

2.1. Первое Android приложение

Сайт: [Samsung Innovation Campus](https://innovationcampus.ru)
Курс: Мобильная разработка на Kotlin
Книга: 2.1. Первое Android приложение

Напечатано:: Murad Rezvan
Дата: понедельник, 3 июня 2024, 17:42

Оглавление

2.1.1 Операционная система Android

- Компоненты Android

2.1.2 Создание проекта в Android Studio

- Упражнение 2.1.1. Создание первого приложения.

2.1.3 Состав Android-проекта

- Манифест приложения
- Ресурсы и код
- Сборщик проекта
- Упражнение 2.1.2 Работа в окне обозревателя проекта.

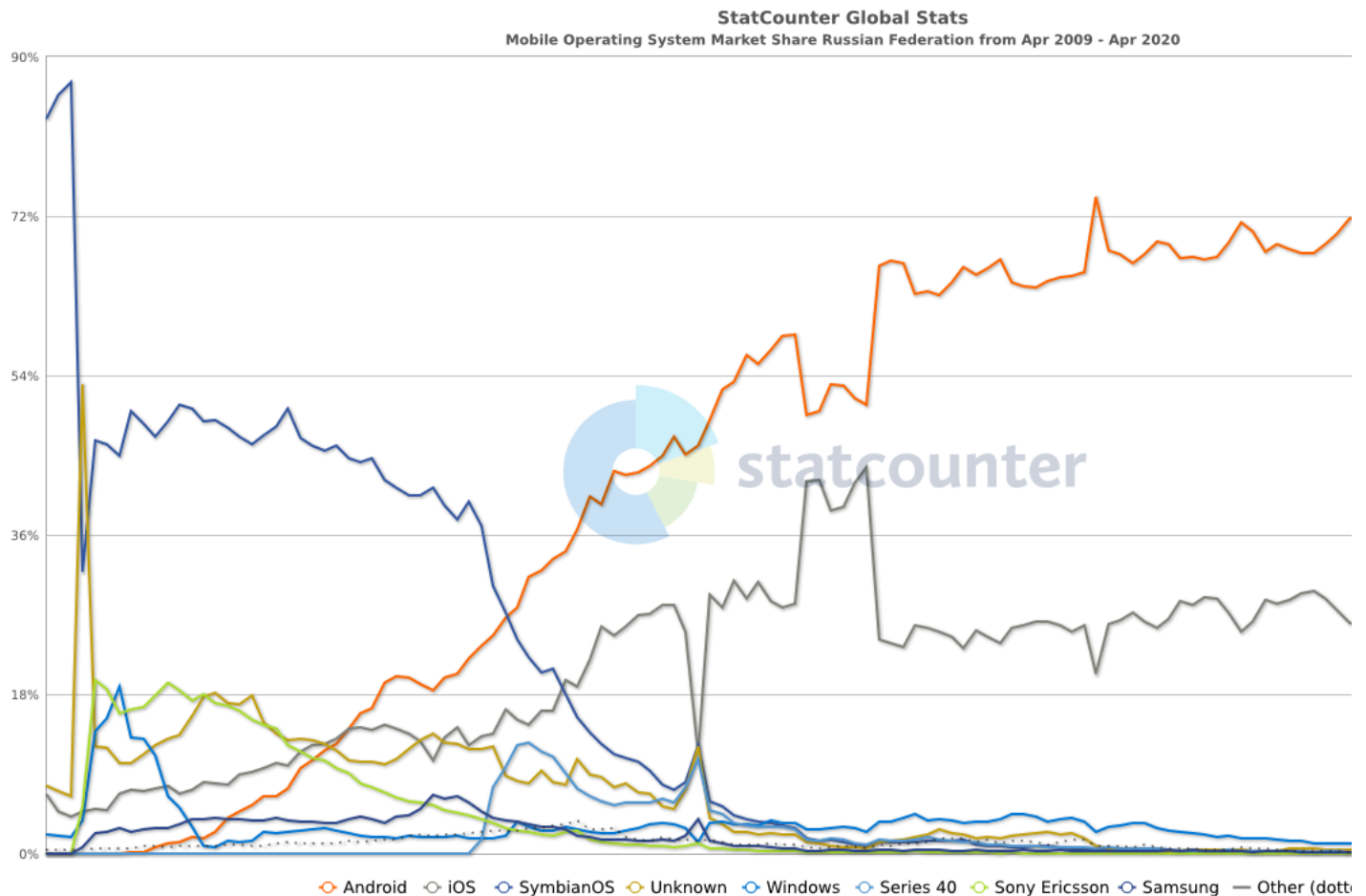
2.1.4 Запуск приложения

- Упражнение 2.1.3 Запуск приложения на устройстве и редактирование настроек.

Домашнее задание

2.1.1 Операционная система Android

Android – свободно распространяемая операционная система для мобильных «умных» устройств, основанная на ядре Linux 2.6. Первая устойчивая версия Android 1.0 для смартфонов была выпущена 23 сентября 2008 года. Android – самая распространённая в мире (в том числе в Российской Федерации) мобильная операционная система, под управлением которой работают не только смартфоны и планшеты, но так же «умные» телевизоры, часы, очки и автомобильная система навигации.



Android SDK

SDK (software development kit) – набор инструментов для разработки приложений, встраиваемый по необходимости в интегрированную среду программирования. В состав Android SDK входят:

- API (application programming interface) – интерфейсы прикладного программирования для различных версий платформы;
- SDK Manager – программа-загрузчик API;
- Debug Monitor – инструмент отладки приложения и отображения происходящих процессов;
- AVD (android virtual device) Manager + Android Emulator – программа для создания виртуального устройства запуска приложений и сам эмулятор.

Информационная поддержка системы и её программные элементы размещаются на [официальном сайте Android](#)

Используемое программное обеспечение

Официальной средой разработки приложений для Android объявлена свободно распространяемая IDE [Android Studio от Google Corp.](#) Данная среда создана на основе продукта [IntelliJ Idea от JetBrains](#), который также содержит плагин ADT (Android Developer Tools), что позволяет и в данной среде вести разработку приложений для операционной системы Android. В учебнике в качестве рабочего инструментария используется Android Studio. При первом запуске, в том числе после установки, среда производит проверку наличия всех необходимых компонентов и при необходимости SDK Manager производит загрузку недостающих инструментов с репозитория разработчика. По этому важно при запуске IDE подключение компьютера к сети Интернет.

Компоненты Android

Архитектура системы включает в себя четыре уровня:

- **уровень ядра.** На этом уровне контролируется аппаратное обеспечение устройства, в том числе работают драйверы межпроцессорного взаимодействия (IPC) и управления питанием. Хотя система и построена на ядре Linux, однако, она имеет некоторые специфические расширения ядра, свойственные Android, а значит, не является Linux-системой;
- **уровень библиотек.** Кроме стандартных SLD (2D графика), OpenGL (3D графика), Media Framework (мультимедиа), LibWebCore(встроенный браузер), FreeType (поддержка шрифтов), SQLite (работа с базой данных), SSL (зашифрованные соединения), разработчики Android создали собственную версию стандартной библиотеки C/C++ - библиотеку Bionic, не поддерживающая исключения C++ и несовместимая с GNU libs и POSIX. На этом же уровне расположен *Менеджер поверхностей (окон)*, позволяющий создавать различные эффекты изображений за счёт хранения рисунков окон в битовых массивах, не отправляя их непосредственно в буфер экрана. Такой подход позволяет создавать различные экранные эффекты, как прозрачность и плавная анимация смены экранов. Ещё одна составляющая данного уровня - это среда запуска приложений *Android Runtime*, использующая виртуальную машину Java *Dalvik Virtual Machine*. Каждый процесс в Android выполняется на отдельной виртуальной машине в отдельном потоке, что позволяет отделить работу приложений друг от друга, тем самым обеспечивая безопасность системы «изоляцией» вредоносных программ. Таким образом, Android не разделяет системные и пользовательские приложения, для этой системы все процессы одинаковы, а «важность» приложения регулируется установкой приоритета потока. Но у такой «защиты» имеется и обратная сторона: каждое приложение «видит» только себя и не имеет возможности использовать ресурсы другого приложения. В этом случае на помощь приходят менеджеры внешних ресурсов (контент-провайдеры), располагающиеся на следующем уровне операционной системы;
- **уровень каркаса приложений.** Включает различные службы, курирующие работу составляющих системы.

Activity Manager — диспетчер активности, который отвечает за функционирование приложения и его жизненный цикл.

Resource Manager — диспетчер ресурсов необходим для доступа к используемым внутренним ресурсам (строковым, графическим и т.п.).

Package Manager — диспетчер пакетов, отвечающий за установку и функционирование пакетов прикладных программ.

Window Manager — диспетчер окон, распределяющий активность окон приложений и порядок их отображения.

Telephony Manager — менеджер телефонии следит за типом доступа и параметрами сети.

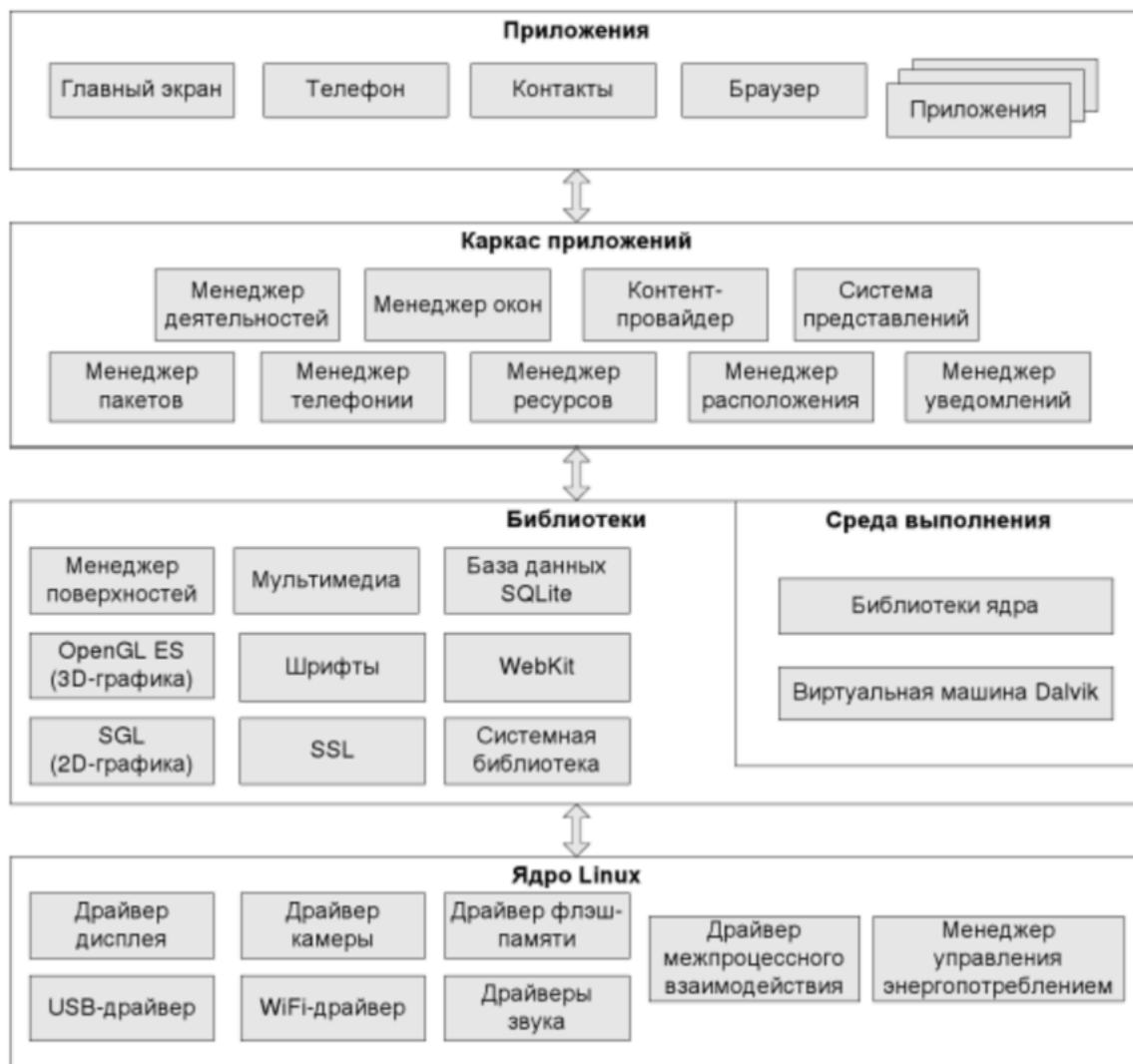
Location Manager — менеджер местоположения — навигационные службы, передающие приложениям информацию о местоположении устройства.

Notification Manager — диспетчер уведомлений позволяет приложению публиковать сообщения в строке состояния.

Content Providers — менеджер внешних ресурсов, открывающий доступ к другим приложениям.

View System — система представлений, используемая для создания внешнего оформления приложения. Имеет расширяемую функциональность.

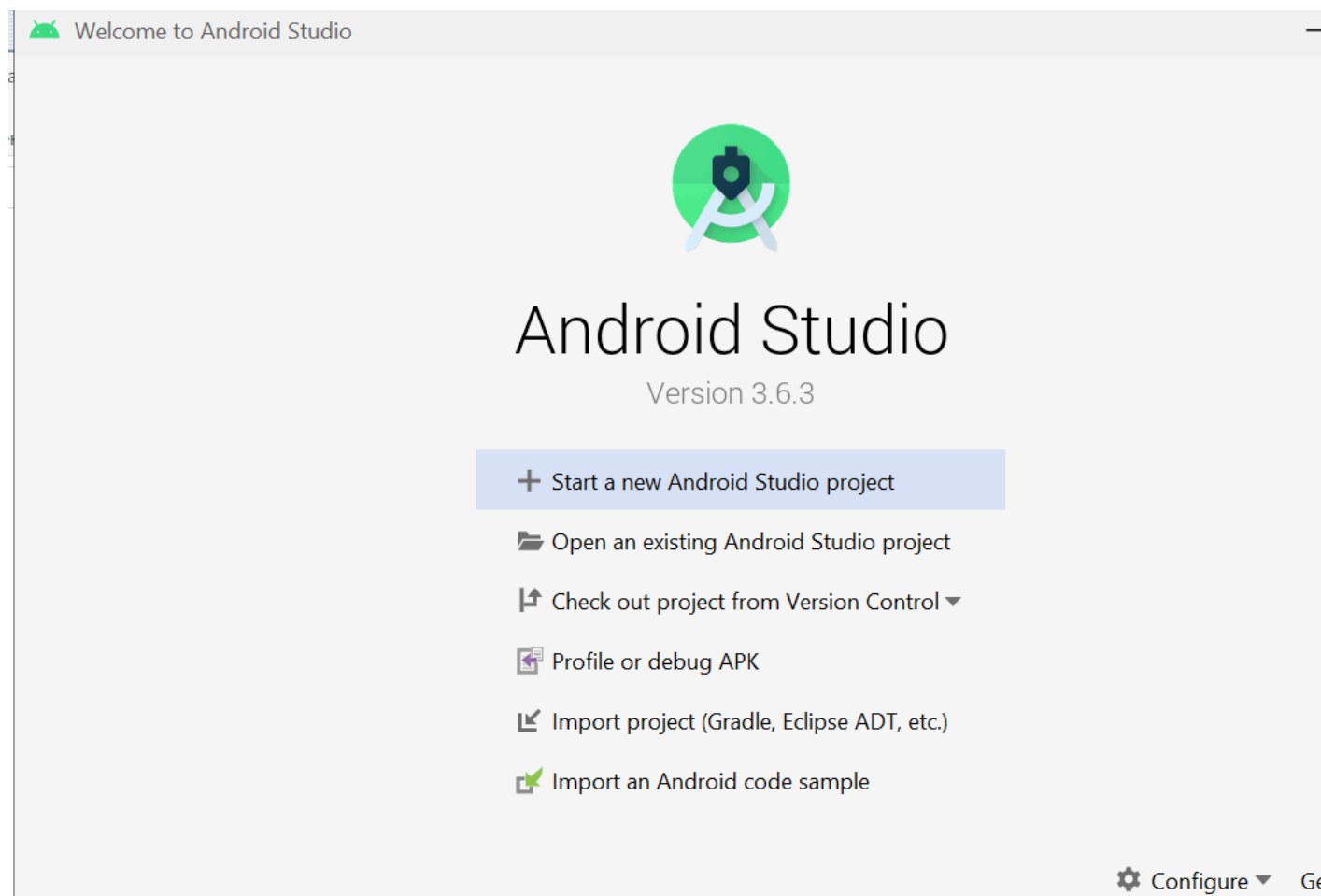
- **уровень приложений.** Платформа Android не различает по правам базово установленные и сторонние приложения, что позволяет менять программную конфигурацию устройства, в том числе на самостоятельно разработанные.



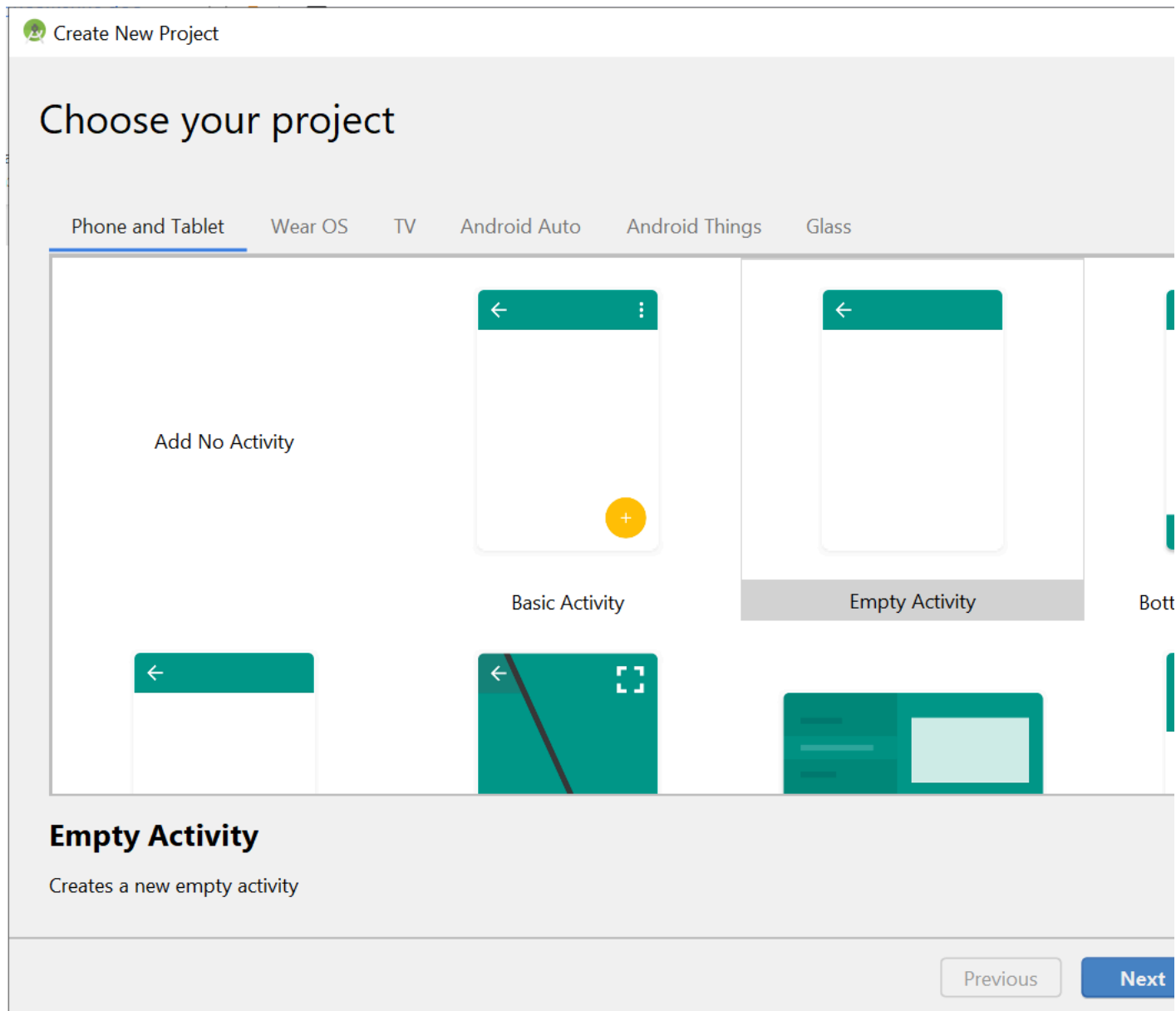
Компания Google на конференции разработчиков I/O - 2019 официально заявила, что основным языком разработки под Android является Kotlin, поддержка которого была добавлена в 2017 году. Таким образом, вся программная и информационная поддержка системы теперь в первую очередь ориентируется на Kotlin. Однако, Android-разработчики до сих пор работают на Java, долгое время являющегося основным инструментом разработки под Android. И, конечно же, есть программисты, разрабатывающие Android-приложения на C++.

2.1.2 Создание проекта в Android Studio


При первом запуске среды разработки появляется экран приветствия с предложениями создать или открыть проект. Если при закрытии Android Studio оставить рабочий проект открытым, то при следующем запуске в рабочую область будет загружен открытый проект. Именно по этому рекомендуется при завершении работы закрывать проект командой File -> Close Project. Для создания нового проекта на стартовом экране необходимо выбрать раздел *Start a new Android Studio project*. То же самое можно сделать в уже запущенной среде Adroid Studio, используя строку меню File ->New ->NewProject...



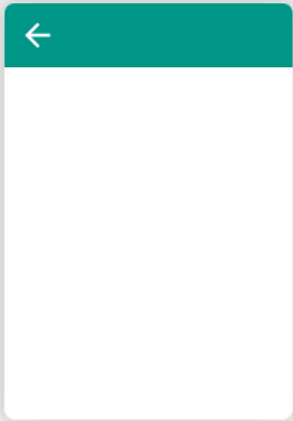
При этом происходит запуск *Мастера создания проектов* в Android Studio, при работе с которым можно просто доверить создание проекта «специалисту» и соглашаться на все установки по умолчанию, либо заполнить параметры проекта самостоятельно. На первом шаге мастер предлагает выбрать устройство, для которого предназначается будущее приложение (на выбор предлагаются устройства под управлением Android: телефон и планшет, ТВ, автосистема, часы, очки) и структуру его экранных представлений. В качестве примера рассмотрено создание простого приложения на смартфон или планшет. Для этого на вкладке Phone&Tablet выбирается самое простое представление экрана: Empty Activity (пустая активность) и совершается переход к следующему шагу мастера кнопкой Next.



На втором шаге предоставляется право дать имя своему проекту, а так же пакету (папке), в которой он будет размещён; выбрать локальное место хранения папки с проектом на компьютере; язык разработки и минимальную конфигурацию устройства, на котором планируется использовать приложение. С того момента, как Google объявила Kotlin основным языком Android-разработки, языком по умолчанию в Мастере установлен именно он.



Configure Your Project



Empty Activity

Name

FirstProject

Package name

ru.samsung.itacademy.mdev.firstproject

Save location

E:\AS\FirstProject

Language

Kotlin

Kotlin

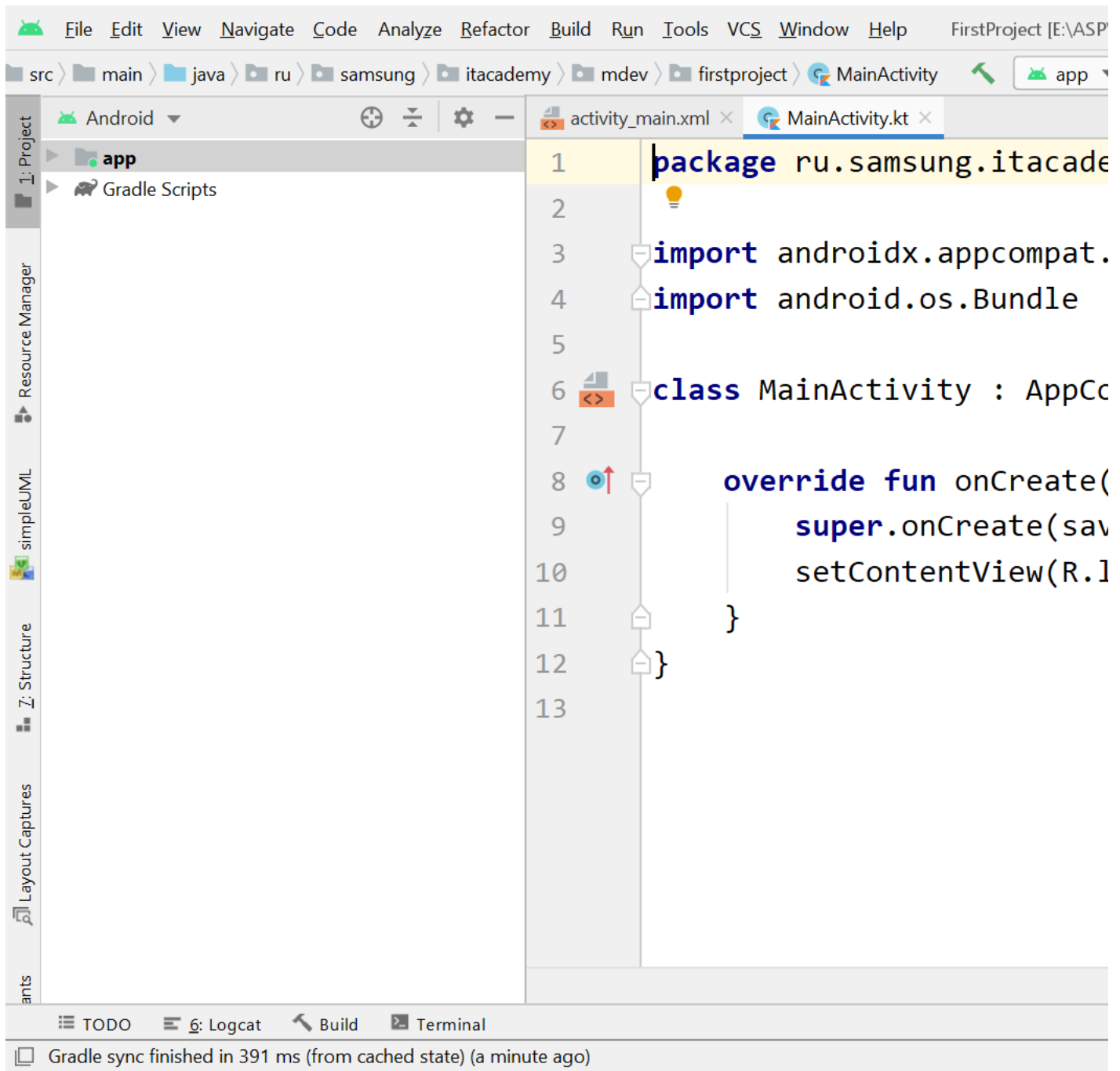
Java

The programming language used for code

i Your app will run on approximately **98,1%** of devices.
[Help me choose](#)

Previous Next

На этом работа мастера завершается и происходит настройка и сборка нового Android-проекта.



Как и любая современная среда разработки, Android Studio имеет многооконный интерфейс, который можно собрать по своему усмотрению. Для удобства и наглядности все окна пронумерованы и свёрнуты в ярлыки по периметру рабочего окна. Рабочая область предназначена для отображения содержимого файлов проекта и всегда присутствует на экране, остальные окна разворачиваются по мере необходимости. Для отображения/скрытия нужного вида достаточно клика по ярлыку соответственного окна. Для работы чаще всего используются два представления: окно проекта (Project) и комбинированное окно сборки, меток, терминала и отладки (Build, TODO, Terminal, LogCat). Именно такое представление элементов считается стандартным и используется в среде разработки по умолчанию.

Упражнение 2.1.1. Создание первого приложения.

Запустите Android Studio и создайте новый проект на шаблоне пустой активности EmptyActivity. Дайте проекту имя My_app, качестве языка разработки выберите Kotlin. Дождитесь сборки проекта. Из закладок по периметру окна Android Studio активируйте и деактивируйте следующие окна:

обозревателя проекта (1: Project),

структуры проекта (7: Structure),

сборщика Gradle,

переменных сборки (Build Variants),

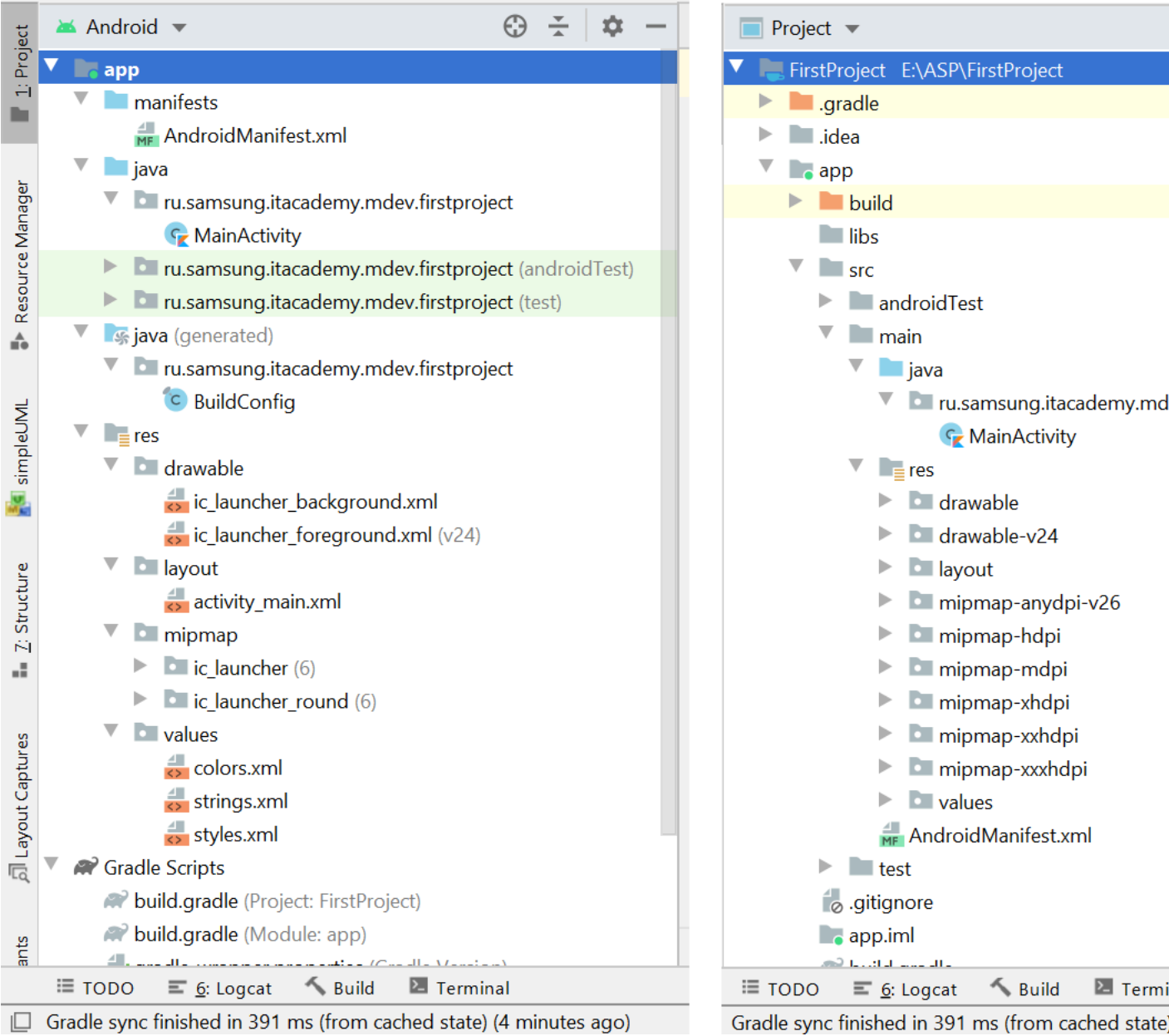
обозревателя файлов устройства (Device File Explorer),

обозревателя ресурсов (Resource Manager).

В окне консоли переключитесь по закладкам Terminal, LogCat, Build.

2.1.3 Состав Android-проекта

Отображение архитектуры проекта происходит в окне обозревателя проекта *Project*. Как и любое приложение, написанное на Kotlin, Android - проект размещается в пакете. Внутри проекта приложение представляется модулем - отдельным элементом. Таким образом, внутри одного проекта можно разместить несколько модулей - уникальных приложений. Обозреватель позволяет выбрать вид представления проекта в зависимости от пожелания разработчика. По умолчанию используется представление *Android*, в котором составляющие сгруппированы по типам файлов и их назначению. При необходимости можно перевести в режим *Project*, отображающий файлы проекта в том виде, как они расположены в папке хранения в долговременной памяти компьютера.



Содержимое проекта условно разделимо на два вида: файлы, отвечающие за состав и наполнение приложения и файлы сборки.

Манифест приложения

Структура проекта описывается в текстовом файле *manifest.xml*. Здесь задаются свойства приложения, регистрируются все его экраны (окна, активности), разрешения на подключения.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="ru.samsung.itacademy.mdev.firstproject">
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
    <uses-sdk android:minSdkVersion="23" />
    <uses-permission android:name="android.permission.CAMERA" />
    <uses-permission android:name="android.permission.INTERNET" />
</manifest>
```

Корневым элементом является `<manifest>`, в свойствах которого указываются пространства имен, в том числе и пакет текущего приложения. На первом уровне вложенности находится описание состава приложения, являющегося объектом `<application>` и используемые настройки. В данном примере используются предустановка минимальной версии операционной системы Android, на которой планируется работа приложения (версия API 23), разрешения `<uses-permission>` на доступ приложения к камере устройства и на выход в интернет. Объект *application* является составным, внутри него располагаются возможные экраны (вложенный объект `<activity>`), из которых состоит приложение. Ровно один из экранов является стартовым: с него запускается приложение. Эта активность в объекте *application* располагается самой верхней и содержит категорию «запускаемая (`android.intent.category.LAUNCHER`)». В приведённом примере это единственная активность с именем MainActivity.

Ресурсы и код

Kotlin, аналогично Java, отделяет ресурсы от кода. По этой причине описание самого проекта содержится в двух папках: файлы кода расположены в папке *src*, файлы ресурсов - в папке *res*.

Папка кодов (*src*) содержит файлы классов, написанные на языке Kotlin. Для удобства можно файлы классов группировать по назначению и помещать в подпапки. Папка ресурсов уже содержит подпапки:

- файлы xml-макетов экранов (папка *layout*);
- файлы xml-описаний изображений (папка *drawable*);
- графические файлы (папка *miptar*);
- строковые и целочисленные константы, а так же описания стилей (папка *values*).

Так же к ресурсам относятся

- файлы xml-описаний разделов меню (папка *menu*);
- файлы xml-описаний эффектов анимации (папка *anim*);
- импортированные текстовые файлы, в том числе файлы баз данных (папка *asset*).

Все ресурсы проекта при создании получают свой уникальный идентификатор, который записывается на этапе сборки проекта в файл ресурсов *R.txt*, находящийся в папке `\app\build\intermediates\runtime_symbol_list\debug`. Поскольку среда сама генерирует этот файл на этапе компиляции проекта, то редактировать его не имеет смысла.

Сборщик проекта

В качестве сборщика проекта AndroidStudio использует Gradle – продукт компании Apache. В проекте Gradle представлен плагином, входящим в состав SDK и подключающим сборщика Gradle при создании проекта. Подключение зависимостей проекта выполняется в файлах сборщика *build.gradle*. На рисунке 2.1.7 можно увидеть, что таких файлов два. Первый относится ко всему проекту и в нём происходит подключение компоновщика

```
dependencies {  
    classpath 'com.android.tools.build:gradle:3.6.3'  
    classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"  
}
```

Во втором файле, принадлежащем модулю проекта, содержатся сведения о структуре приложения, дублируемые в файле манифеста

```
android {  
    compileSdkVersion 29  
    buildToolsVersion "29.0.3"  
    defaultConfig {  
        applicationId "ru.samsung.itacademy.mdev.firstproject"  
        minSdkVersion 21  
        targetSdkVersion 29  
        versionCode 1  
        versionName "1.0"  
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  
    }  
}
```

и сведения о подключаемых библиотеках

```
dependencies {  
    implementation fileTree(dir: 'libs', include: ['*.jar'])  
    implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"  
    implementation 'androidx.appcompat:appcompat:1.0.2'  
    implementation 'androidx.core:core-ktx:1.0.2'  
    implementation 'androidx.constraintlayout:constraintlayout:1.1.3'  
    testImplementation 'junit:junit:4.12'  
    androidTestImplementation 'androidx.test.ext:junit:1.1.0'  
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.1.1'  
}
```

Упражнение 2.1.2 Работа в окне обозревателя проекта.

По умолчанию Android Studio использует представление проекта Android.

1.Переведите представление к видам Project, Package, Project Source File.

2.Во всех представлениях найдите папки src и res.

3.Раскройте все папки и просмотрите их содержимое.

4.Откройте в рабочей области содержимое файла AndroidManifest.xml. Выделите объект , изучите его свойства.

5.Измените в манифесте минимальную версию SDK на API 19, добавив объект

```
<uses-sdk android:minSdkVersion="19"/>
```

6.Установите разрешение на чтение внешнего диска

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

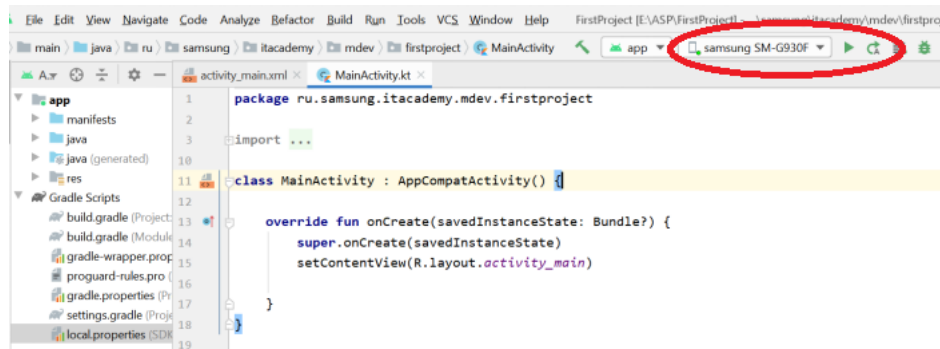
7.Откройте xml-описание иконки приложения res\drawable\ic_launcher_background.xml

8.Откройте папку res\mipmap с иконкой ic_launcher.png для разных параметров экрана и просмотрите все возможные разрешения.

2.1.4 Запуск приложения

Для проверки работоспособности приложения необходимо наличие устройства под управлением операционной системы Android не ниже версии, указанной в строке `minSdkVersion` файла `build.gradle` (Module). До запуска приложения нужно убедиться, что устройство распознаётся компьютером, а на самом гаджете включен режим разработчика и разрешена отладка приложений через USB. Включить данный режим можно из раздела настроек Об устройстве ->Номер сборки. Семикратное нажатие по этому разделу открывает новый пункт «Для разработчиков». В новом разделе нужно включить опцию «Отладка по USB». При отсутствии реального устройства, а порой и для ускорения тестирования, приложения запускают на виртуальном устройстве (эмуляторе). Можно установить сторонний эмулятор, а можно воспользоваться поставляемым в составе SDK. Следует учесть, что встроенный эмулятор HAXM работает на процессоре Intel, но с Android Studio 3.5 для процессора AMD в SDK Tools добавлен элемент Android Emulator Hypervisor Driver for AMD Processors. Это значит, что у операционной системы, имеющей установленную виртуальную оболочку Hyper-V, появилась возможность создания эмулятора на процессоре AMD. Как настроить виртуальную машину в операционной системе MS Windows10 можно узнать на [официальном сайте Microsoft](#) Создать новое устройство помогает *Менеджер виртуальных устройств*, вызываемый из меню Tools -> AVD Manager (Android Virtual Device Manager) или через пиктограмму на панели инструментов окна программы. Как создать новый эмулятор описано в [документации для разработчиков](#).

При подключении реального устройства, так же как и при созданном виртуальном, на панели инструментов в Android Studio появится имя подключенного устройства (рисунок 2.1.8). Запуск приложения выполняется через меню Run -> Run app, комбинацией клавиш Shift+F10 или кнопкой, расположенной на панели инструментов справа от имени устройства запуска.




Упражнение 2.1.3 Запуск приложения на устройстве и редактирование настроек.

1. Включите на телефоне/планшете режим отладки по USB и подключите устройство к компьютеру.
2. Убедитесь, что в окне выбора устройства отображается Ваше подключенное устройство.
3. Запустите приложение и дождитесь появления на экране телефона активности приложения.
4. Разблокируйте на устройстве поворот экрана и поверните устройство. Убедитесь в том, что активность приложения разворачивается при повороте экрана.
5. Закройте приложение и найдите на рабочем столе иконку запуска. Запустите приложение через эту иконку.
6. Откройте файл манифеста и отредактируйте свойства приложения:

6.1 Замените название "My_app" приложения на строку «Моё приложение», отредактировав свойство

```
android:label="Моё приложение"
```

6.2 Добавьте в папку ресурсов res\drawable файл  logo.png и замените иконку приложения на этот логотип. Для добавления файла достаточно скопировать его в буфер обмена и затем вставить в папку, используя контекстное меню.

6.3 Добавьте в объект активности свойство

```
<activity android:name=".MainActivity"
    android:screenOrientation="portrait">
```

7. Вновь запустите приложение и посмотрите изменения, которые произошли в приложении.
8. Закройте приложение и найдите его иконку запуска на рабочем столе. Убедитесь, что она изменилась.
9. В файле манифеста измените тему приложения на

```
android:theme="@style/Theme.AppCompat.Light.NoActionBar"
```

Ещё раз запустите приложение и убедитесь в отсутствии на экран панели заголовка.

Домашнее задание

Запуск приложения на эмуляторе устройства и редактирование настроек.

Настройте виртуальное устройство, изучите его интерфейс и выполните на эмуляторе задание 2.1.3. При назначении ориентации активности используйте горизонтальную ориентацию (landscape).

[Начать тур для пользователя на этой странице](#)