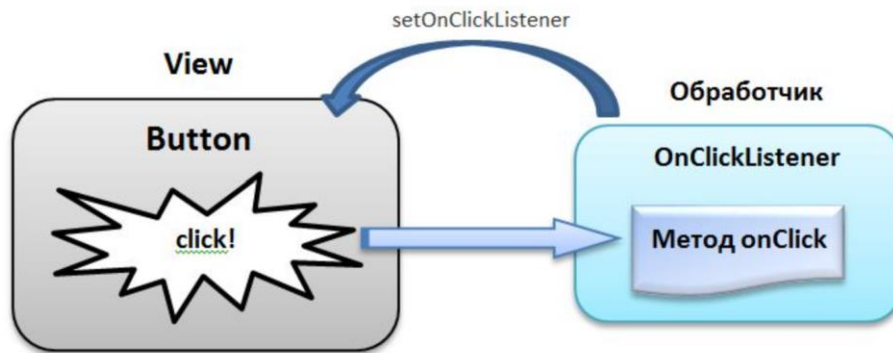


Рис 3.1 Обработчик событий

Наша задача состоит в том, чтобы написать скрипт, увеличивающий число в окне на единицу, каждый раз, когда была нажата кнопка. Итого, нам понадобятся всего два элемента, сама кнопка и текстовое поле, для отображения числового значения.

Создадим новый модуль, добавим кнопку и текстовое поле.

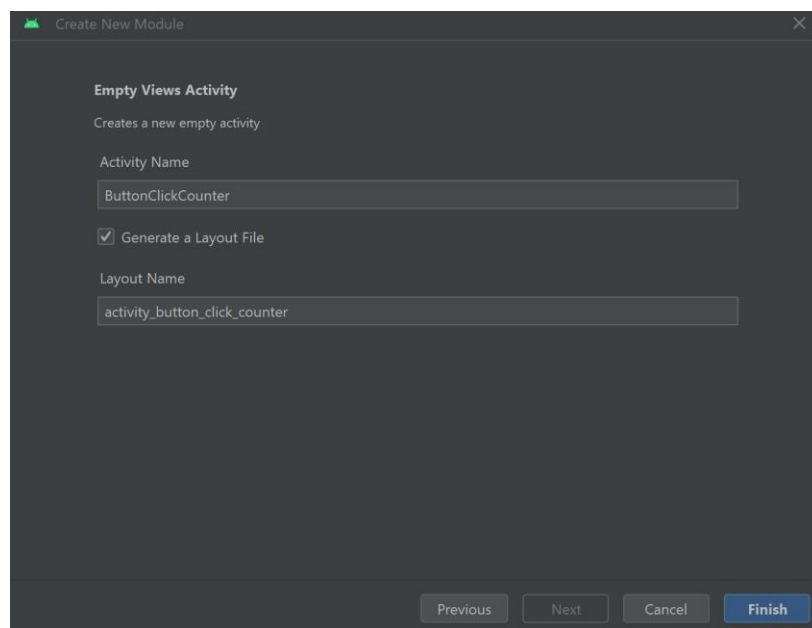


Рис 3.2 Создание нового модуля

По аналогии с предыдущим проектом, объявим две переменные:

```
TextView myTextView;
```

```
Button myButton;
```

Сохраним ссылки на объекты в переменные внутри метода onCreate():

```
myTextView = findViewById(R.id.textview);
```

```
myButton = findViewById(R.id.button);
```



```
myButton.setOnClickListener(oMyButton);
```

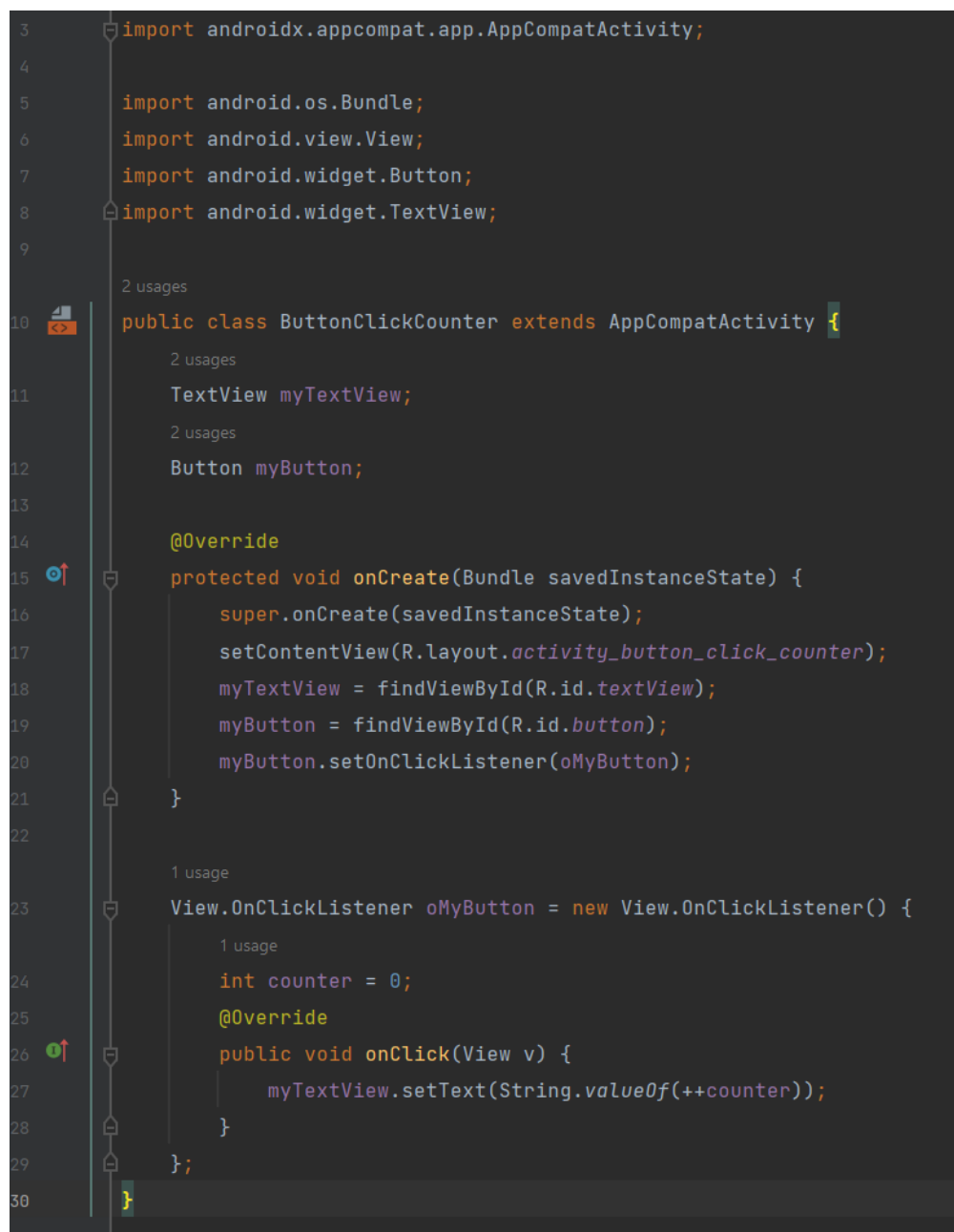
Внутри объекта класса `View.OnClickListener` создадим переменную:

```
int counter = 0;
```

А внутри метода `onClick()` будем увеличивать переменную `counter` на единицу и устанавливать в текстовое поле, предварительно преобразовав `counter` в строку:

```
myTextView.setText(String.valueOf(++counter));
```

В итоге, должен получиться следующий скрипт:



```
3 import androidx.appcompat.app.AppCompatActivity;
4
5 import android.os.Bundle;
6 import android.view.View;
7 import android.widget.Button;
8 import android.widget.TextView;
9
10 public class ButtonClickCounter extends AppCompatActivity {
11     TextView myTextView;
12     Button myButton;
13
14     @Override
15     protected void onCreate(Bundle savedInstanceState) {
16         super.onCreate(savedInstanceState);
17         setContentView(R.layout.activity_button_click_counter);
18         myTextView = findViewById(R.id.textview);
19         myButton = findViewById(R.id.button);
20         myButton.setOnClickListener(oMyButton);
21     }
22
23     View.OnClickListener oMyButton = new View.OnClickListener() {
24         int counter = 0;
25         @Override
26         public void onClick(View v) {
27             myTextView.setText(String.valueOf(++counter));
28         }
29     };
30 }
```

Рис 3.3 Скрипт лабораторной работы №3

## **Лабораторная № 14 Индивидуальное задание**

Можно выбрать любой вариант индивидуального задания из предложенных или придумать свой. Язык программирования, используемые технологии, библиотеки, СУБД для реализации индивидуального задания остаются на усмотрение студента. Прикрепить ссылку на репозиторий с индивидуальным заданием. При разработке ИЗ приветствуется работа в команде и активное использование систем контроля версий.

### **Варианты индивидуальных заданий:**

#### **Задача 1. Приложение "Погода"**

На 1-м экране (отображение) должно быть:

Возможность выбрать город

Возможность выбрать единицы измерения температуры (градус Цельсия, градус Фаренгейта, Кельвин)

Ссылка на видео с примером:

[https://www.youtube.com/watch?v=zzV2aML\\_zNg](https://www.youtube.com/watch?v=zzV2aML_zNg)

Результат выложить на Github.com

#### **Задача 2. Репозитории GitHub**

При помощи GitHub API отобразить список репозиторий у организации xxx (github.com/xxx)

Дизайн - полностью на ваше усмотрение

Код разместить на свой GitHub и прислать преподавателю ссылку

Обязательно:

Код на Java/Kotlin/Swift/Unity

Желательно:

Фильтр поиска по названию, объему, популярности(количеству звезд), дате создания, дате последнего комита и т.д.

### **Задача 3. Пользователи GitHub**

Главный экран:

Users (список всех Github пользователей). Использовать API <https://developer.github.com/v3/users/#get-all-users>

В элементе списка отрисовать avatar, login (title), id (subtitle)

По нажатию на элемент списка реализовать переход на UserDetails

UserDetails (экран с информацией о пользователе):

Использовать API <https://developer.github.com/v3/users/#get-a-single-user>

Поля: Avatar, Name, Email, Organization (если есть), Following count, Followers count, Дата создания аккаунта

Требования:

Язык Java/Kotlin/Swift/Unity

Код поместить в репозиторий на GitHub/Bitbucket/GitLab

Неоднозначности задания трактуются на усмотрение разработчика

### **Задача 4. Приложение "Вечеринка"**

На экране располагаются:

Картинка (загружается по URL)

Название вечеринки

Имя пригласившего

Фото пригласившего (извлекается по URL)

Список гостей с фото, которые идут вместе с текущим пользователем. Фото загружаются по URL

Все данные должны загружаться из файла в формате JSON, зашифрованного в ресурсах приложения.

Критерии проверки: корректность работы: отсутствие крашей и багов, соответствие требованиям, код как для продакшена, читабельный и аккуратный, с разделением обязанностей между классами, код должен легко модифицироваться.

### **Задача 5. Приложение из 1-го экрана**

Необходимо создать приложение состоящее из 1 экрана

Реализовать данное задание с применением RxJava/RxSwift, любого DI (например Dagger 2) и MVP.

Реализовать:

База данных 5 полей и заполнить любыми данными

Вывод в активити или фрагмент в виде списка

Live search по базе данных

Реализовать сортировку по любому параметру

### **Задача 6. Приложение "Авто"**

Приложение, должно иметь следующие функции:

Отображение списка автомобилей с характеристиками (10-12 автомобилей, 3 производителя, 1-3 марки у каждого производителя)

Добавление нового автомобиля

Редактирование деталей автомобиля

Желательно:

Фильтрация по производителю и марке

Сортировка по цене

## **Задача 7. Проверка адреса email**

Создать хороший UX для пользователей, вводящих адрес электронной почты и пароль при регистрации в приложении.

Требования:

Проверка формата электронной почты. Пример: user@gmail не является действительным адресом электронной почты

Пользовательский интерфейс должен показывать, действителен или нет адрес электронной почты. При необходимости интерфейс должен указать, что не так с адресом

Автозаполнение и проверка доступности домена. Пользователи часто опечатываются при вводе адреса. Например, указывают неправильно доменное имя (gmail.con вместо gmail.com)

Проверка пароля. Нет ограничения на вводимые символы. Есть ограничение минимальной и максимальной длины

При необходимости, интерфейс должен указать, что неправильно

Проверить, что заполнены все поля, и указать, какое именно не заполнено

Для автозаполнения необходимо:

Проверить существование введённого домена

Указать, что неправильно в введённом имени

## **Задача 8 Планировщик задач**

Создать менеджер задач, в котором можно создавать активности и задавать время для их выполнения. Выполненные активности маркируются определенным цветом в зависимости от того, выполнены ли они или нет.

#### **Задача 9 Мини викторина(mini quiz)**

Создать приложение-квиз, в котором можно задавать список вопросов и ответы на них. По окончании викторины организовать подсчет баллов, набранных за игру.

#### **Задача 10 Тренажер азбуки морзе**

В поле на стартовой странице вводится строка или предложение. Задача программы преобразовать введенную строку в последовательность звуков азбуки морзе и проиграть ее с помощью динамиков мобильного устройства.

#### **Задача 11 Игра на Unity 2D/3D**

Написать игру с любым сюжетом на любом языке программирования под мобильное устройство IOS или Android.

#### **Задача 12 Свой вариант приложения для мобильного устройства**

Согласовать вариант с лектором по дисциплине “разработка мобильных приложений”

## БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Хашими С., Коматинени С., Маклин Д. Разработка приложений для Android. – Санкт-Петербург: Питер, 2011.-736 с.
2. Харди Б., Филлипс Б., Стюарт К., Марсикано К. Android. Программирование для профессионалов. 2-е изд. - Санкт-Петербург: Питер, 2016. - 640 с
3. Меднике З., Дорнин Л., Мик Б., Накамура. –Москва Программирование под Android. 2-е изд. - СПб.: Питер, 2013. - 560 с.
4. Дейтел ГГ, Дейтел Х., Уолд А. Android для разработчиков. 3-е изд. - Санкт-Петербург: Питер, 2016. - 512 с.
5. Start Android - учебник по Android для начинающих и продвинутых [Электронный ресурс] Режим доступа: <https://startandroid.ru/ru/>
6. Введение в разработку приложений для ОС Android [Электронный ресурс] - Национальный Открытый Университет «ИНТУИТ» 2014г. Режим доступа: <https://www.intuit.ru/studies/courses/12643/1191/info>
7. Разработка приложений для смартфонов на ОС Android [Электронный ресурс] - Национальный Открытый Университет «ИНТУИТ». Режим доступа: <https://www.intuit.ru/studies/courses/12786/1219/info>
8. Сайт Александра Климова. Изучаем Android.[Электронный ресурс] Режим доступа: <http://developer.alexanderklimov.ru/android/>