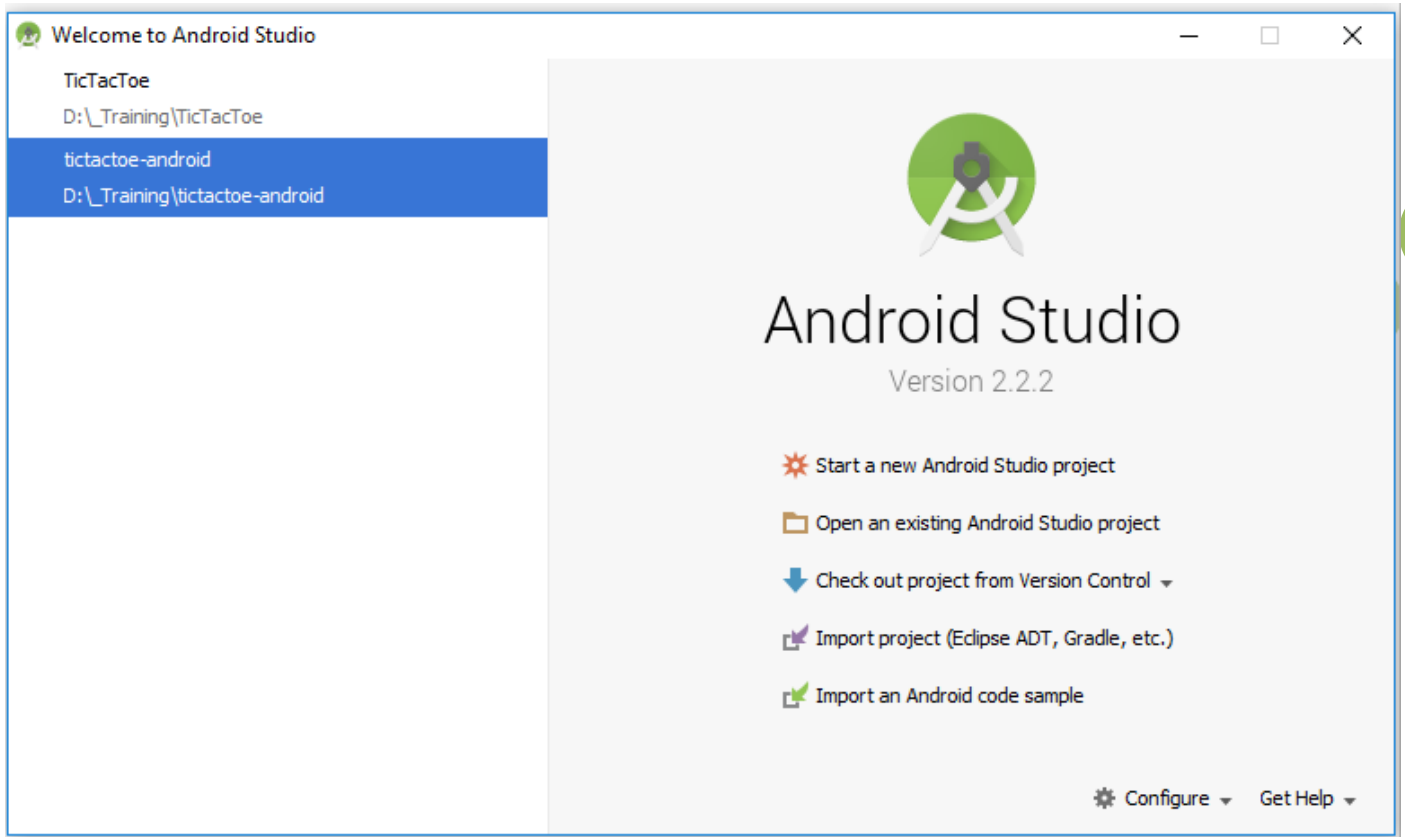


**Пример 2 FirstAndroidApp.** Итоговый пример программного обращения к компонентам

**Последовательность создания:**

Запускаем Андроид Студию и начинаем цикл создания нового приложения – нажимаем - «Start a new Android Studio project»:

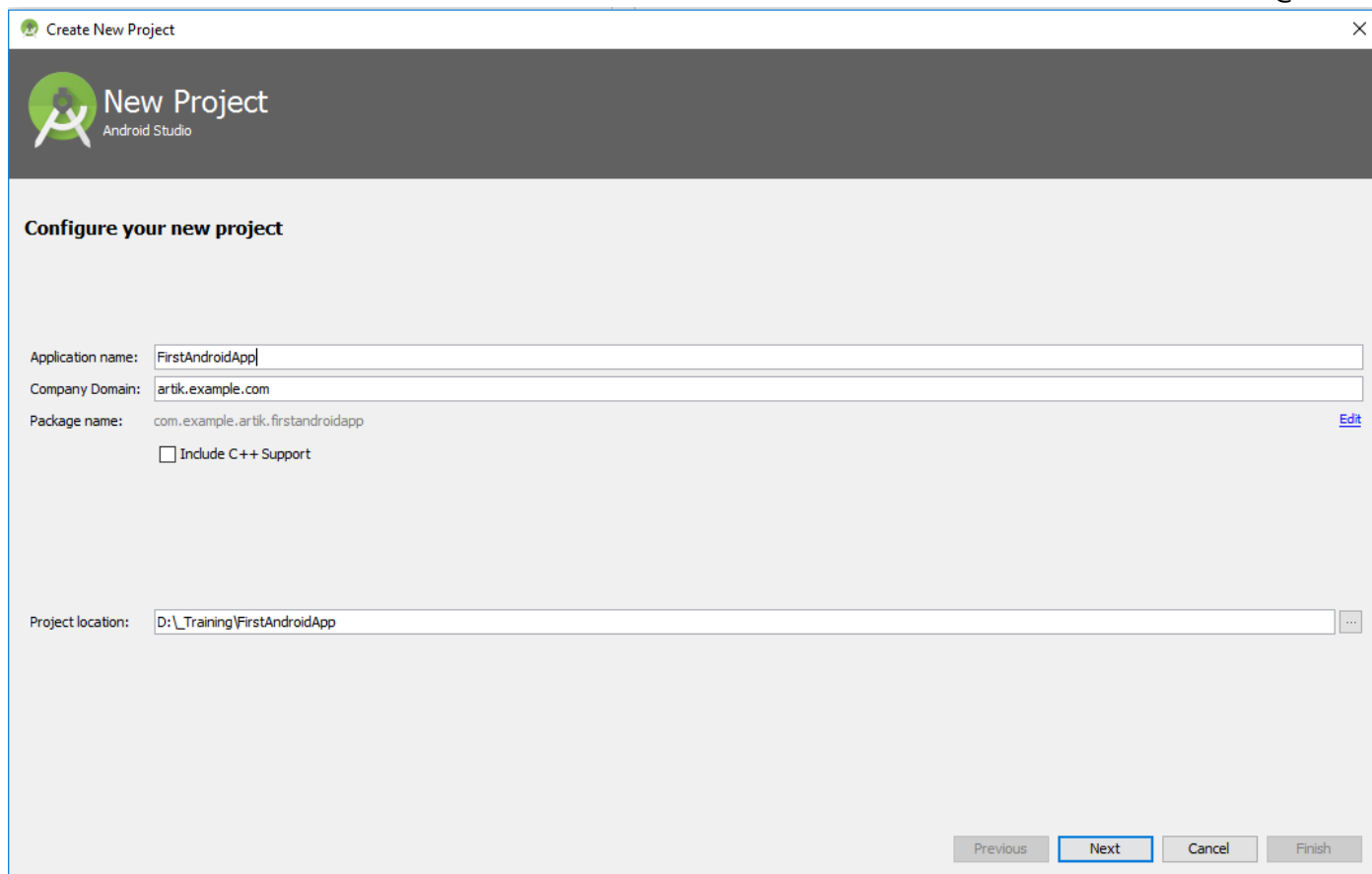


Далее задаем основные настройки проекта:

**Application name** – имя приложения

**Company name** – пакет размещения

**Project location** – каталог, в котором хранится проект – у Студи должны быть полные права доступа к этому каталогу



Create New Project

## New Project

Android Studio

### Configure your new project

Application name: FirstAndroidApp

Company Domain: artik.example.com

Package name: com.example.artik.firstandroidapp [Edit](#)

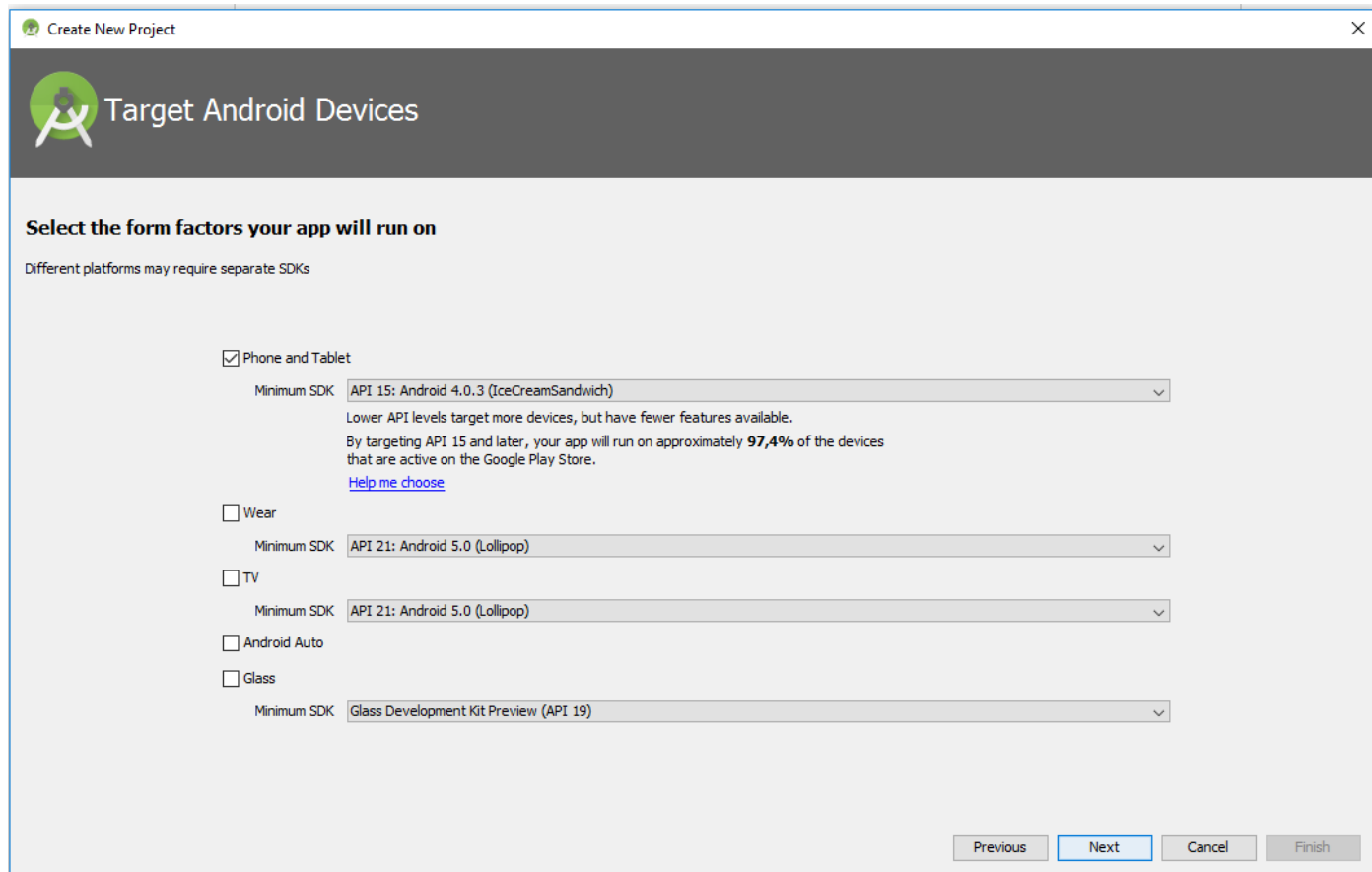
☐ Include C++ Support

Project location: D:\\_Training\FirstAndroidApp

Previous Next Cancel Finish

2

### Задание минимальной платформы



Create New Project

## Target Android Devices

### Select the form factors your app will run on

Different platforms may require separate SDKs

☒ Phone and Tablet

Minimum SDK: API 15: Android 4.0.3 (IceCreamSandwich)

Lower API levels target more devices, but have fewer features available.

By targeting API 15 and later, your app will run on approximately **97,4%** of the devices that are active on the Google Play Store.

[Help me choose](#)

☐ Wear

Minimum SDK: API 21: Android 5.0 (Lollipop)

☐ TV

Minimum SDK: API 21: Android 5.0 (Lollipop)

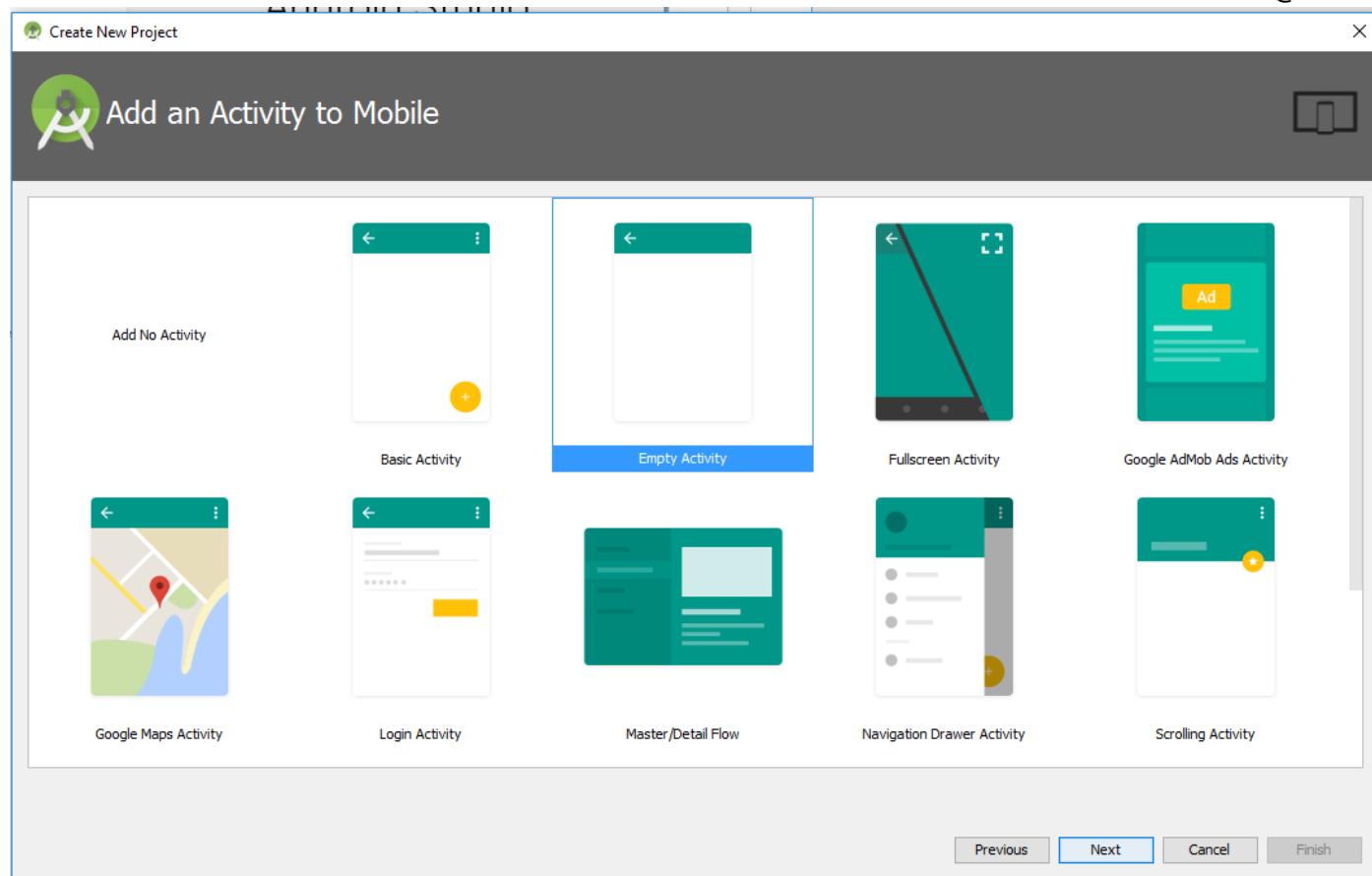
☐ Android Auto

☐ Glass

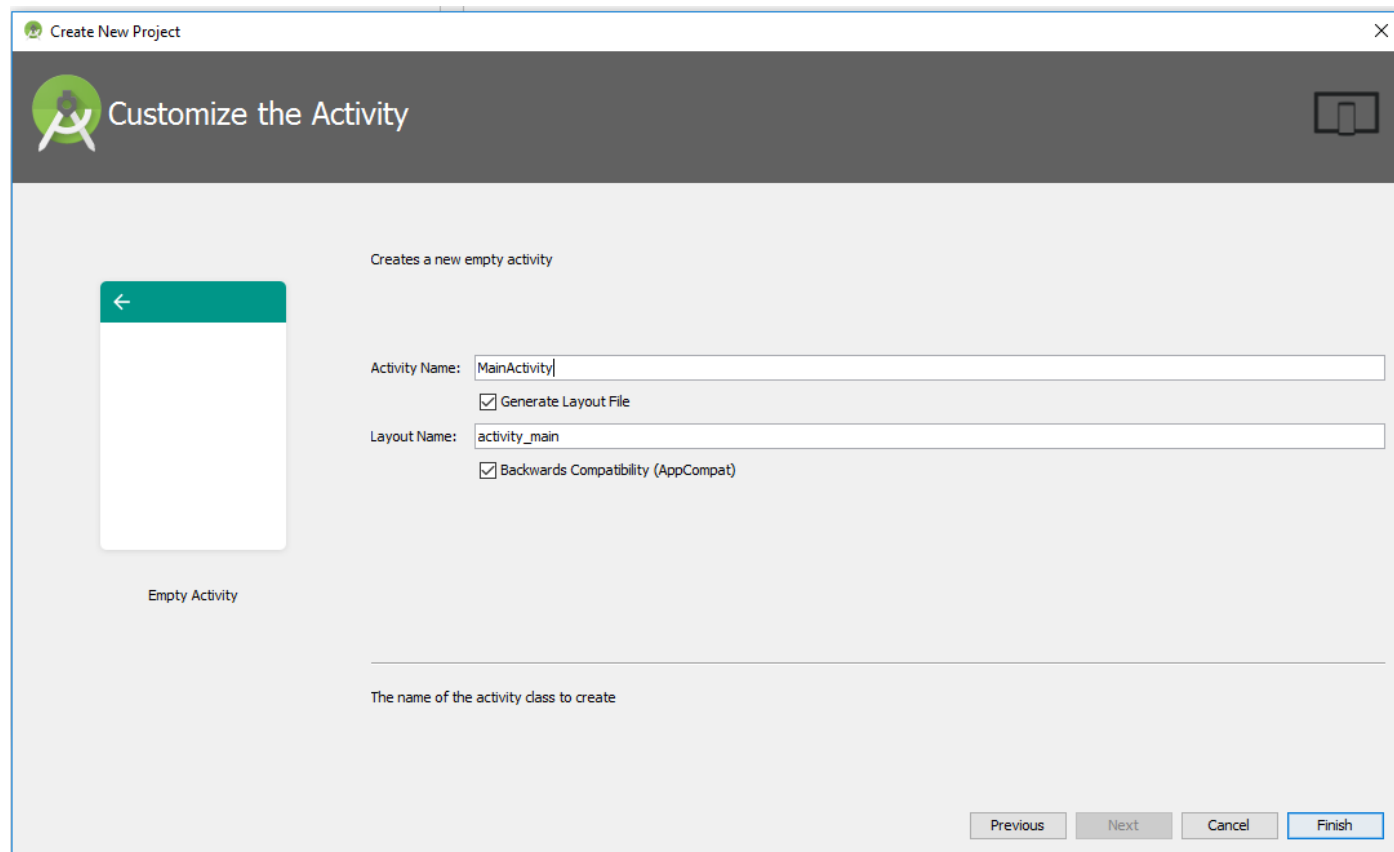
Minimum SDK: Glass Development Kit Preview (API 19)

Previous Next Cancel Finish

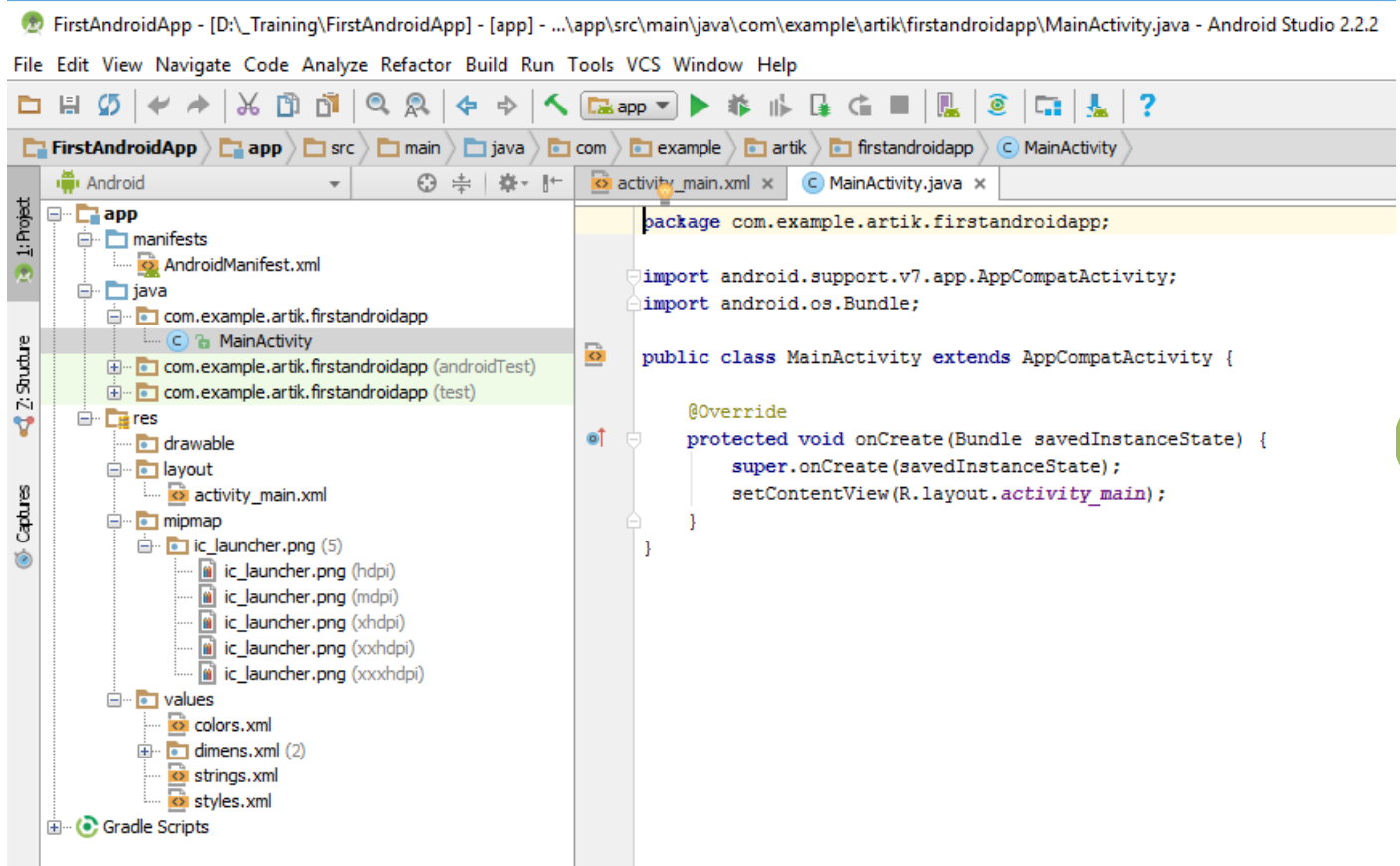
Далее выбираем или Blank Activity или - Empty Activity – «чистую» заготовку для приложения:



Последняя стадия:



В итоге получаем структуру проекта:



**AndroidManifest.xml** – структура проекта-приложения

**Activity\_main.xml** – разметка главной активности

**MainActivity.java** – программный код главной активности

Разметка layout-файл **activity\_main.xml** напишем следующее и сохраним:

Было:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.artik.firstandroidapp.MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!" />
</RelativeLayout>
```

Стало:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">
    <LinearLayout
        android:id="@+id/linearLayout1"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_margin="30dp"
        android:orientation="vertical">
        <TextView
            android:id="@+id/tvOut"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:layout_marginBottom="50dp"
            android:text="TextView">
        </TextView>
        <Button
            android:id="@+id/btnOk"
            android:layout_width="100dp"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:text="OK">
        </Button>
        <Button
            android:id="@+id/btnCancel"
            android:layout_width="100dp"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:text="Cancel">
        </Button>
    </LinearLayout>
</LinearLayout>
```

**MainActivity.java****Было:**

```
package com.example.artik.firstandroidapp;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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

В итоге должен получиться такой код:

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    TextView tvOut;
    Button btnOk;
    Button btnCancel;

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

        // найдем View-элементы
        tvOut = (TextView) findViewById(R.id.tvOut);
        btnOk = (Button) findViewById(R.id.btnOk);
        btnCancel = (Button) findViewById(R.id.btnCancel);

        // создаем обработчик нажатия
        android.view.View.OnClickListener oclBtnOk = new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Меняем текст в TextView (tvOut)
                tvOut.setText("Нажата кнопка ОК");
            }
        };
        // присвоим обработчик кнопке ОК (btnOk)
        btnOk.setOnClickListener(oclBtnOk);
    }
}
```

Использование анонимных классов усложняет чтение программного кода в случае. Если в обработке нужно выполнить достаточно много действий:

```
public class MainActivity extends AppCompatActivity {

    TextView tvOut;
    Button btnOk;
    Button btnCancel;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // найдем View-элементы
        tvOut = (TextView) findViewById(R.id.tvOut);
        btnOk = (Button) findViewById(R.id.btnOk);
        btnCancel = (Button) findViewById(R.id.btnCancel);
        // создаем обработчик нажатия
        android.view.View.OnClickListener oclBtnOk = new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Меняем текст в TextView (tvOut)
                // по id определяем кнопку, вызвавшую этот обработчик
                switch (v.getId()) {
                    case R.id.btnOk:
                        // кнопка ОК
                        tvOut.setText("Нажата кнопка ОК");
                        break;
                    case R.id.btnCancel:
                        // кнопка Cancel
                        tvOut.setText("Нажата кнопка Cancel");
                        break;
                }
            }
        };
        // присвоим обработчик кнопке ОК (btnOk)
        btnOk.setOnClickListener(oclBtnOk);
        btnCancel.setOnClickListener(oclBtnOk);
    }
}
```

**Другие способы задания обработчиков:****1. Передача обработки самой Activity**

Создадим проект FirstAndroidApp2, который структурно должен повторять все созданное в проекте FirstAndroidApp, но в нем для активности реализуем интерфейс OnClickListener

```
public class MainActivity extends AppCompatActivity implements
    android.view.View.OnClickListener{
```

```
    TextView tvOut;
    Button btnOk;
    Button btnCancel;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // найдем View-элементы
        tvOut = (TextView) findViewById(R.id.tvOut);
        btnOk = (Button) findViewById(R.id.btnOk);
        btnCancel = (Button) findViewById(R.id.btnCancel);
        // присвоим обработчик кнопке OK (btnOk)
        btnOk.setOnClickListener(this);
        btnCancel.setOnClickListener(this);
    }
    @Override
    public void onClick(View v) {
        // по id определяем кнопку, вызвавшую этот обработчик
        switch (v.getId()) {
            case R.id.btnOk:
                // кнопка OK
                tvOut.setText("Нажата кнопка OK");
                break;
            case R.id.btnCancel:
                // кнопка Cancel
                tvOut.setText("Нажата кнопка Cancel");
                break;
        }
    }
}
```

**2. Использование разметки и внутренних методов:**

Есть еще один способ реализации. В layout-файле (main.xml) при описании кнопки пишем:

```
1  <Button
2      android:id="@+id/btnStart"
3      android:layout_width="wrap_content"
4      android:layout_height="wrap_content"
5      android:onClick="onClickStart"
6      android:text="start">
7  </Button>
```

Т.е. используем атрибут **onClick**. В нем указываем имя метода из Activity. Этот метод и сработает при нажатии на кнопку.

Далее, добавляем этот метод в Activity (MainActivity.java). Требования к методу: public, void и на вход принимает View:

```
1  public void onClickStart(View v) {
2      // действия при нажатии на кнопку
3  }
```

В методе прописываете необходимые вам действия, и они будут выполнены при нажатии кнопки



