|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

**Отчет по практической работе №1**

по дисциплине «Разработка мобильных приложений»

|  |  |
| --- | --- |
| **Выполнили:**  Студенты группыИКБО-20-23 | Комисарик М.А. |
| **Проверил:**  Старший преподаватель кафедры МОСИТ | Шешуков Л.С. |

Москва 2025 г.

СОДЕРЖАНИЕ

[**РТУ МИРЭА** 1](#_Toc191493399)

[СОДЕРЖАНИЕ 2](#_Toc191493400)

[1 ТЕОРЕТИЧЕСКОЕ ВВЕДЕНИЕ 3](#_Toc191493401)

[1.1 Установка Android Studio 3](#_Toc191493402)

[1.2 Создание проекта 5](#_Toc191493403)

[1.3 Структура проекта 9](#_Toc191493404)

[1.4 Запуск проекта 11](#_Toc191493405)

[1.5 Создание графического интерфейса 20](#_Toc191493406)

[1.6 Верстка в Android. Язык XML 23](#_Toc191493407)

[1.7 Ресурсы в Android 25](#_Toc191493408)

[1.8 Создание интерфейса пользователя с помощью ресурса 28](#_Toc191493409)

[1.9 Компоненты разметки 29](#_Toc191493410)

[1.9.1 TextView 29](#_Toc191493411)

[1.9.2 EditText 29](#_Toc191493412)

[1.9.3 Button 30](#_Toc191493413)

[1.9.4 ImageView 31](#_Toc191493414)

[2 ПРАКТИЧЕСКОЕ ЗАДАНИЕ 32](#_Toc191493415)

[2.1 Создание проекта 32](#_Toc191493416)

[2.2 Разметка 32](#_Toc191493417)

[2.2.1 Настройка корневых элементов 32](#_Toc191493418)

[2.2.2 TextView 35](#_Toc191493419)

[2.2.3 EditText 36](#_Toc191493420)

[2.2.4 Button 37](#_Toc191493421)

[2.2.5 ImageView 39](#_Toc191493422)

[2.3 Результат 40](#_Toc191493423)

[ЗАКЛЮЧЕНИЕ 43](#_Toc191493424)

# ТЕОРЕТИЧЕСКОЕ ВВЕДЕНИЕ

## Установка Android Studio

Множество сред разработки поддерживают создание приложений для Android, но наиболее предпочтительной считается Android Studio. Она разработанна специально для Android, предлагает интегрированные инструменты и оптимизированный рабочий процесс, что делает ее наиболее предпочтительным выбором для разработчиков. Для начала работы с Android Studio, необходимо скачать установщик с официального сайта: <https://developer.android.com/studio>. На рисунке Рисунок 1 представлен раздел для скачивания установщика на официальном сайте.

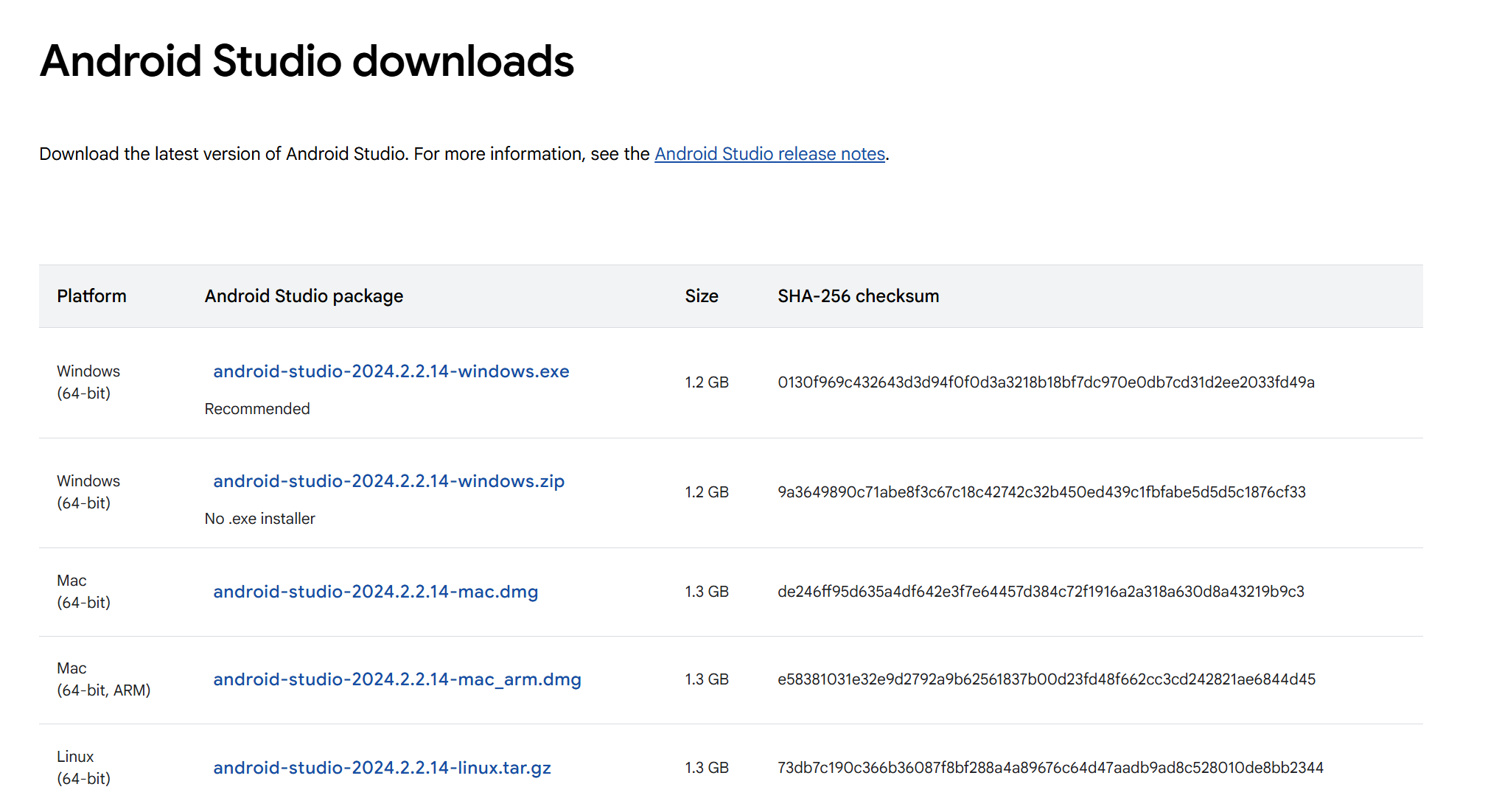


Рисунок 1 – Раздел загрузки установщика Android Studio

Android Studio это очень требовательная среда разработки и требует много системных ресурсов. Поэтому для комфортной работы рекомендуется высокопроизводительный компьютер. Например, рекомендуемый объем оперативной памяти составляет как минимум 8 ГБ.

Установщик Android Studio предлагает выбрать, какие компоненты установить. В списке присутствует Android Studio — основная среда разработки, содержащая инструменты для создания, тестирования и отладки Android-приложений. Также доступен Android Virtual Device (AVD) — эмулятор Android-устройств, который позволяет запускать и тестировать приложения без необходимости использования физического устройства (Рисунок 2).

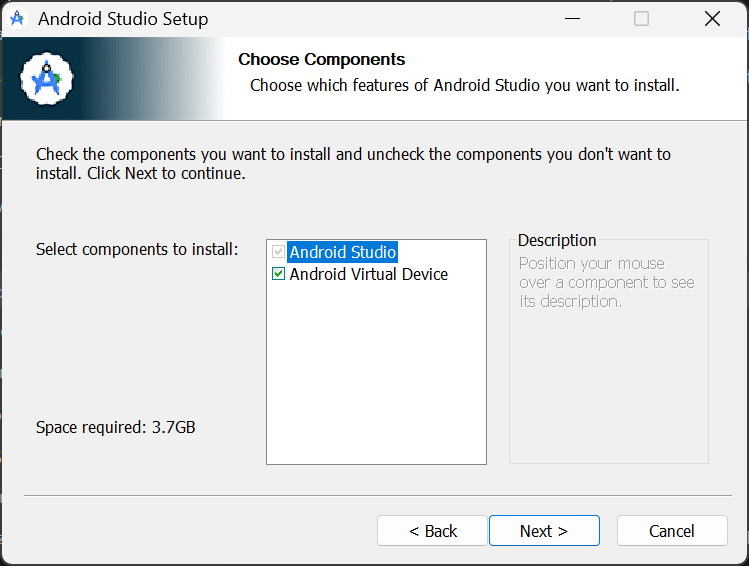


Рисунок 2 – Выбор компонентов при установке Android Studio

Для разработки в Android Studio также требуется установить Android Software Development Kit (SDK). Этот набор инструментов является обязательным для создания и разработки приложений, так как включает в себя библиотеки API, эмуляторы устройств, инструменты для отладки и профилирования и многое другое.

При первом запуске Android Studio среда автоматически проверит наличие всех необходимых для работы компонентов и предложит их установить или указать их местоположение на компьютере. К этим компонентам, помимо Android SDK, могут относиться различные дополнения и инструменты, которые улучшат и облегчат процесс разработки. Например, это могут быть дополнительные пакеты API, инструменты для работы с базами данных, утилиты для профилирования производительности и другие полезные ресурсы, которые можно установить позже самостоятельно (Рисунок 3).

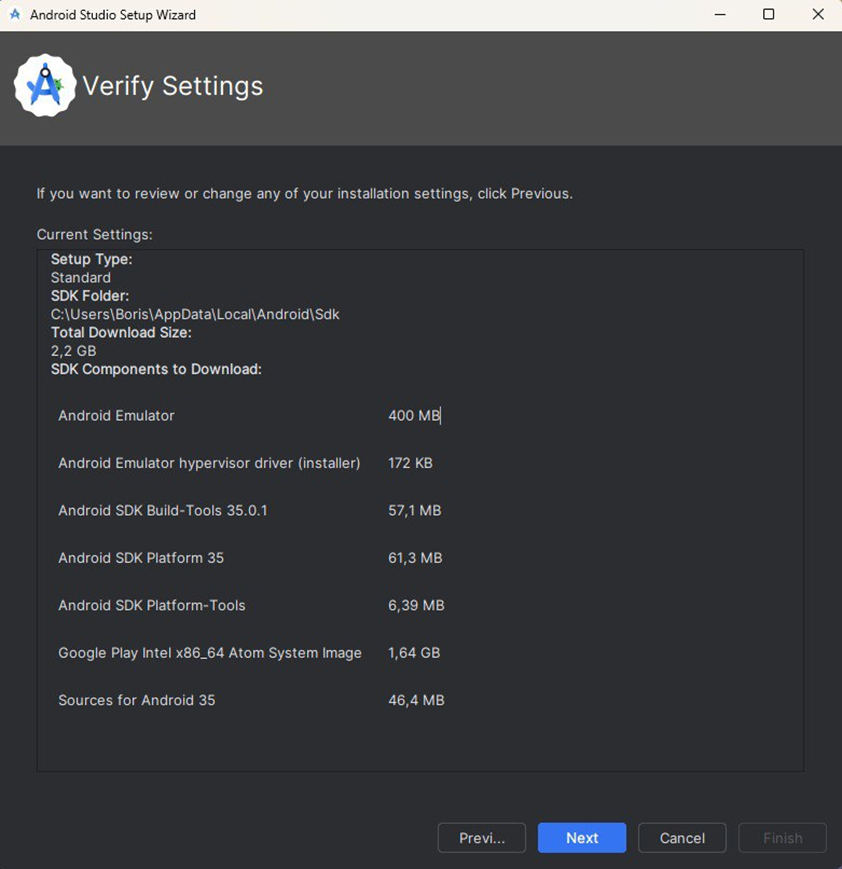


Рисунок 3 – Подтверждение настроек установки Android Studio по умолчанию

## Создание проекта

Чтобы начать разработку приложения требуется создать новый проект. Для этого на начальном экране требуется выбрать кнопку “New project” (Рисунок 4).

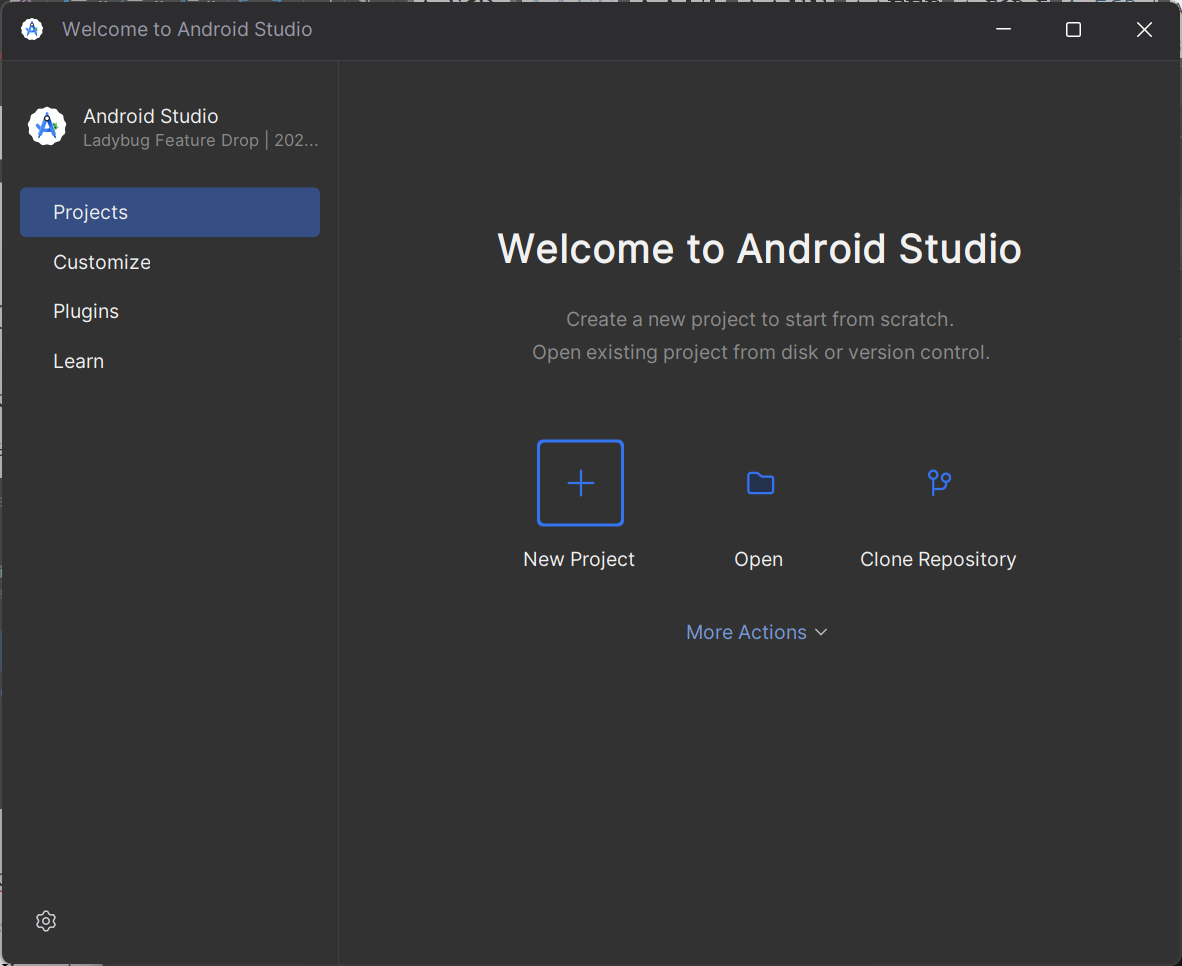


Рисунок 4 – Начальный экран Android Studio

Android Studio предлагает различные шаблоны для нового проекта. Шаблоны активностей в Android Studio помогают разработчикам быстрее создавать проекты, предоставляя готовые структуры с преднастроенными элементами интерфейса и логики. Они позволяют сократить время на базовую настройку, следовать лучшим практикам Android-разработки и избежать рутинного создания одинаковых компонентов. В зависимости от целей проекта можно выбрать пустую активность для полной кастомизации, готовые варианты с навигацией, адаптивными интерфейсами или даже шаблоны для работы с C++ и искусственным интеллектом (Рисунок 5).

Существуют разные типы шаблонов. Шаблоны Phone and Tablet ориентированы на смартфоны и планшеты, Wear OS предназначены для разработки приложений для умных часов на платформе Wear OS, Television – для создания приложений для телевизоров, Automotive – для разработки приложений для автомобильных систем.

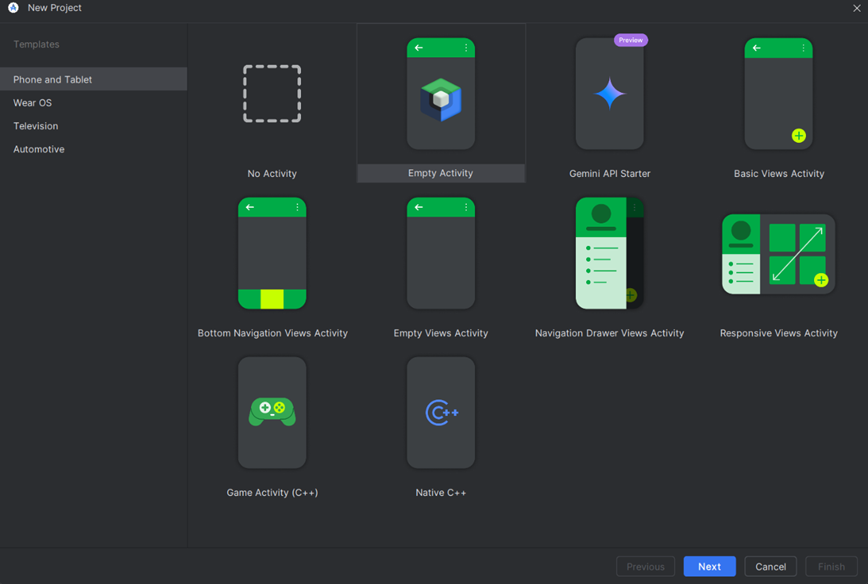


Рисунок 5 – Выбор шаблона нового проекта

Опишем вкратце каждый из предложенных шаблонов в разделе Phone and Tablet:

* No Activity: создаёт проект без активностей, используется для полностью кастомных решений, например, при разработке на Jetpack Compose или если нужен только фоновый процесс;
* Empty Activity: минимальная активность с одной разметкой, содержит MainActivity и activity\_main.xml, стандартный выбор для создания кастомного UI;
* Gemini API Starter: шаблон с преднастроенной интеграцией Gemini API от Google, предназначен для разработки приложений с искусственным интеллектом;
* Basic Views Activity: активность с базовыми элементами интерфейса, такими как FloatingActionButton и AppBar, используется для простых UI-экранов;
* Bottom Navigation Views Activity: создаёт активность с нижней навигационной панелью, предназначена для приложений с несколькими основными разделами;
* Empty Views Activity: активность без предустановленных UI-элементов, используется, если требуется чистый холст для кастомного интерфейса;
* Navigation Drawer Views Activity: активность с боковым меню (Navigation Drawer), подходит для приложений со сложной навигацией и несколькими разделами;
* Responsive Views Activity: создаёт адаптивный макет, который автоматически подстраивается под разные экраны, полезно для разработки под планшеты и складные устройства;
* Game Activity (C++): шаблон для разработки игр, использует NativeActivity и предоставляет доступ к C++ для высокой производительности;
* Native C++: проект с основной логикой на C++, включает взаимодействие с JNI (Java Native Interface), используется для низкоуровневого программирования и работы с производительным кодом.

После этого в появившемся окне необходимо заполнить настройки проекта (Рисунок 6).

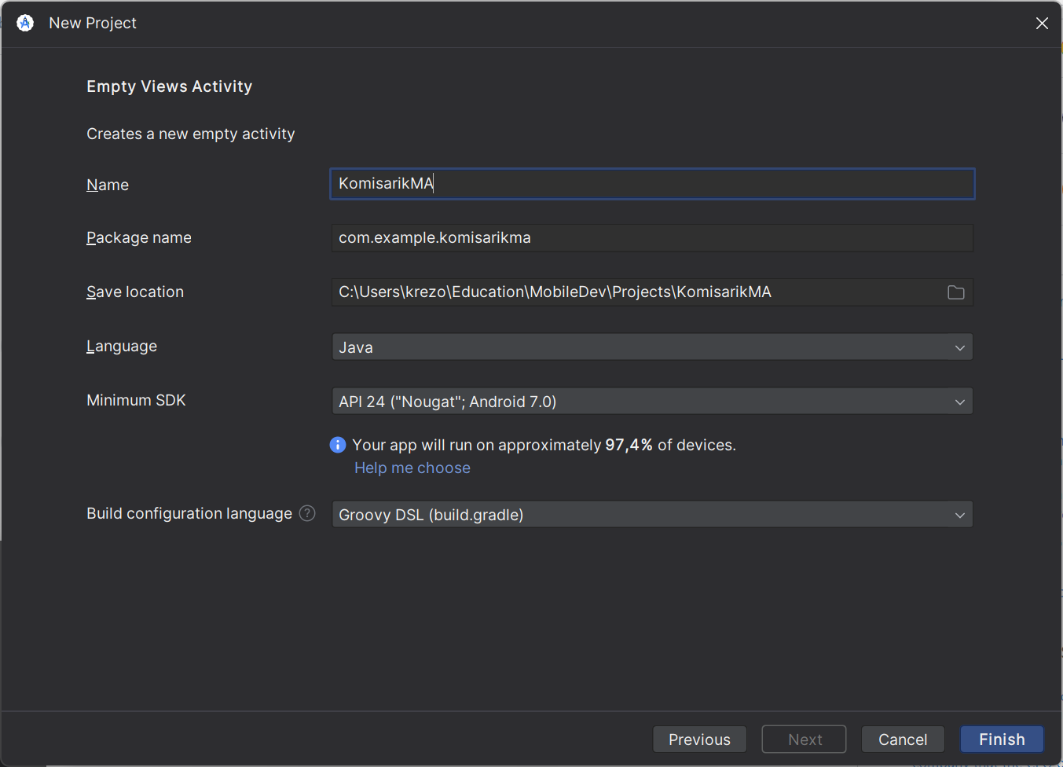


Рисунок 6 – Заполнение настроек проекта

На данном окне представлено 6 настроек:

* Name: отвечает за название проекта;
* Package name: название пакета. Уникальный идентификатор приложения, обычно следует формату com.example.myapp;
* Save location: Путь к папке, где будет храниться проект;
* Language: Язык программирования, на котором будет написан проект;
* Minimum SDK: Минимальная версия Android, которую будет поддерживать выше приложение;
* Build configuration language: Язык конфигурации системы сборки.

Чтобы завершить настройку проекта следует нажать кнопку “Finish”.

## Структура проекта

Структура проекта изображена на Рисунке Рисунок 7.

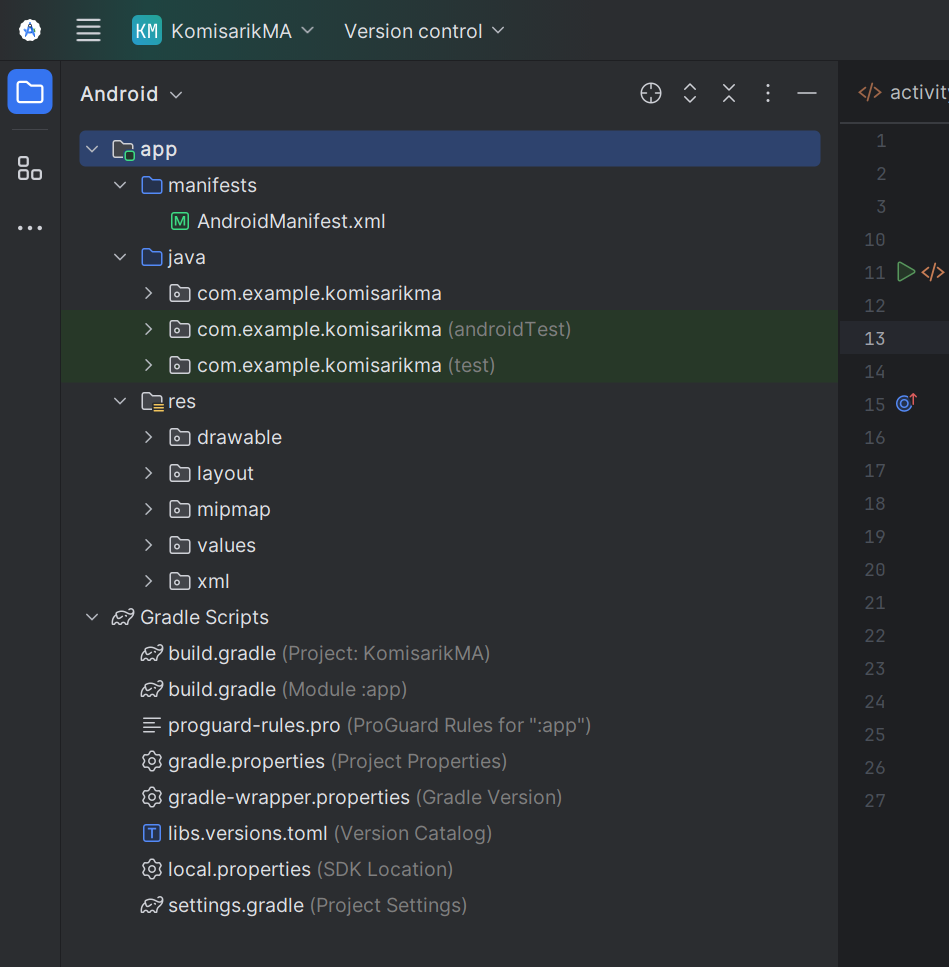


Рисунок 7 – Структура проекта на Android Studio

Модуль "app" является основным компонентом проекта Android и содержит файлы, необходимые для сборки проекта. В нём хранятся следующие элементы:

1. manifests: содержит файл AndroidManifest.xml, основной файл конфигурации Android-приложения, в котором описаны его компоненты (активности, службы, приёмники широковещательных сообщений и поставщики контента), необходимые разрешения (например, доступ в интернет, использование камеры или отправка уведомлений), а также ключевые настройки, такие как название, тема, иконка, минимальная и целевая версия SDK. Минимальная версия SDK (minSdkVersion) определяет самую раннюю версию Android, на которой приложение может быть запущено, а целевая версия SDK (targetSdkVersion) указывает версию Android, для которой приложение оптимизировано. Манифест также определяет точку входа в приложение, фильтры интентов для обработки определённых действий и параметры безопасности, влияющие на взаимодействие приложения с другими системами,
2. java: содержит три пакета, отвечающих за: код приложения, код для инструментальных тестов, выполняющихся на Android устройствах и код для модульных тестов, выполняющихся на вашем компьютере.
3. res: содержит все ресурсы, не связанные с кодом, такие как XML-макеты, UI элементы (кнопки, текстовые поля), строки, изображения (drawables), стили и темы.

Модуль "Gradle Scripts" управляет процессом сборки проекта. В нём хранятся следующие элементы:

1. build.gradle (Project: «Название вашего проекта»): Этот файл на уровне проекта содержит конфигурацию, применяемую ко всем модулям в проекте, а также включает ссылки на плагины Gradle, используемые проектом.
2. build.gradle (Module :app): Этот файл на уровне модуля содержит конфигурацию сборки, специфичную для данного модуля. Здесь определяются настройки конкретного модуля, такие как версия SDK, версии зависимостей, конфигурации сборки (например, отладка и выпуск) и другие настройки Android, специфичные для модуля.
3. settings.gradle: Содержит ссылки на модули, включенные в проект. Каждый модуль, который должен быть собран как часть проекта, должен быть заявлен здесь.

## Запуск проекта

Созданный проект можно запустить как на реальном, так и на виртуальном устройства. Далее будут рассмотрены оба варианта.

Для запуска проекта на физическом устройстве, на нем необходимо активировать режим разработчика и разрешить отладку по USB.

Ниже указана последовательность действий для активации режима разработчика на телефоне Xiaomi Redmi Note 9 Pro.

Чтобы активировать режим разработчика, требуется нажать 7 раз на надпись «Версия MIUI» в приложении «Настройки» в разделе «О телефоне» (Рисунок 8). После этого появится уведомление о том, что пользователь стал разработчиком, или, что пользователь уже разработчик.

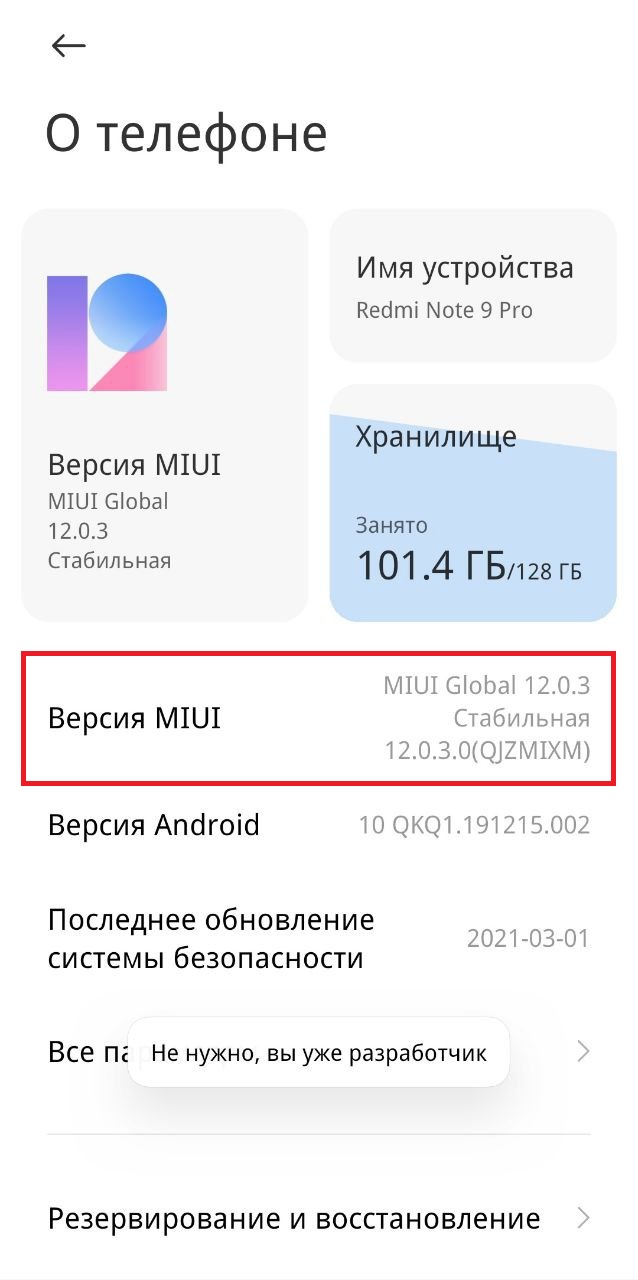


Рисунок 8 – Активация режима разработчика

Затем в разделе «Расширенные настройки» появится раздел «Для разработчиков» (Рисунок 9).

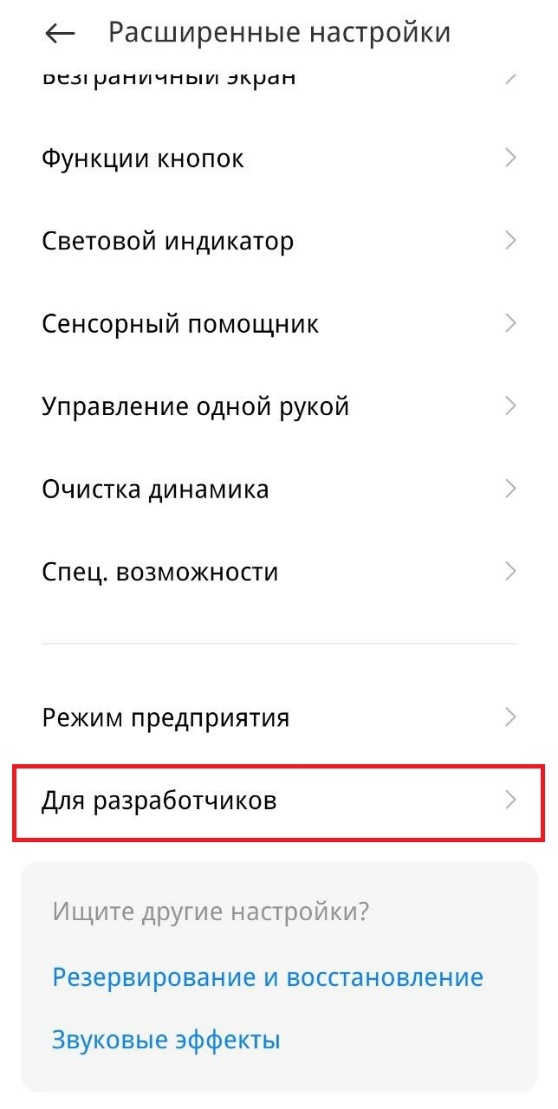


Рисунок 9 – Раздел «Для разработчиков»

В этом разделе требуется включить опцию режим разработчика и разрешить отладку по USB (Рисунок 10).

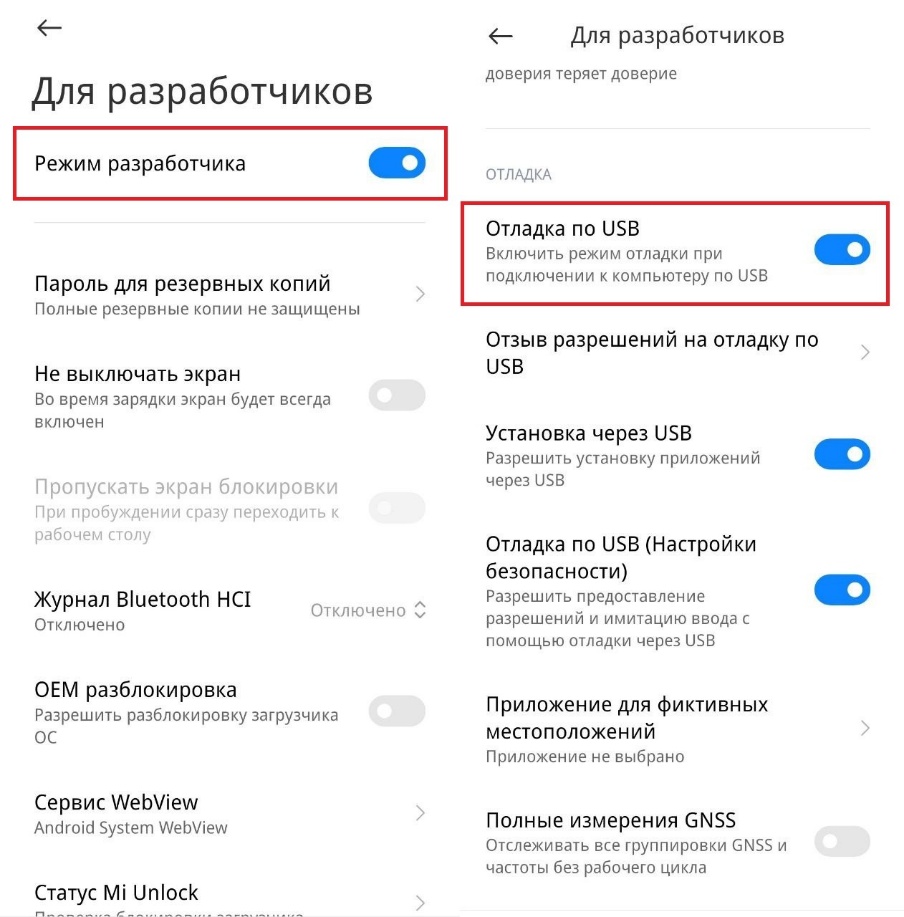


Рисунок 10 – Включение опции «Отладка по USB»

Далее нужно подключить устройство к компьютеру с помощью USB провода с открытым Android Studio. Программа должна автоматически обнаружить устройство и отобразить его в разделе “Available Devices” (Рисунок 11). Для запуска проекта надо нажать зеленую кнопку справа (или сочетание клавиш Ctrl+F5).

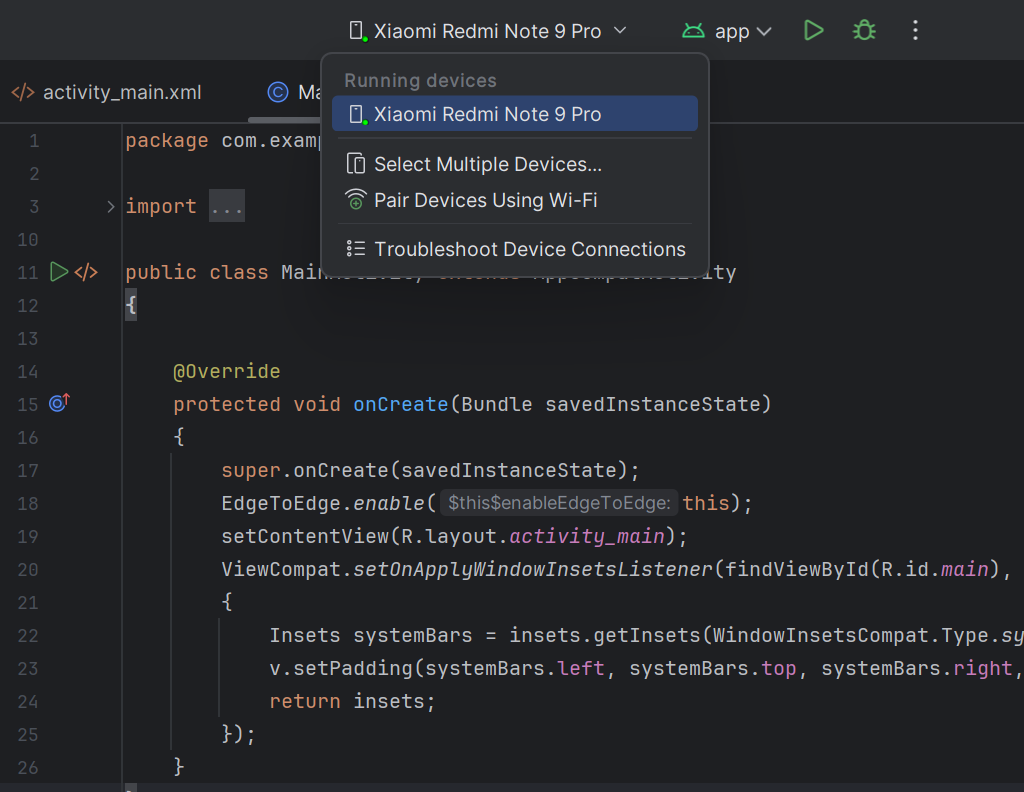


Рисунок 11 – Подключение физического устройства для отладки

После сборки приложения оно откроется на телефоне (Рисунок 12).



Рисунок 12 – Приложение, запущенное на физическом устройстве

Для запуска на виртуальном устройстве необходимо создать его в Android Studio. Чтобы это сделать нужно перейти в пункт “Tools” и выбрать раздел “Device Manager” (Рисунок 13).

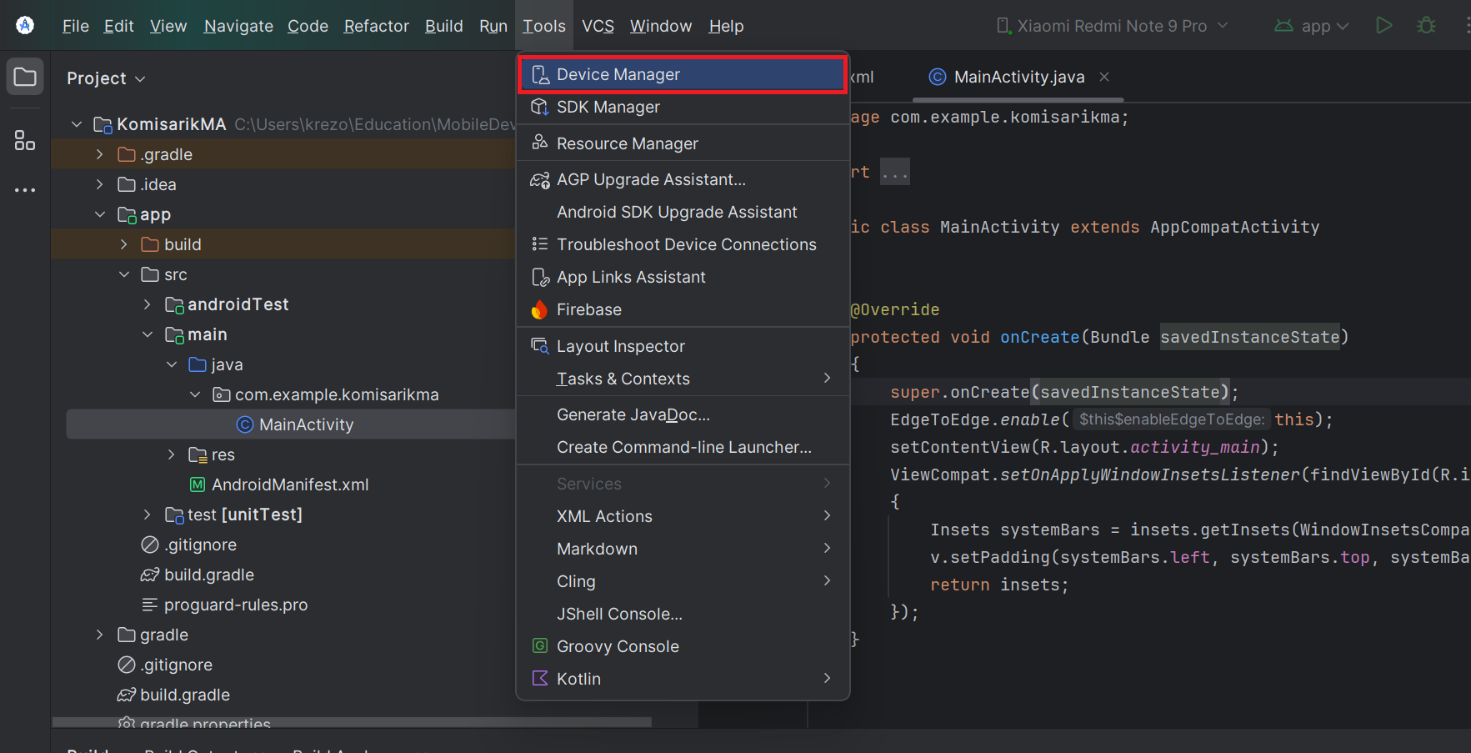


Рисунок 13 – Раздел “Device Manager”

Затем, в открывшемся боковом окне нажать «+» для создания нового устройства (Рисунок 14).

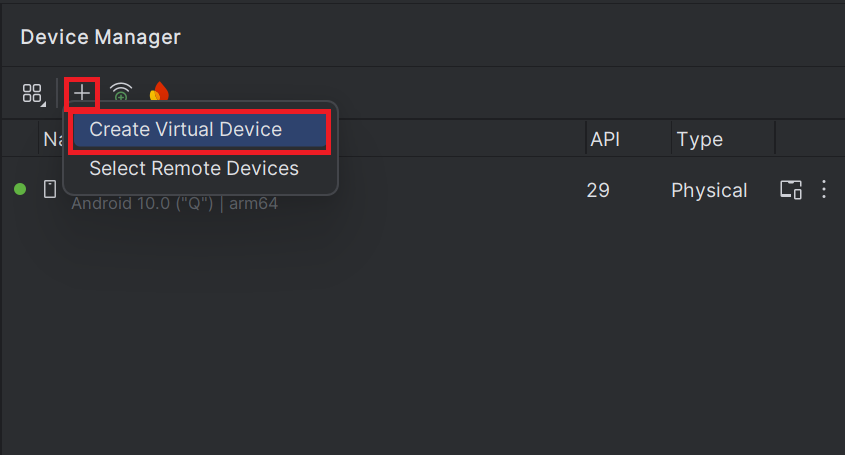


Рисунок 14 – Создание устройства в менеджере устройств

Теперь нужно выбрать параметры устройства. Сначала настраивается тип устройства и его размеры (Рисунок 15).

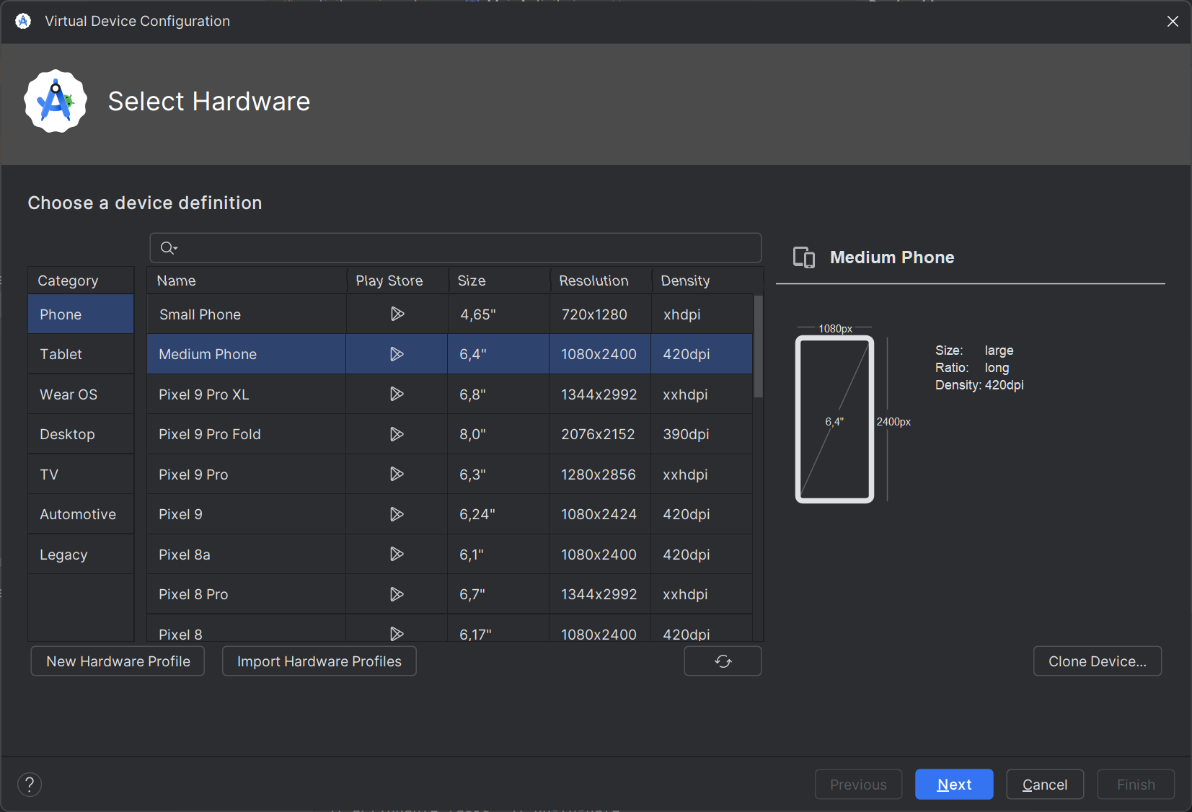


Рисунок 15 – Настройка типа виртуального устройства и его размеров

Далее выбирается версия операционной системы (ОС) Android, которая будет установлена на устройство. Если перед этим требуемая ОС не скачана, её можно скачать, для этого надо нажать на кнопку загрузки рядом с версией ОС (Рисунок 16).

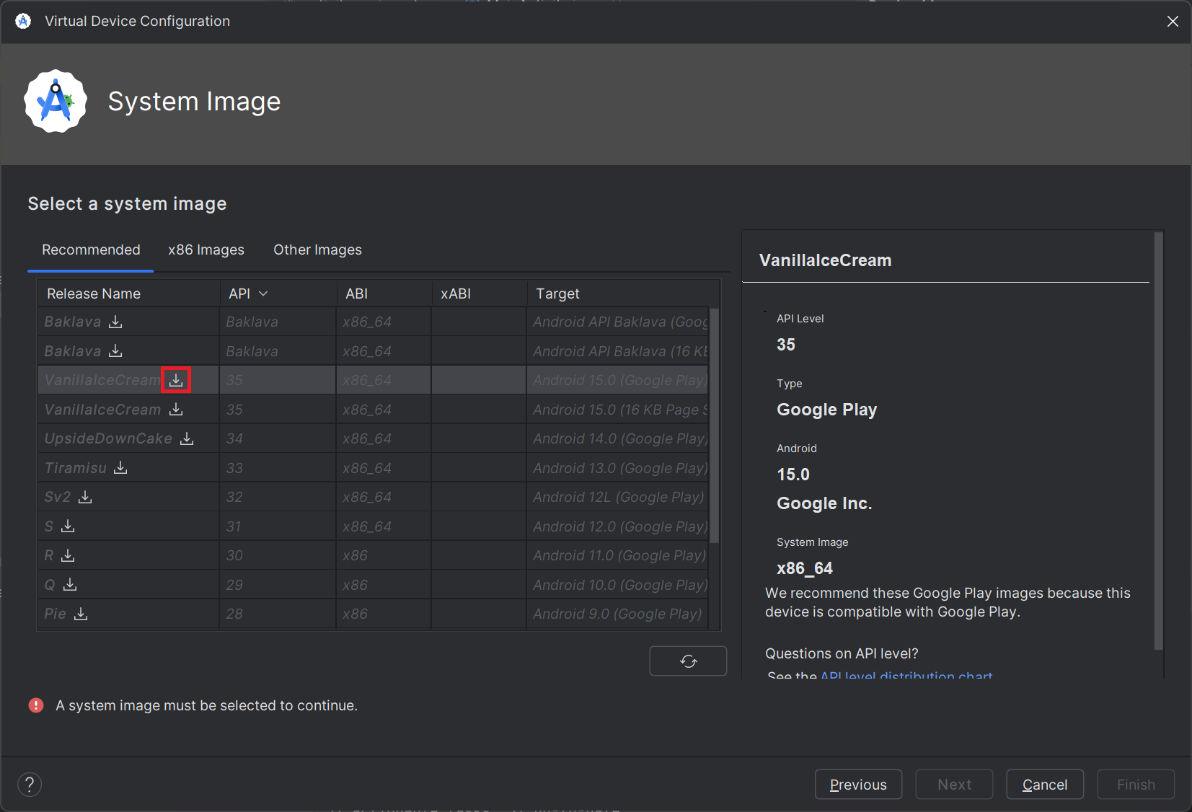


Рисунок 16 – Выбор версии ОС Android на виртуальное устройство

После выбора устройства и версии Android можно еще раз посмотреть выбранные характеристики, выбрать тип ориентации экрана и создать устройство (Рисунок 17).

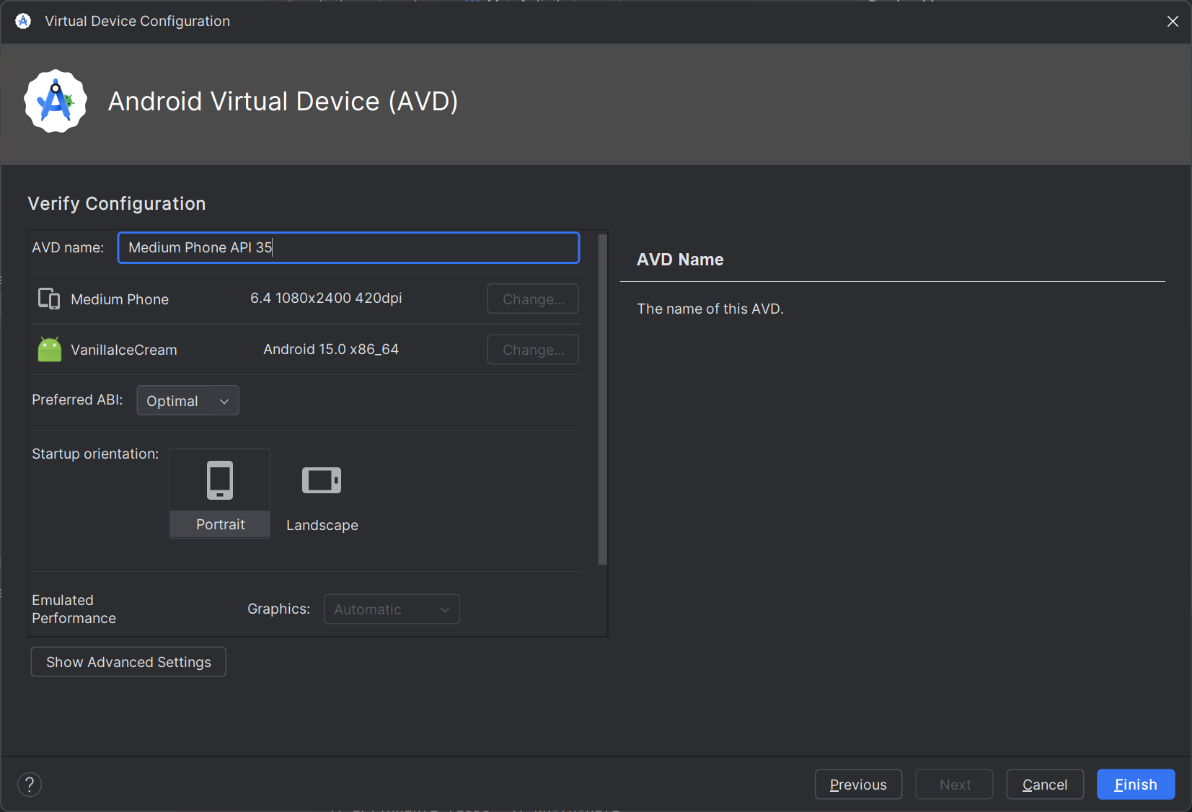


Рисунок 17 – Завершение создания виртуального устройства

Когда устройство создано, его можно выбрать в качестве основного в разделе “Available Devices” и запустить приложение на нем (Рисунок 18).

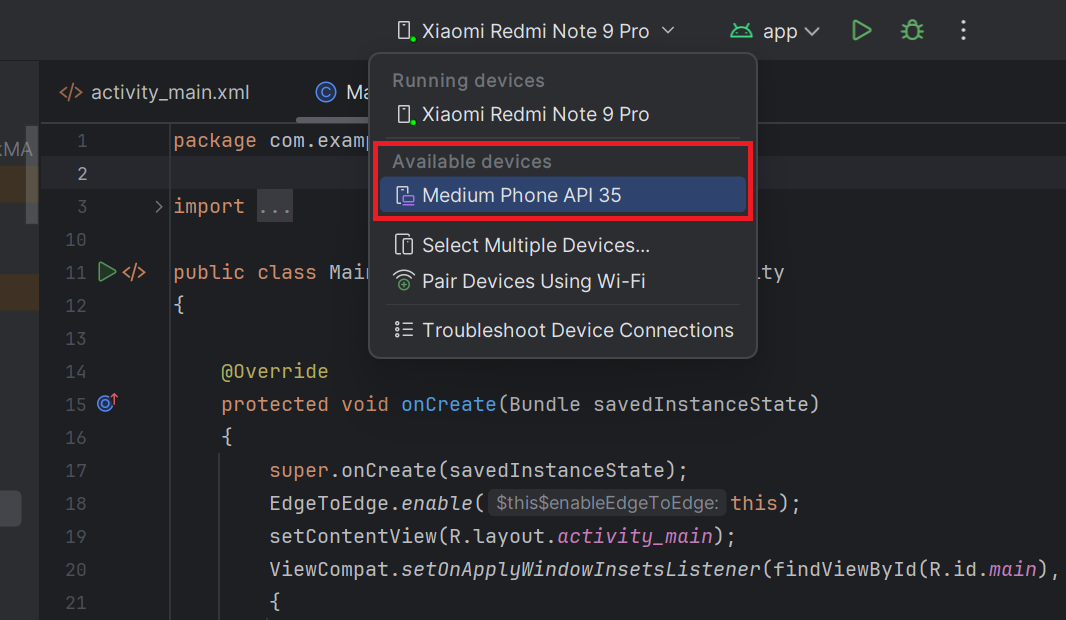


Рисунок 18 – Выбор виртуального устройства в качестве основного

Запущенное устройство будет доступно в окне “Running Devices”, которое автоматически откроется при окончании сборки проекта (Рисунок 19). Функционал приложения полностью идентичен с физическим устройством.

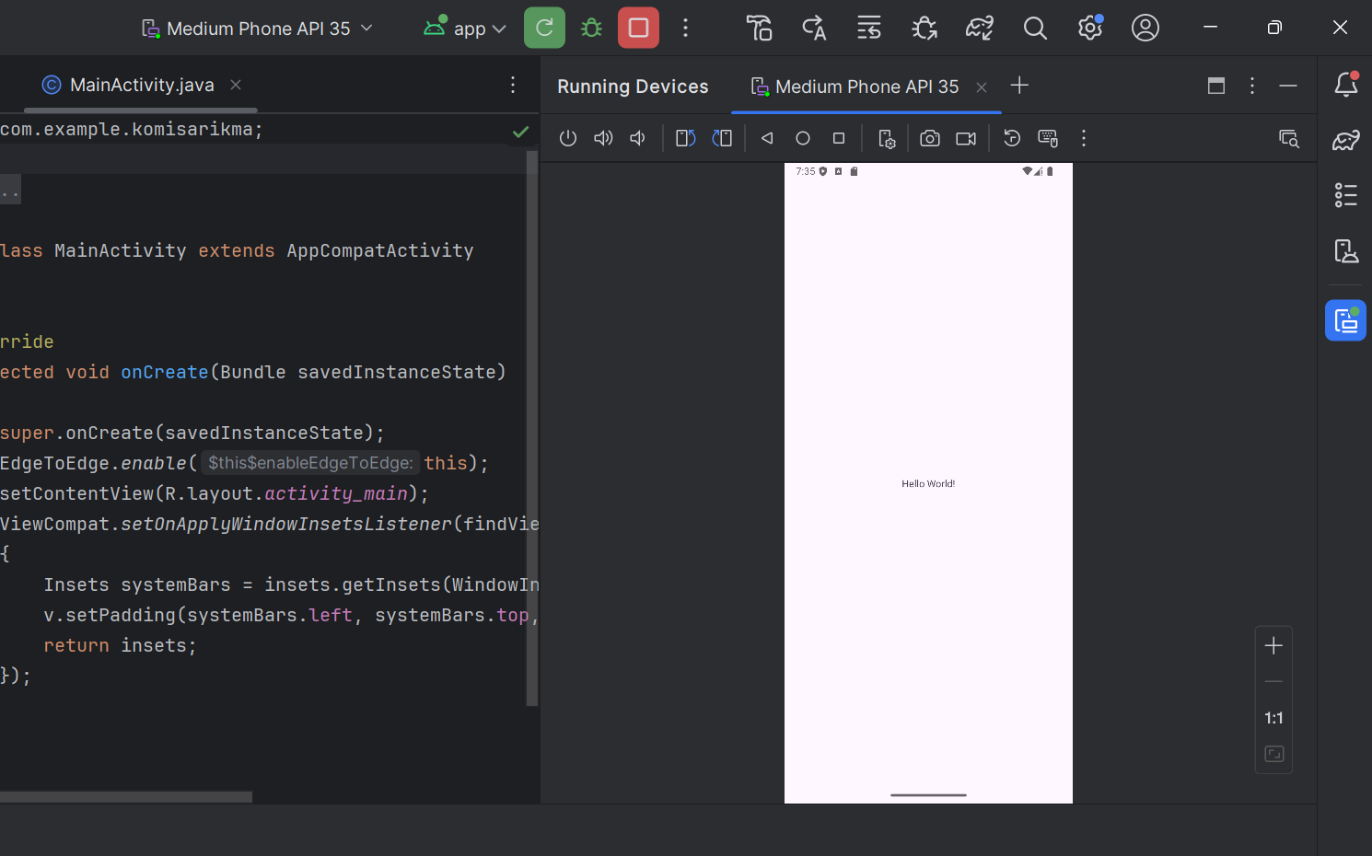


Рисунок 19 – Виртуальное устройство в окне “Running Devices”

И на физическом, и на виртуальном устройстве приложение можно запустить как в стандартном режиме с помощью нажатия на кнопку со значком «cтрелки», так и в режиме отладки с помощью нажатия на кнопку со значком «жука» правее (Рисунок 20).

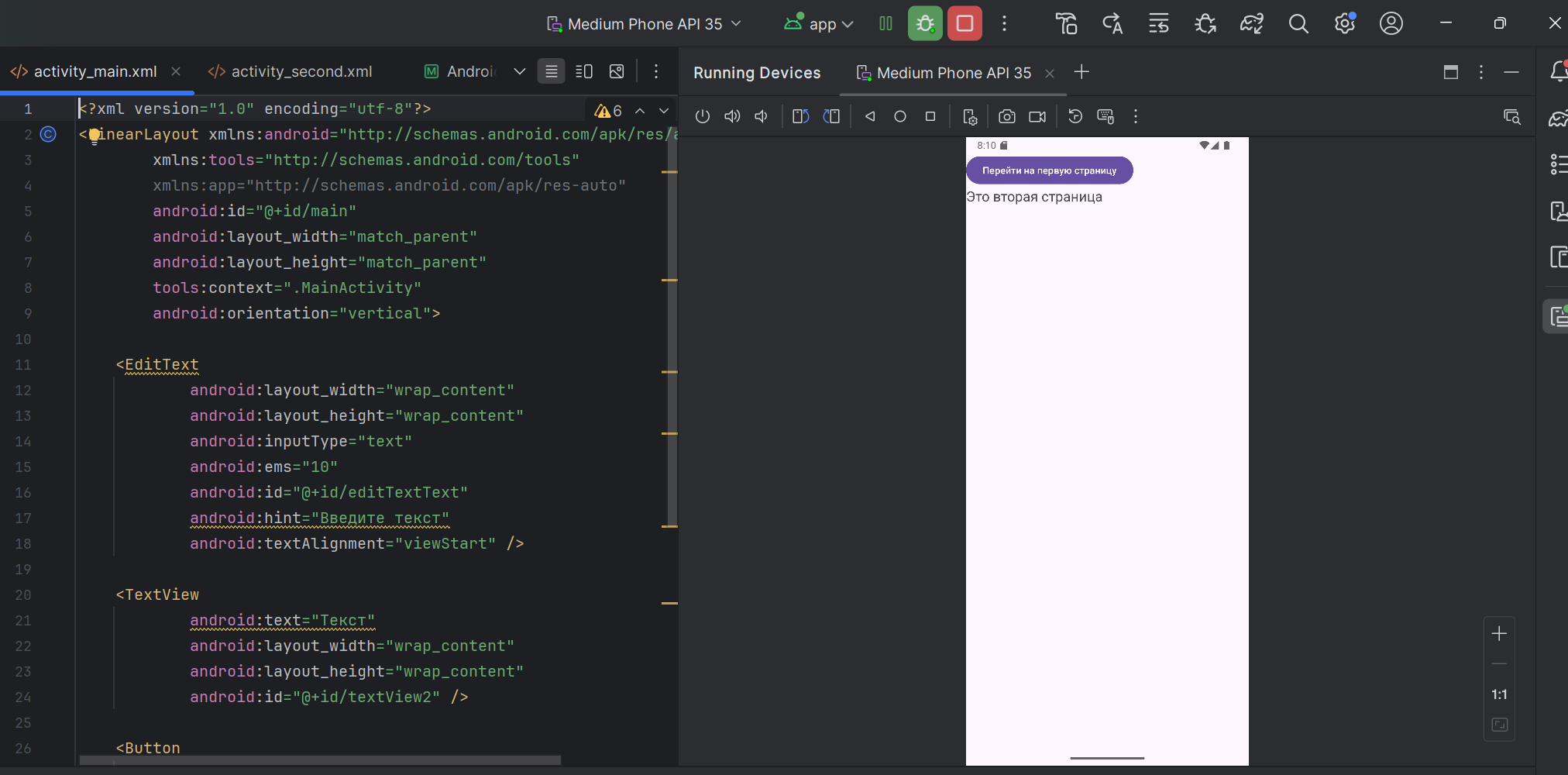


Рисунок 20 – Виртуальное устройство в режиме отладки

В режиме отладки можно устанавливать брейкпоинты и исследовать состояние системы (просмотр значения переменных, пошаговое исполнение кода).

## Создание графического интерфейса

Выполнение приложения Android по умолчанию начинается с класса MainActivity, который по умолчанию открыт в Android Studio (Рисунок 21).

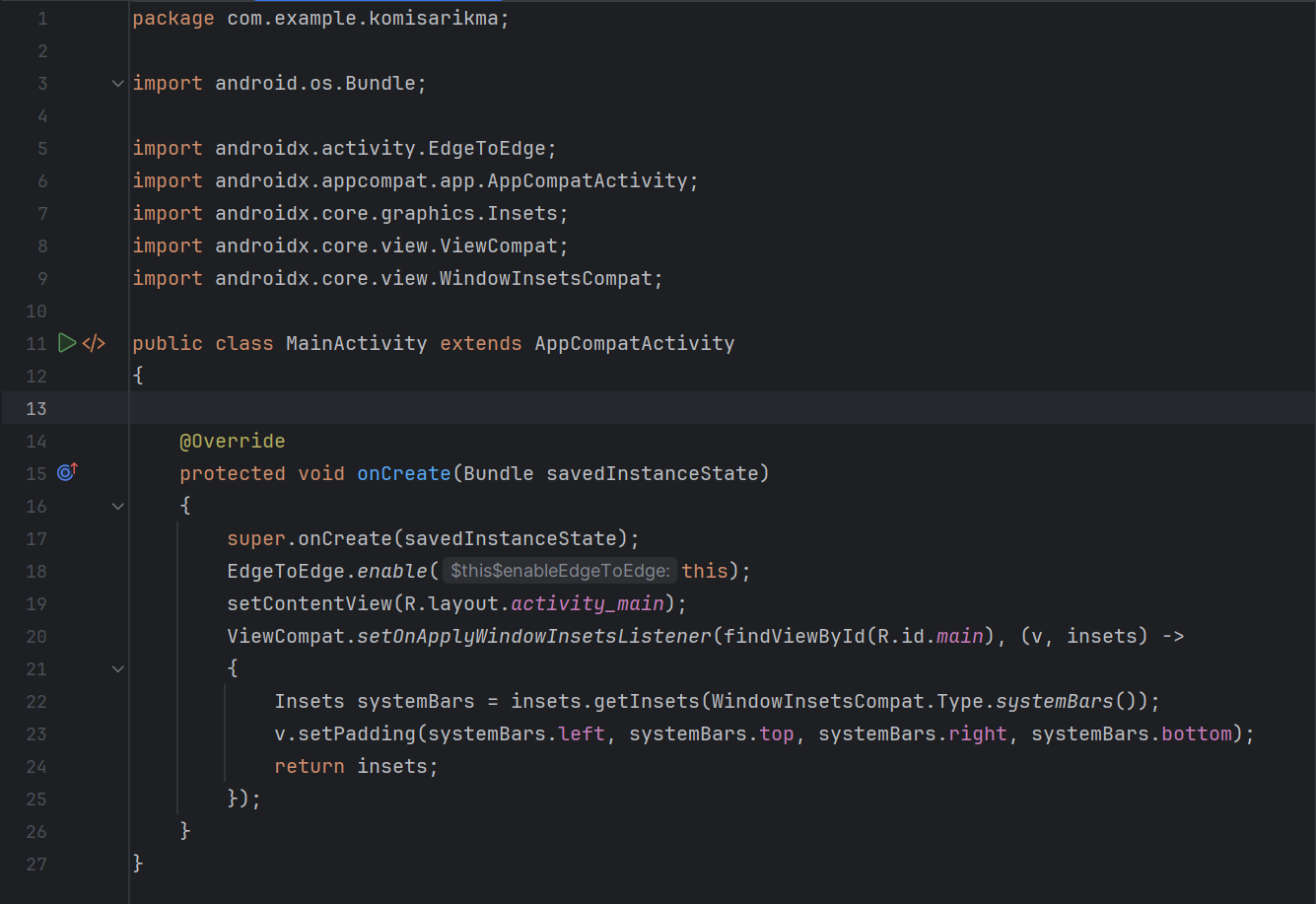


Рисунок 21 – Класс MainActivity

Каждый отдельный экран или страница в приложении описывается таким понятием как activity. В литературе могут использоваться различные термины: экран, страница, активность. По умолчанию MainActivity содержит только один метод onCreate(), в котором создается весь интерфейс приложения.

В метод setContentView() передается ресурс разметки графического интерфейса. В этом файле определяется, какой визуальный интерфейс будет иметь MainActivity (Рисунок 21).

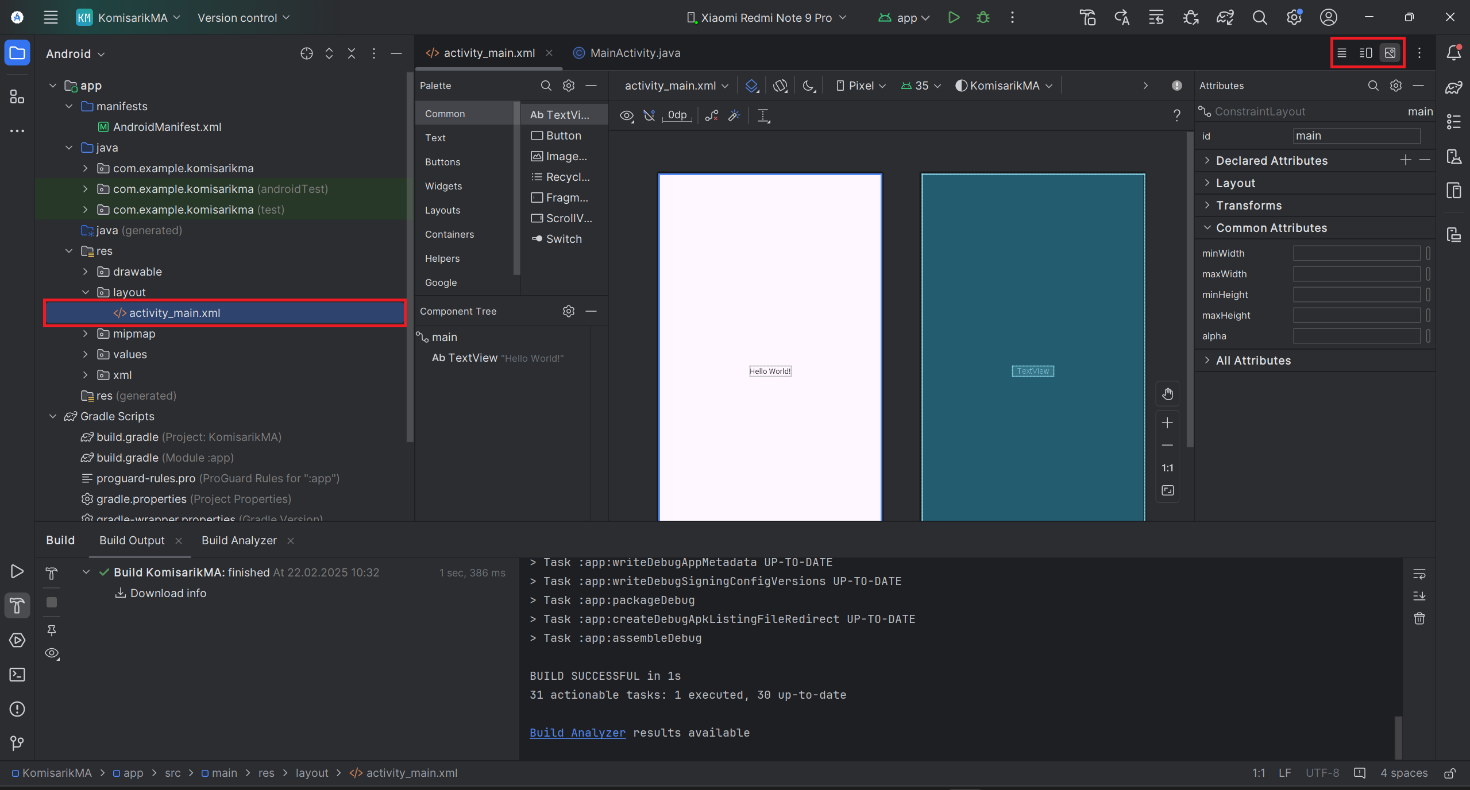


Рисунок 22 – Файл разметки графического интерфейса

Android Studio позволяет работать с визуальным интерфейсом как в режиме кода, так и в графическом режиме. Так, по умолчанию файл открыт в графическом режиме. В графическом режиме настраивание интерфейса происходит с помощью “Drag and Drop”, элементы графического интерфейса перетаскиваются курсором на экран телефона, все изменения сразу же видны (Рисунок 22).

Но также можно работать с файлом в режиме кода, поскольку activity\_main.xml — это xml файл. Для переключения к коду нужно нажать на кнопку Code над графическим представлением (Рисунок 23).

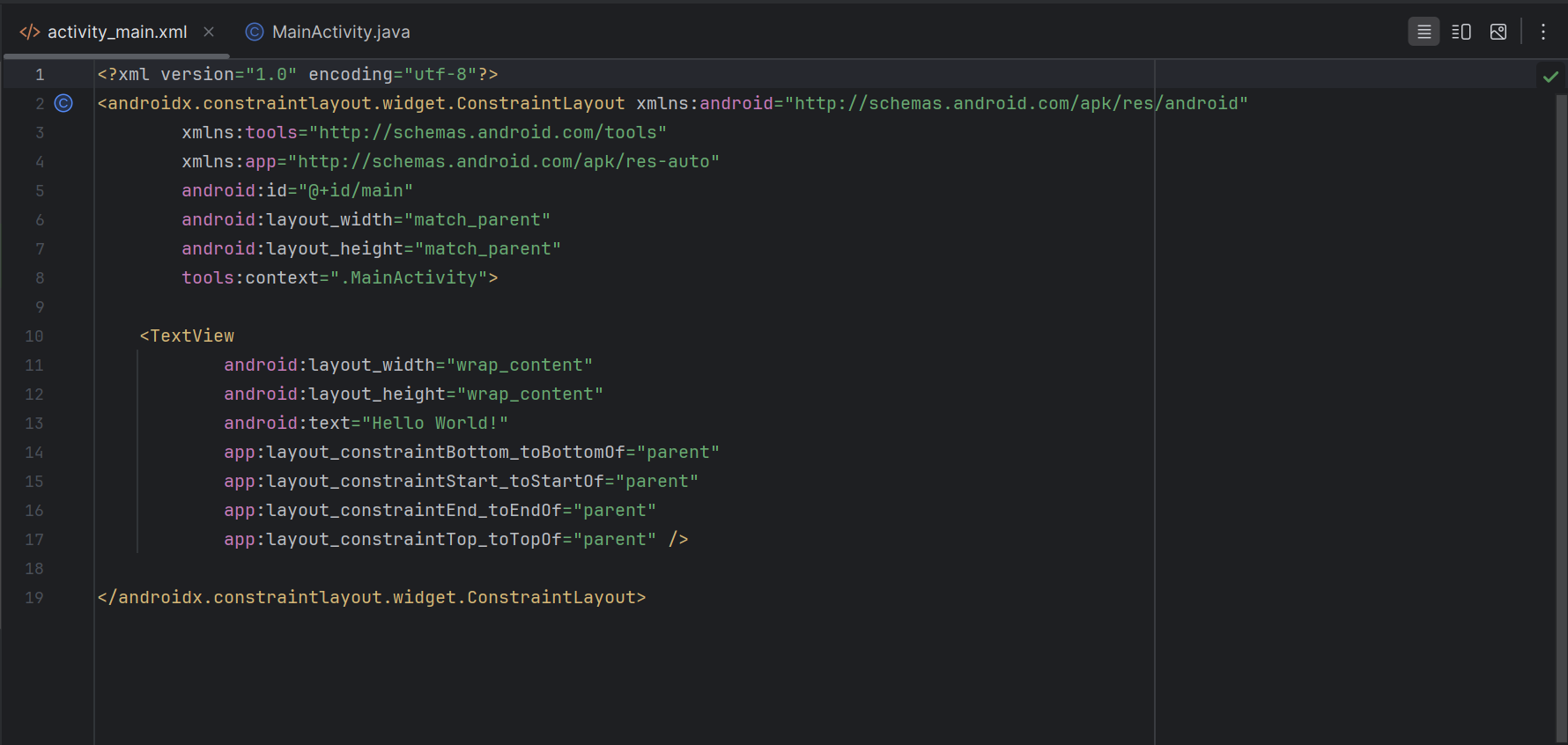


Рисунок 23 – Редактирование файла разметки в режиме кода

С помощью кнопки Split можно переключиться на комбинированное представление код + графический дизайнер (Рисунок 24).

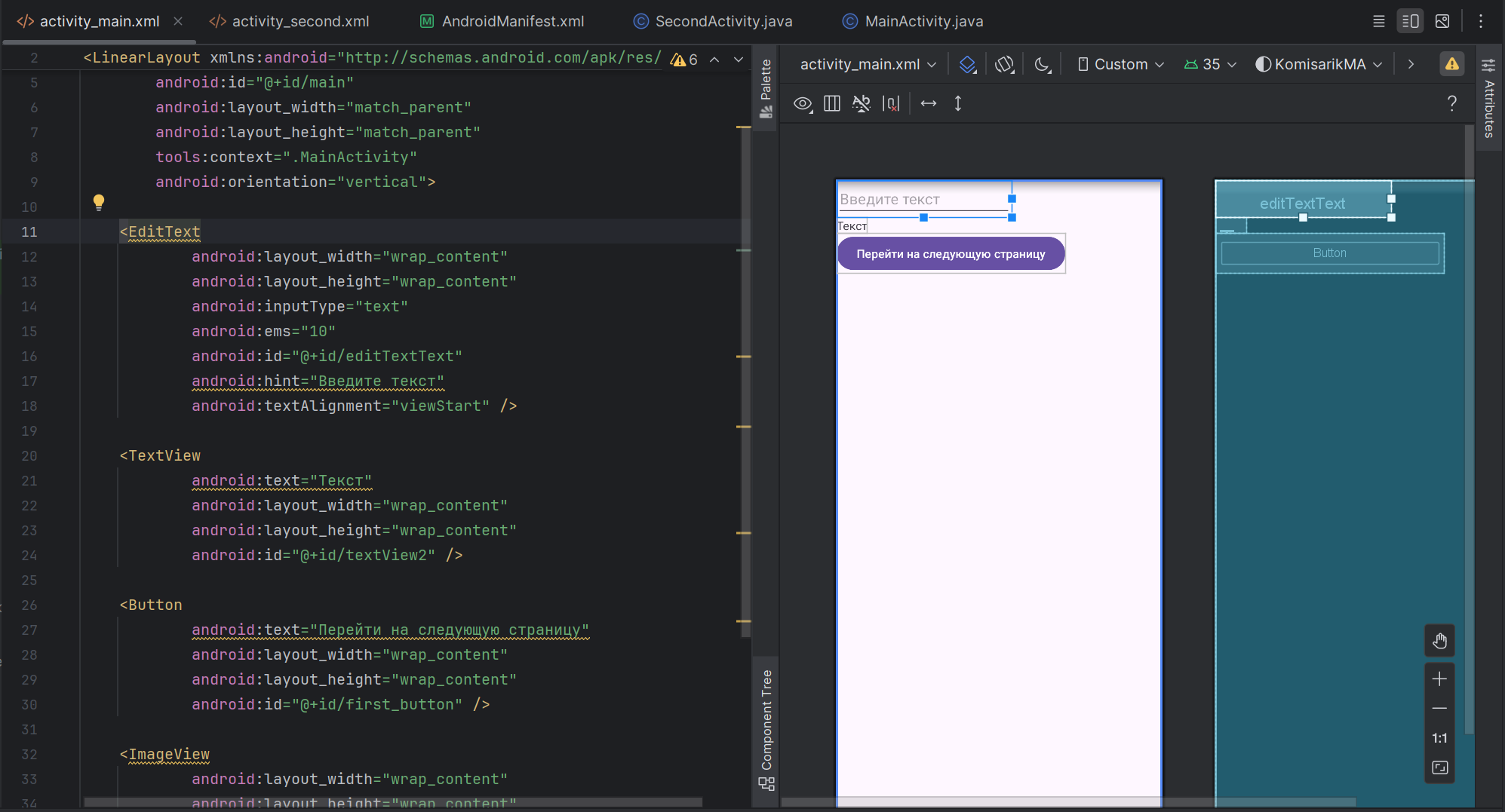


Рисунок 24 – Редактирование файла разметки в комбинированном режиме

Большинство визуальных элементов, наследующихся от класса View, такие как кнопки, текстовые поля и другие, располагаются в пакете android.widget.

При определении визуального интерфейса есть три стратегии:

* Создать элементы управления программно в коде java.
* Объявить элементы интерфейса в XML.
* Сочетание обоих способов - базовые элементы разметки определить в XML, а остальные добавлять во время выполнения.

## Верстка в Android. Язык XML

Как правило, для определения ресурсов, а в том числе и визуального интерфейса, в проектах под Android используются специальные файлы xml. Эти файлы являются ресурсами разметки и хранят определение визуального интерфейса в виде кода eXtensible Markup Language (XML).

Объявление пользовательского интерфейса в файлах XML позволяет отделить интерфейс приложения от кода. Что означает, что можно изменять определение интерфейса без изменения кода java. Кроме того, объявление разметки в XML позволяет легче визуализировать структуру интерфейса и облегчает отладку.

XML — это язык свободного описания структур документов. То есть, если необходимо, чтобы в документе присутствовал какой-либо элемент, то для него определяется некоторый тег (маркер в тексте). Например, для описания элемента «текстовая строка» можно условиться использовать тег <string>, где первая метка указывает начало описания элемента, а вторая (со знаком /) — конец описания. Между парой тегов помещается текстовое представления содержимого элемента. Для каждого элемента применяется своя пара тегов, при этом однотипные элементы описываются одинаковой парой тегов. Таким образом, для описания двух строк нужны две пары тегов (Рисунок 25).



Рисунок 25 – Создание двух строковых XML-элементов

В открывающем теге можно поместить атрибуты описываемого элемента, такие как цвет, размер, начертание, выравнивание и т. п., то есть описать особенности формируемого элемента. Атрибут — это свойство описываемого элемента. При этом у однотипных элементов полный набор атрибутов будет совпадать, но в описании можно использовать не все свойства. Каждому имени атрибута присваивается значение, записанное в виде текстовой строки, то есть заключенное в двойные кавычки. Разделяются свойства пробелом либо переносом строки. Пример атрибутов представлен на Рисунок 26.



Рисунок 26 – Пример использования XML атрибутов

Данная разметка описывает текстовую строку, написанную красным шрифтом (начертание и размер установлены по умолчанию, поскольку эти свойства не указаны при описании) с выравниванием в центре страницы.

Существуют правила форматирования XML-документа.

* Все теги парные. Это значит, что у каждого открывающего тега обязательно должен присутствовать закрывающий тег. Это правило позволяет описывать вложенные элементы, то есть помещать внутри элемента другие. Если тело тега пусто, то два тега записываются в один, который завершается косой чертой;
* Документ может содержать декларацию – строку заголовка, в которой указывается версия языка и используемая текстовая кодировка;
* Имена тегов могут содержать буквы, цифры и специальные знаки, такие как знак подчеркивания (\_), но должны начинаться с буквы. Теги записываются с соблюдением регистра, поскольку XML регистрозависим;
* Если возникает необходимость использования одинаковых имен элементов для разного типа структур документа, применяют понятие пространства имен. Чтобы различать такие элементы, необходимо задать соответствие — специальный уникальный идентификатор ресурса или URI с конкретным именем элемента. В качестве идентификатора чаще всего используется адрес своего (необязательно реально существующий) ресурса. Пространство имен определяется благодаря атрибуту xmlns в начальном теге элемента;
* В XML-тексте комментарии выделяются тегами <! >;

Элемент – это структурная единица XML-документа. Границы элементов маркируются одинаковыми начальным и конечным тегами. Внутри этой границы может быть текстовая строка значения элемента. Элемент может быть также представлен пустым тегом, то есть не включающим в себя другие элементы и/или символьные данные.

Помимо текстового значения элемент может включать другие элементы. Такие элементы называются дочерними (child) элементами. Дочерних элементов может быть несколько. Элемент, который окружает дочерний элемент, называется родительским (parent). У дочернего элемента может быть только один родительский. Важно, чтобы любой дочерний элемент располагался целиком внутри родительского. То есть пары открывающих и закрывающих тегов всех дочерних элементов должны быть заключены (окружены) парой открывающего и закрывающего тегов родительского элемента. В случае нарушения этого правила любая программа не сможет прочитать ваш документ и выдаст сообщение об ошибочности. Автор документа, вкладывая одни элементы в другие, задает иерархическую структуру внутри документа.

## Ресурсы в Android

Создавая приложение для Android, помимо написания программ на языке Java необходимо также работать с ресурсами. В экосистеме Android принято отделять такие файлы, как изображения, музыка, анимации, стили, макеты окон, строковые константы — в общем все части оформления GUI (Graphical User Interface — графический интерфейс пользователя) от программного кода. Большая часть ресурсов (за исключением мультимедийных) хранятся во внешних XML-файлах. При создании и развитии программного проекта внешние ресурсы легче поддерживать, обновлять и редактировать.

Каждое приложение на Android содержит каталог для ресурсов res/. Доступ к информации в каталоге ресурсов из приложения осуществляется через класс R, который автоматически генерируется средой разработки.

В общем случае ресурсы представляют собой файл (например, изображение) или значение (например, заголовок программы), связанные с создаваемым приложением по имени ресурса. Удобство использования ресурсов заключается в том, что их можно заменять/изменять без изменения программного кода приложения или компиляции. Поскольку имена файлов для ресурсов фактически будут использованы как имена констант в R, то они должны удовлетворять правилам написания имен переменных в Java. Так как разработка ведется на различных ОС (Windows, Mac, Linux), то также есть еще ограничения. В итоге имена файлов должны состоять исключительно из букв в нижнем регистре, чисел и символов подчеркивания.

В Android используются два подхода к процессу создания ресурсов — первый подход заключается в том, что ресурсы задаются внутри файла и тогда его имя задается в месте его описания. Второй подход — ресурс задается в виде самого файла, и тогда имя файла уже и есть имя ресурса.

Для различных типов ресурсов, определенных в проекте, в каталоге res создаются подкаталоги. Поддерживаемые подкаталоги:

* animator/: xml-файлы, определяющие анимацию свойств;
* anim/: xml-файлы, определяющие tween-анимацию;
* color/: xml-файлы, определяющие список цветов;
* drawable/: Графические файлы (.png, .jpg, .gif);
* dimensions/: xml-файлы, определяющие размерности элементов;
* mipmap/: Графические файлы, используемые для иконок приложения под различные разрешения экранов;
* layout/: xml-файлы, определяющие пользовательский интерфейс приложения
* menu/: xml-файлы, определяющие меню приложения;
* raw/: различные файлы, которые сохраняются в исходном виде;
* Каталог values/ является одной из ключевых частей проекта Android и предназначен для хранения XML-файлов с различными ресурсами, которые используются в приложении. Эти ресурсы позволяют организовать и централизовать данные, что делает код более гибким, удобным в сопровождении и адаптируемым для разных конфигураций устройств, локализаций и тем оформления. Каталог values/ играет важную роль в структурировании данных внутри Android-приложения. Использование XML-файлов из этого каталога позволяет избежать «жесткого» кодирования строк, цветов, размеров и других параметров, что делает проект более гибким, удобным для локализации и адаптации к различным устройствам;
* xml/: Произвольные xml-файлы;
* font/: файлы с определениями шрифтом и расширениями .ttf, .otf или .ttc, либо файлы XML, который содержат элемент.

Чаще других используют следующие ресурсы: разметка (layout), строки (string), цвета (color) и графические рисунки (bitmap, drawable) (Рисунок 27).

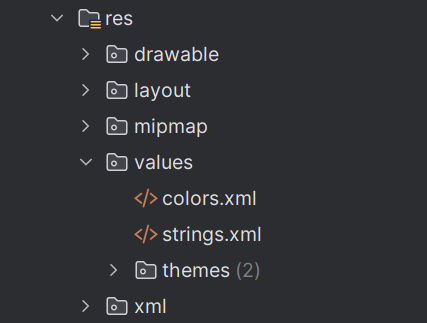


Рисунок 27 – Структура ресурсов в Android Studio

## Создание интерфейса пользователя с помощью ресурса

Графический интерфейс создается с помощью представлений (View) и групп представлений (ViewGroup). Эти элементы размещаются на активности, их описания помещаются в файл манифеста, а действия с объектами прописываются программно в файле кода MainActivity.java в виде методов классов, наследуемых от классов View и ViewGroup или атрибутивно в файле разметки layout/activity\_main.xml. У файла разметки также имеется графический вид Graphical layout — системная имитация мобильного устройства.

В файле определяются все графические элементы и их атрибуты, которые составляют интерфейс. При создании разметки в XML следует соблюдать некоторые правила: каждый файл разметки должен содержать один корневой элемент, который должен представлять объект View или ViewGroup.

По умолчанию при создании проекта с пустой activity уже есть один файл ресурсов разметки activity\_main.xml. В нем корневым элементом является элемент ConstraintLayout, который содержит элемент TextView. Как правило, корневой элемент содержит определение используемых пространств имен XML (Рисунок 22).

В XML-разметке ресурсы подключаются через специальные ссылки вида @type/name. Например, в файле разметки можно задать текст кнопки с помощью строки из файла strings.xml (Рисунок 28).

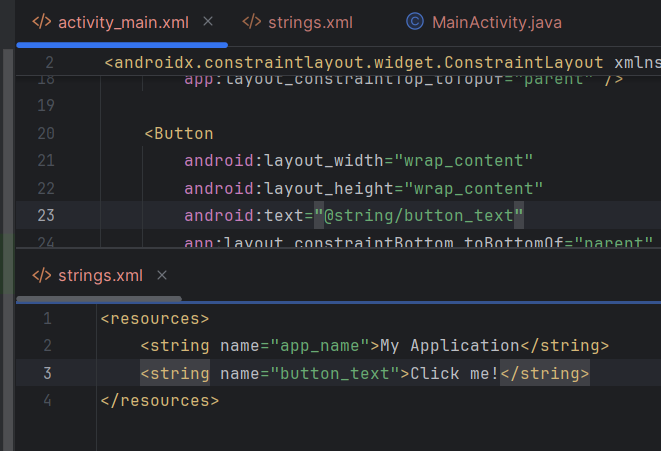


Рисунок 28 – Подключение строкового ресурса в XML-разметке

## Компоненты разметки

Компоненты разметки имеют конкретный внешний вид и конкретные задачи. В SDK Android находятся множество различных компонентов, однако будет рассмотрен основной перечень из них.

### TextView

TextView – компонент, предназначенный для простого вывода текста на экран. Он просто отображает текст без возможности его редактирования (Рисунок 29).

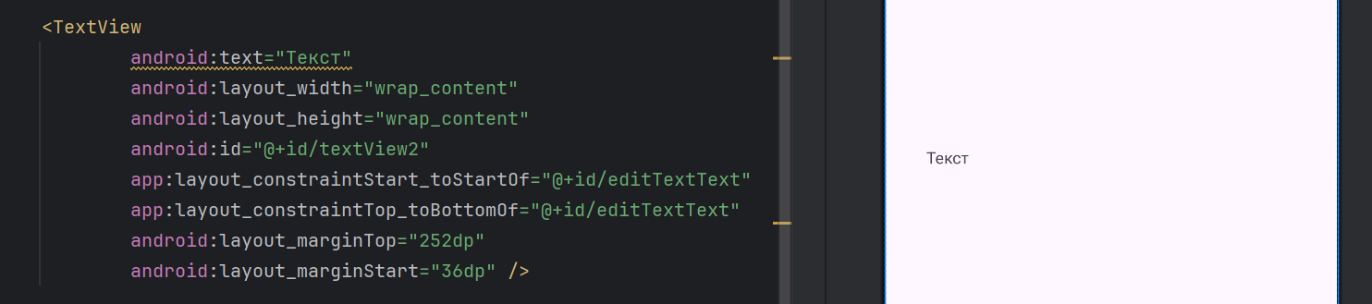


Рисунок 29 – Пример компонента TextView

Основные атрибуты компонента TextView:

* android:text – устанавливает текст, который будет отображаться,
* android:textSize – размер текста,
* android:textColor – цвет текста,
* android:textStyle – стиль (например, bold, italic),
* android:gravity – выравнивание текста внутри компонента.

### EditText

EditText – является подклассом класса TextView. Он также представляет текстовое поле, но теперь уже с возможностью ввода и редактирования текста. Таким образом, в EditText мы можем использовать все те же возможности, что и в TextView (Рисунок 30).



Рисунок 30 – Пример компонента EditView

Основные атрибуты компонента EditText:

* android:hint – текст-подсказка внутри поля,
* android:inputType – тип вводимых данных (например, text, number, password),
* android:maxLength – ограничение на количество символов,
* android:singleLine – ввод только в одну строку.
* android:gravity – выравнивание текста внутри компонента.

### Button

Button – компонент, представляющий из себя кнопку, на которую может нажать пользователь и которая может содержать текст (Рисунок 31).

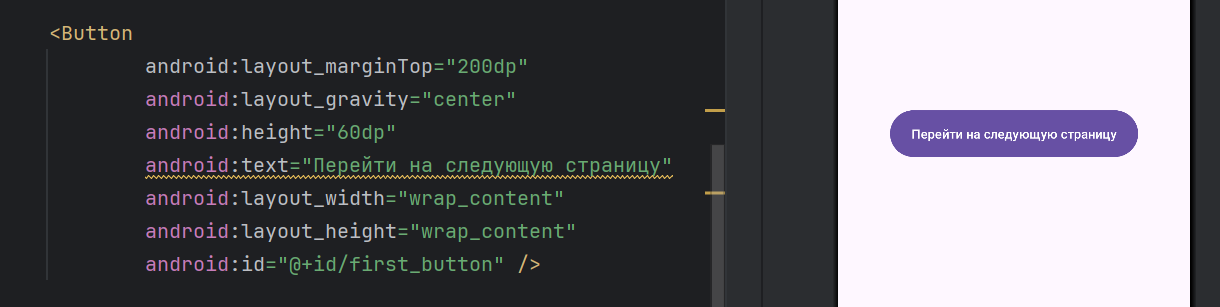


Рисунок 31 – Пример компонента Button

Основные атрибуты компонента Button:

* android:text – текст на кнопке,
* android:onClick – метод, который будет вызван при нажатии,
* android:background – фоновое изображение или цвет,
* android:drawableLeft / android:drawableRight – добавление иконок слева/справа от текста,
* android:enabled – доступность кнопки (true / false).

### ImageView

ImageView – является базовым элементом-контейнером для отображения графики. Можно загружать изображения из разных источников, например, из ресурсов программы или интернета.

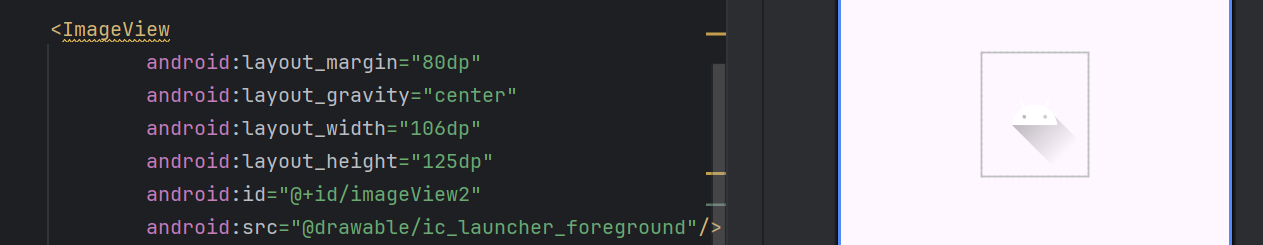


Рисунок 32 – Пример компонента ImageView

Основные атрибуты компонента ImageView:

* android:src – источник изображения (@drawable/... или URL);
* android:scaleType – способ масштабирования (fitCenter, centerCrop и т.д.);
* android:adjustViewBounds – сохранение соотношения сторон изображения;
* android:contentDescription – описание для доступности (например, для скринридеров).

# ПРАКТИЧЕСКОЕ ЗАДАНИЕ

## Создание проекта

Сначала с официального сайта была скачана и установлена среда разработки Android Studio версии 2024.2.2.14 и Android SDK Platform 35. Далее для создания проекта был выбран шаблон “Empty Views Activity” из категории шаблонов “Phone and Tablet”, поскольку его хватает для выполнения поставленной задачи (Рисунки Рисунок 4-Рисунок 5).

После выбора шаблона были установлены настройки проекта (Рисунок 6). В качестве названия проекта было выбрано “KomisarikMA” (мое ФИО). Языком программирования был выбран Java. В качестве минимального SDK был оставлен API 24 ("Nougat"; Android 7.0), который выбран по умолчанию, при такой настройке, приложение будет запускаться примерно на 97,4% процентах устройств (Рисунок 6).

## Разметка

### Настройка корневых элементов

После создания и настройки проекта в папке ресурсов присутствует файл разметки main\_activity.xml (Рисунок 23), в котором находится элемент TextView. Удалим этот элемент (Рисунок 33).

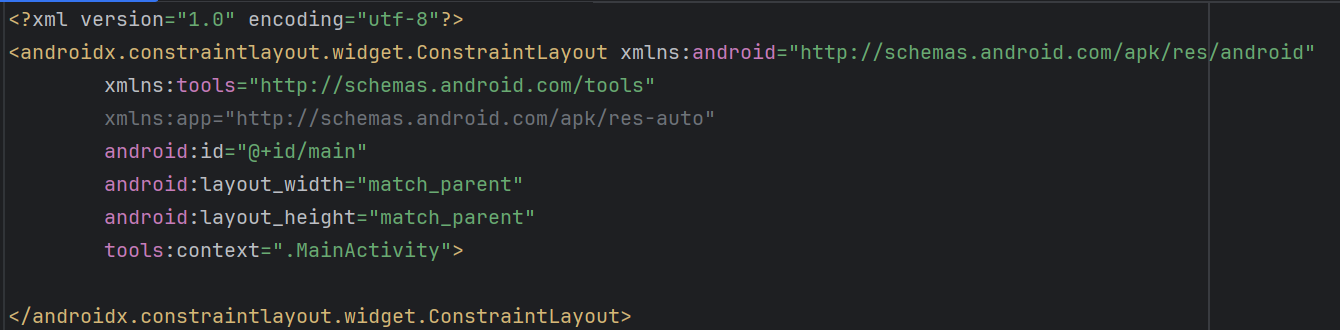


Рисунок 33 – Пустой файл разметки

После этого заменим тип лейаута на LinearLayout. Для этого заменим корневой тег на android.widget.LinearLayout. Также установим атрибут **orientation** в “vertical”, чтобы поменять ориентацию лейаута с горизонтальной на вертикальную атрибут (Рисунок 34).

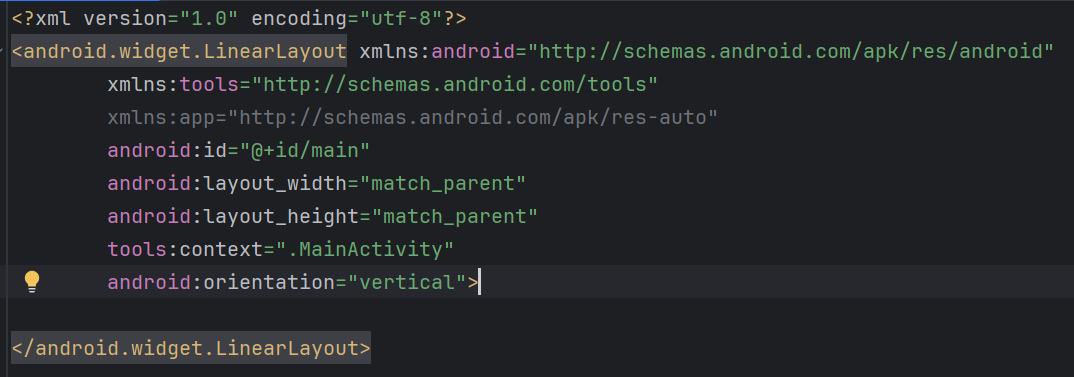


Рисунок 34 – Смена лейаута первой активити на LinearLayout

Далее создадим еще один активити по шаблону Empty Views Activity (Рисунок 35).

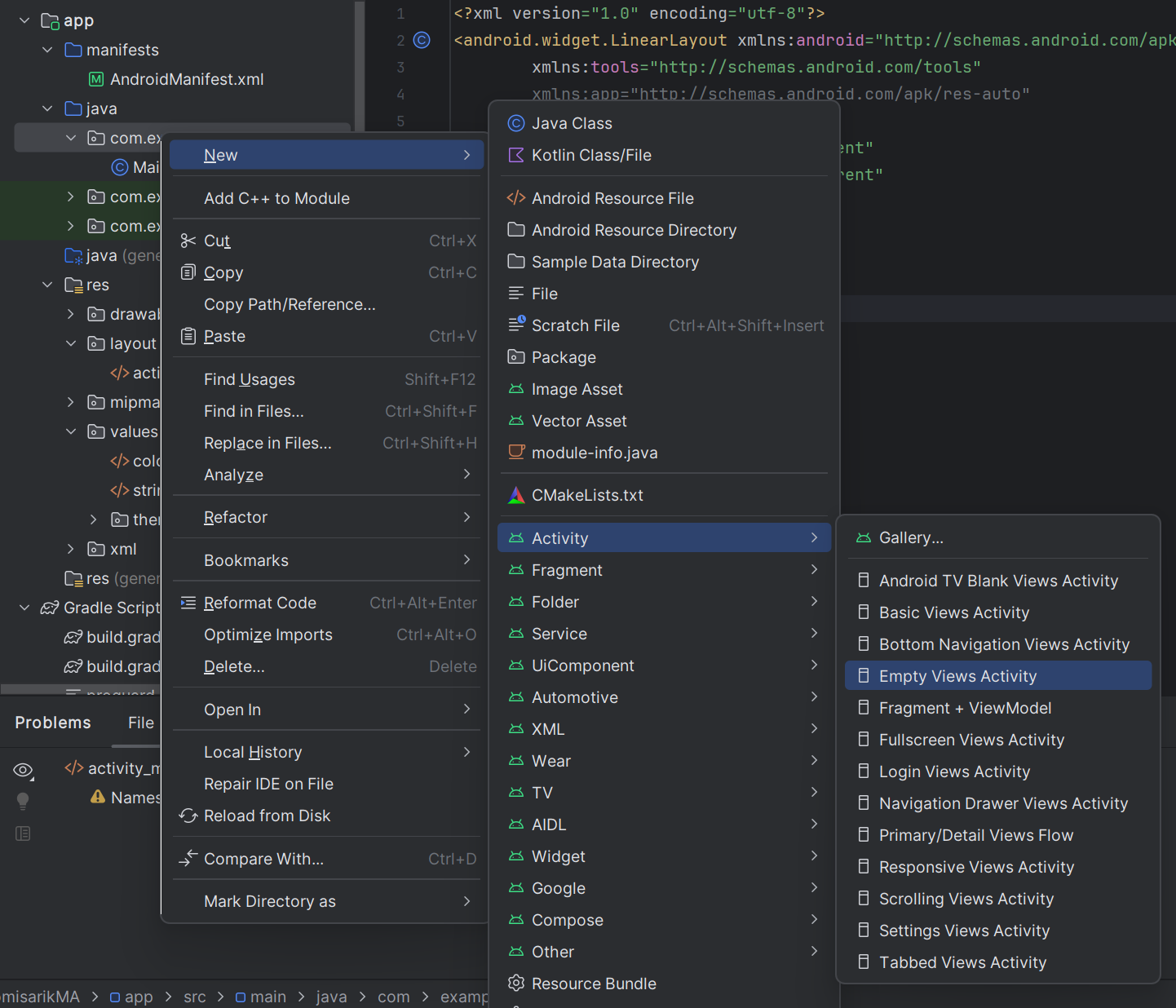


Рисунок 35 – Создание второй активити

В настройках выбираем название, а также оставим галочку “Generate a Layout File”, чтобы вместе с активити создался и его файл разметки (Рисунок 36).

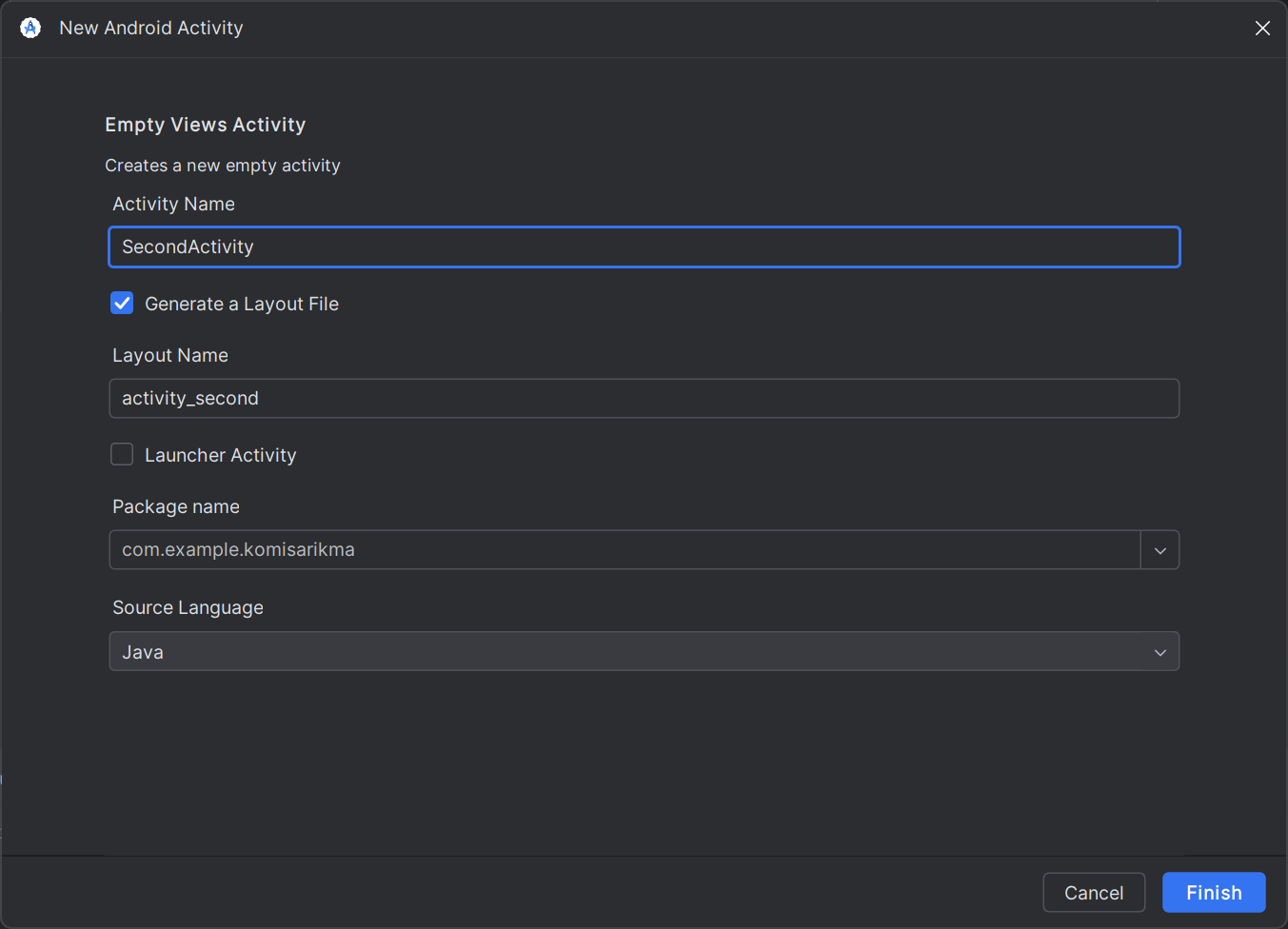


Рисунок 36 – Выбор настроек второй активити

В результате был создан новый файл разметки activity\_second.xml и файл активити SecondActivity.java (Рисунок 37).

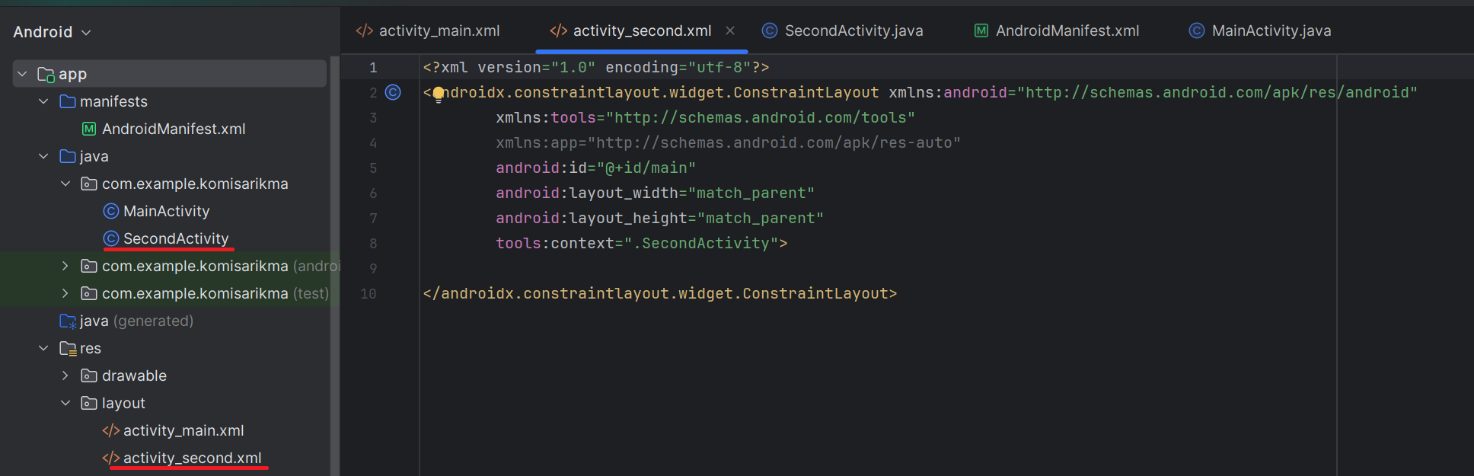


Рисунок 37 – Файл разметки activity\_second.xml

Аналогично файлу разметки activity\_main.xml, заменим лейаут на LinearLayout и сменим ориентацию на вертикальную (Рисунок 38).

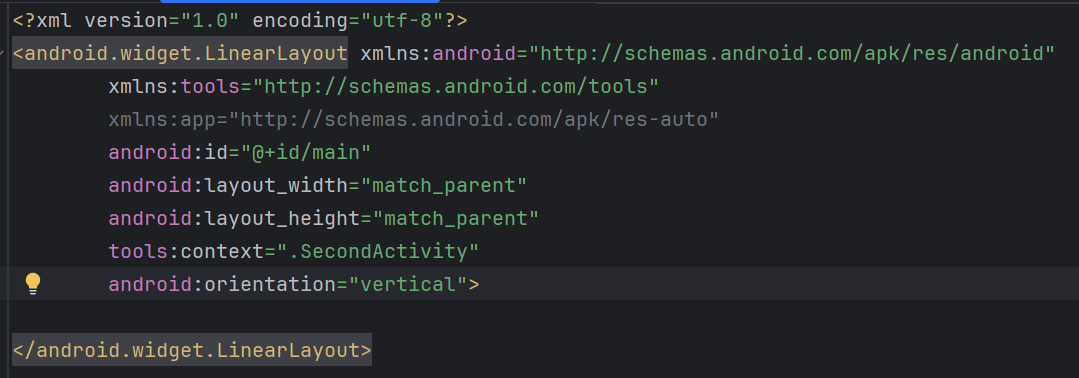


Рисунок 38 – Смена лейаута второй активити на LinearLayout

### TextView

В файле разметки activity\_main.xml создадим элемент TextView со следующими атрибутами:

* layout\_gravity=”center”: размещает элемент в середине отведенного ему пространства внутри родительского компонента;
* layout\_height=”wrap\_content”: высота элемента устанавливается в соответствием с его содержимым;
* layout\_margin=”40dp”: устанавливает отступы со всех сторон от элемента, равными 40dp. Dp или dip (density independent pixel) – это величина, схожая с пикселем (px), но не зависящая от плотности пикселей на экране;
* layout\_width=”wrap\_content”: устанавливает ширину элемента в соответствием с его содержимым;
* text=”Текст”: устанавливает текст поля в «Текст».

XML код и отображение элемента представлены на рисунке Рисунок 39.



Рисунок 39 – XML код и отображение созданного элемента TextView первого активити

В файле разметки activity\_second.xml создадим элемент TextView и зададим ему те же самые параметры (Рисунок 40).

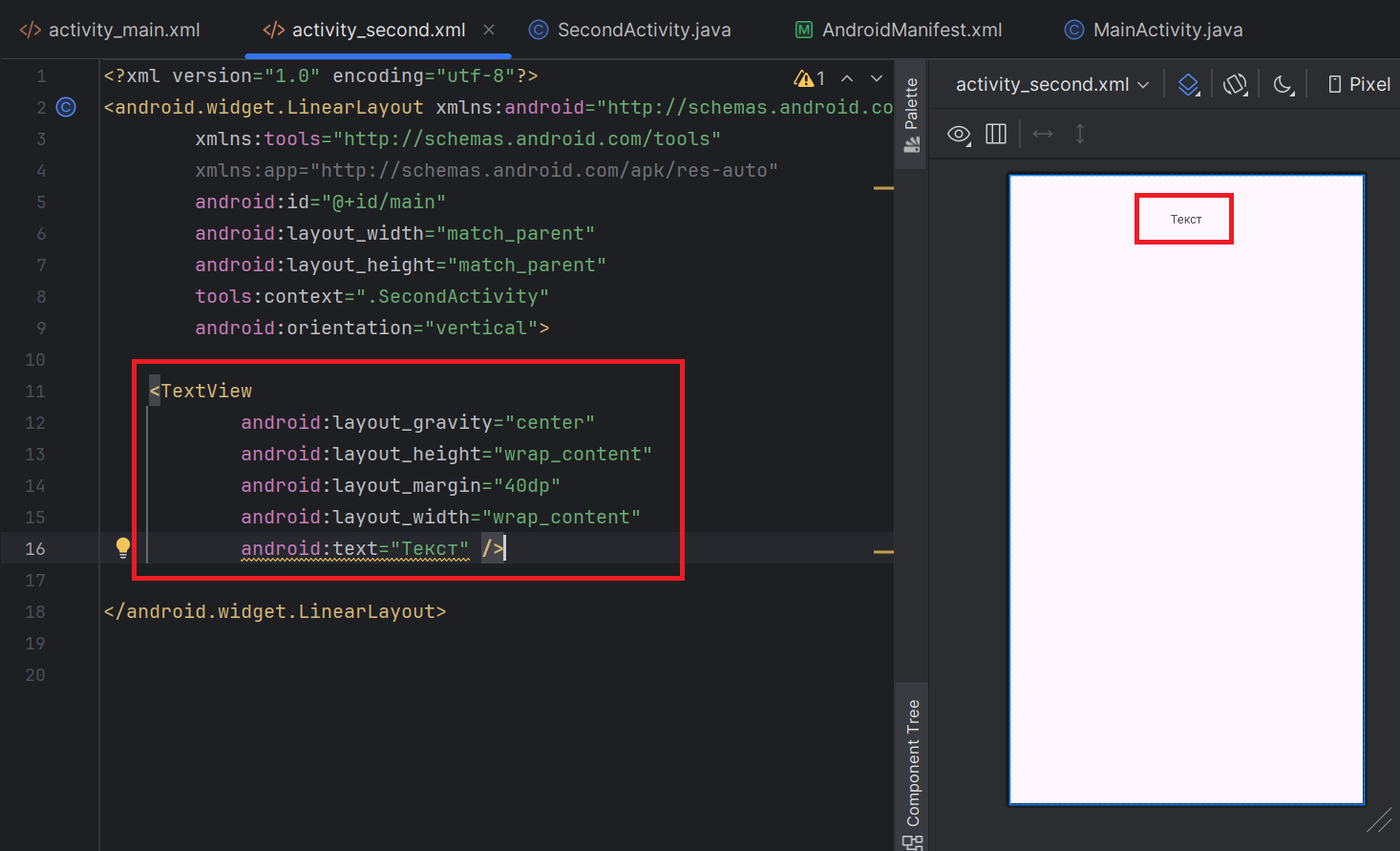


Рисунок 40 – XML код и отображение созданного элемента TextView второго активити

### EditText

В файле разметки activity\_main.xml создадим элемент EditText со следующими атрибутами:

* ems=”10”: устанавливает ширину поля для ввода текста 10ems. Ems – единица измерения, равная ширине буквы ‘M’ в выбранном шрифте;
* hint=”Введите текст”: отображает текст-подсказку “Введите текст”, если пользователь не вводил текст;
* input\_type=”text”: задает тип вводимых данных как обычный текст;
* layout\_gravity=”center”: размещает элемент в середине отведенного ему пространства внутри родительского компонента;
* layout\_height=”48dp”: устанавливает высоту элемента в 48dp;
* layout\_marginTop=”30dp”: устанавливает отступ сверху от элемента в 40dp;
* layout\_width=”wrap\_content”: устанавливает ширину элемента в соответствием с его содержимым;
* text\_alignment=”center”: выравнивает текст элемента по центру.

XML код и отображение элемента представлены на рисунке Рисунок 41.

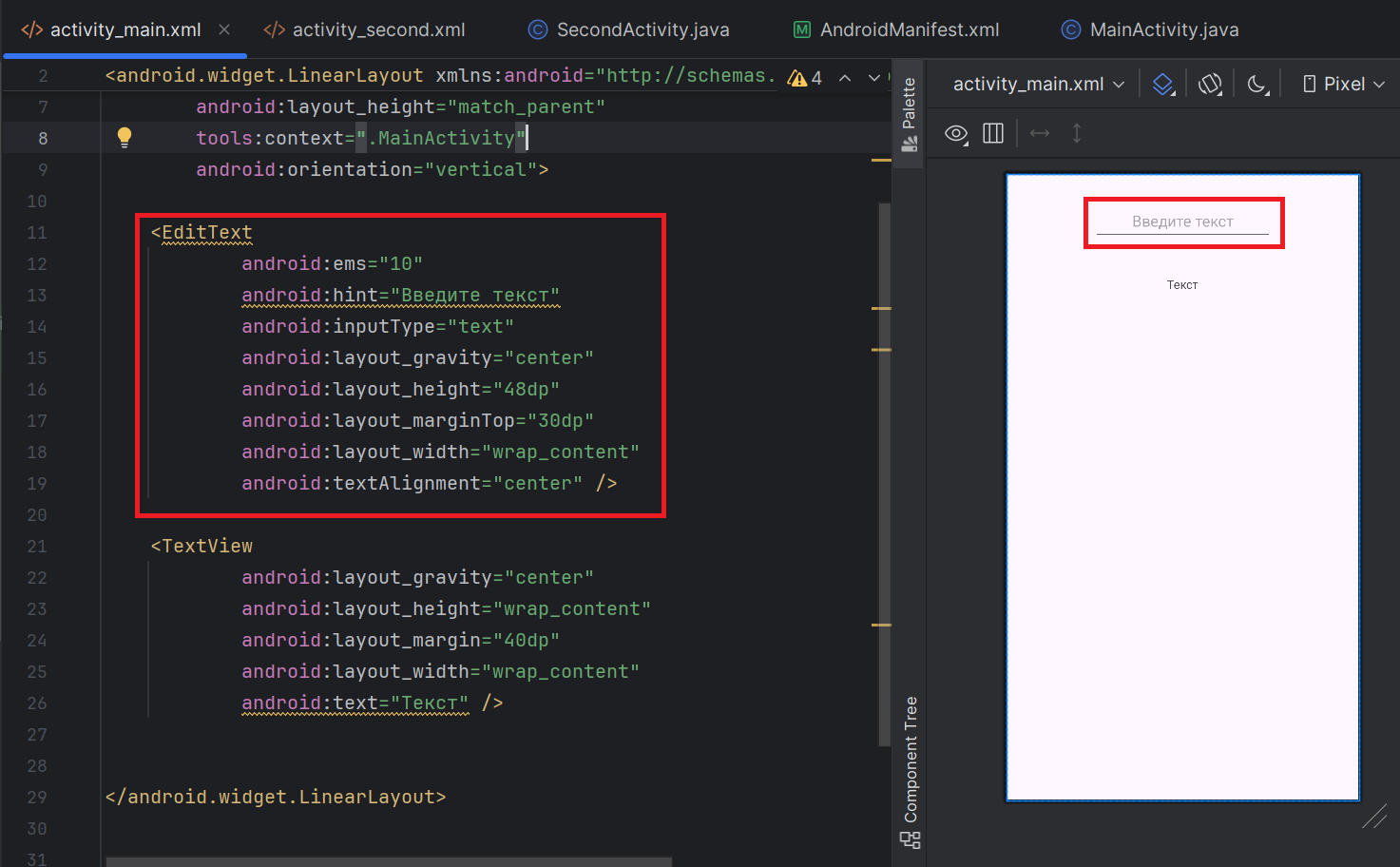


Рисунок 41 – XML код и отображение созданного элемента EditText

### Button

В файле разметки activity\_main.xml создадим элемент Button со следующими атрибутами:

* layout\_gravity=”center”: размещает элемент в середине отведенного ему пространства внутри родительского компонента;
* layout\_height=”80dp”: устанавливает высоту элемента в 80dp;
* layout\_marginTop=”100dp”: устанавливает отступ сверху от элемента в 100dp;
* layout\_width=”160dp”: устанавливает ширину элемента в 160dp;
* text=”Кнопка”: устанавливает текст поля в «Кнопка»;
* textSize=”20sp”: устанавливает размер шрифта текста в 20sp. Sp (scalable pixel) – то же самое, что и dp, но меняет свой размер в зависимости от предпочитаемого размера текста пользователя.

XML код и отображение элемента представлены на рисунке Рисунок 42.

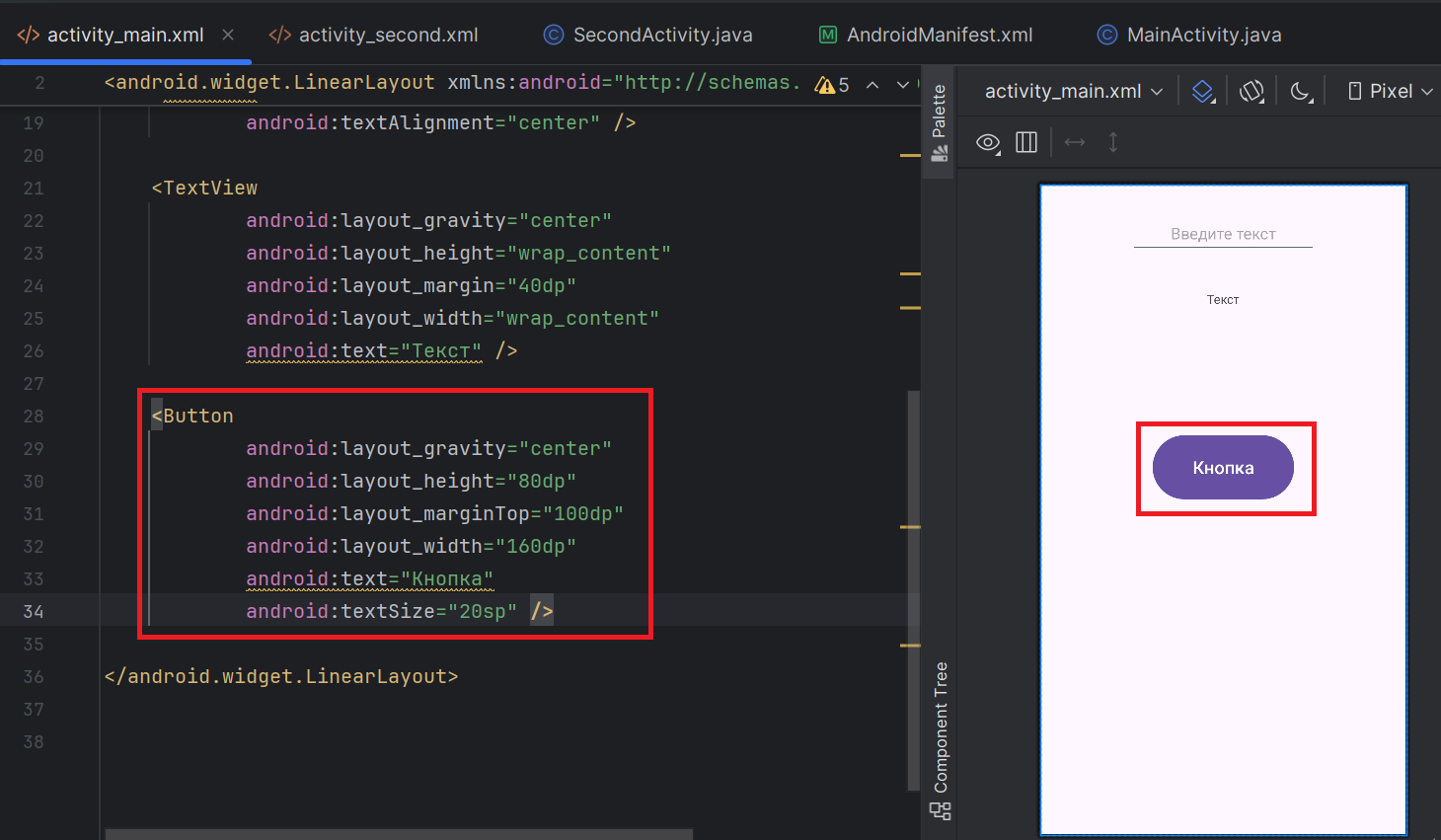


Рисунок 42 – XML код и отображение созданного элемента Button первого активити

В файле разметки activity\_second.xml создадим элемент Button со следующими атрибутами:

* layout\_gravity=”center”: размещает элемент в середине отведенного ему пространства внутри родительского компонента;
* layout\_height=”wrap\_content”: устанавливает высоту элемента в соответствием с его содержимым;
* layout\_margin=”50dp”: устанавливает все отступы от элемента в 50dp;
* layout\_width=”wrap\_content”: устанавливает ширину элемента в соответствием с его содержимым;
* text=”Кнопка”: устанавливает текст поля в «Кнопка».

XML код и отображение элемента представлены на рисунке Рисунок 43.

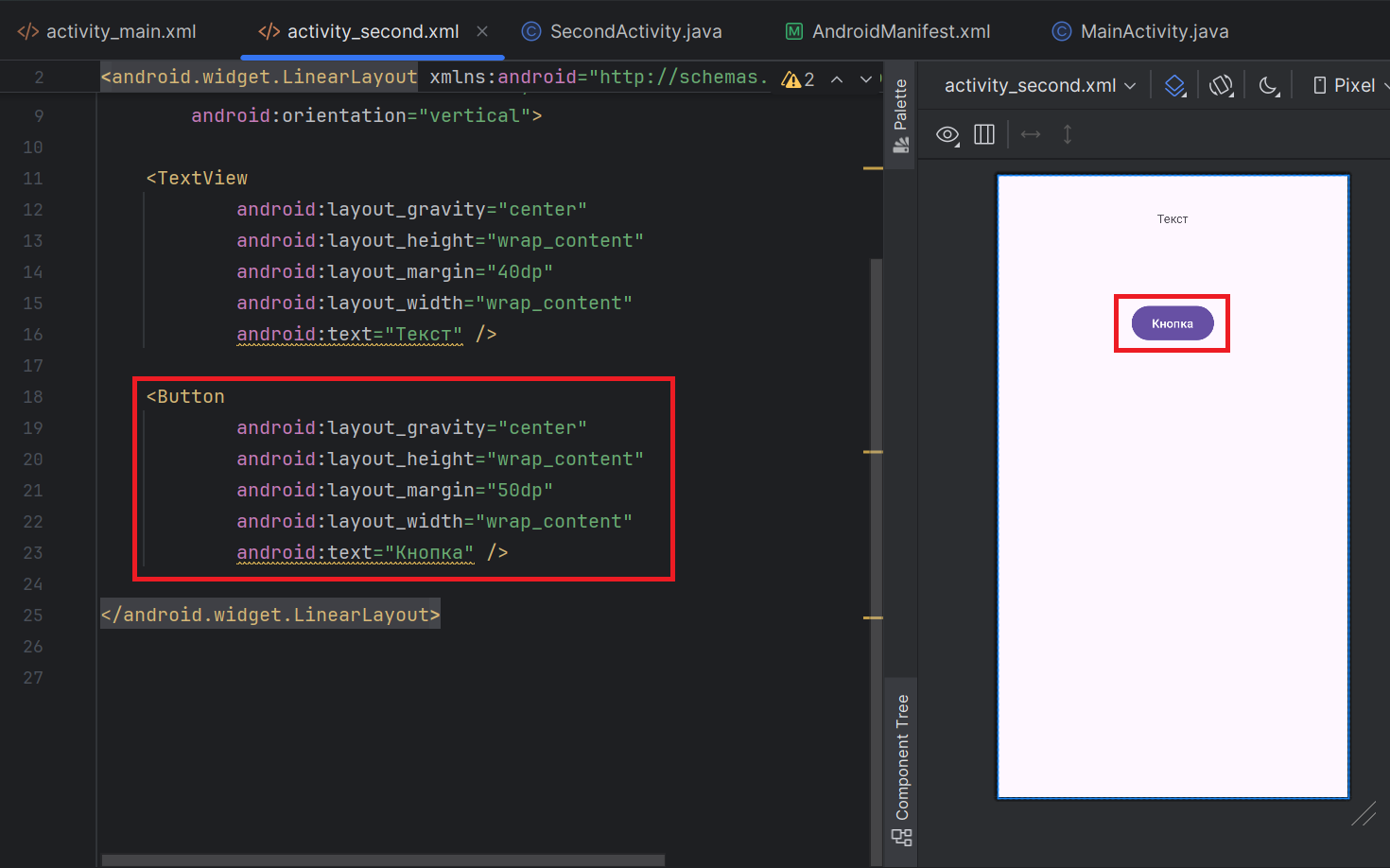


Рисунок 43 – XML код и отображение созданного элемента Button второго активити

### ImageView

В файле разметки activity\_main.xml создадим элемент ImageView со следующими атрибутами:

* layout\_gravity=”center”: размещает элемент в середине отведенного ему пространства внутри родительского компонента;
* layout\_height=”125dp”: устанавливает высоту элемента в 125dp;
* layout\_margin=”80dp”: устанавливает все отступы от элемента в 80dp;
* layout\_width=”106dp”: устанавливает ширину элемента в 106dp;
* src=”@drawable/ic\_launcher\_foreground”: устанавливает изображение из ресурса drawable/ic\_launcher\_foreground.xml.

XML код и отображение элемента представлены на рисунке Рисунок 44.

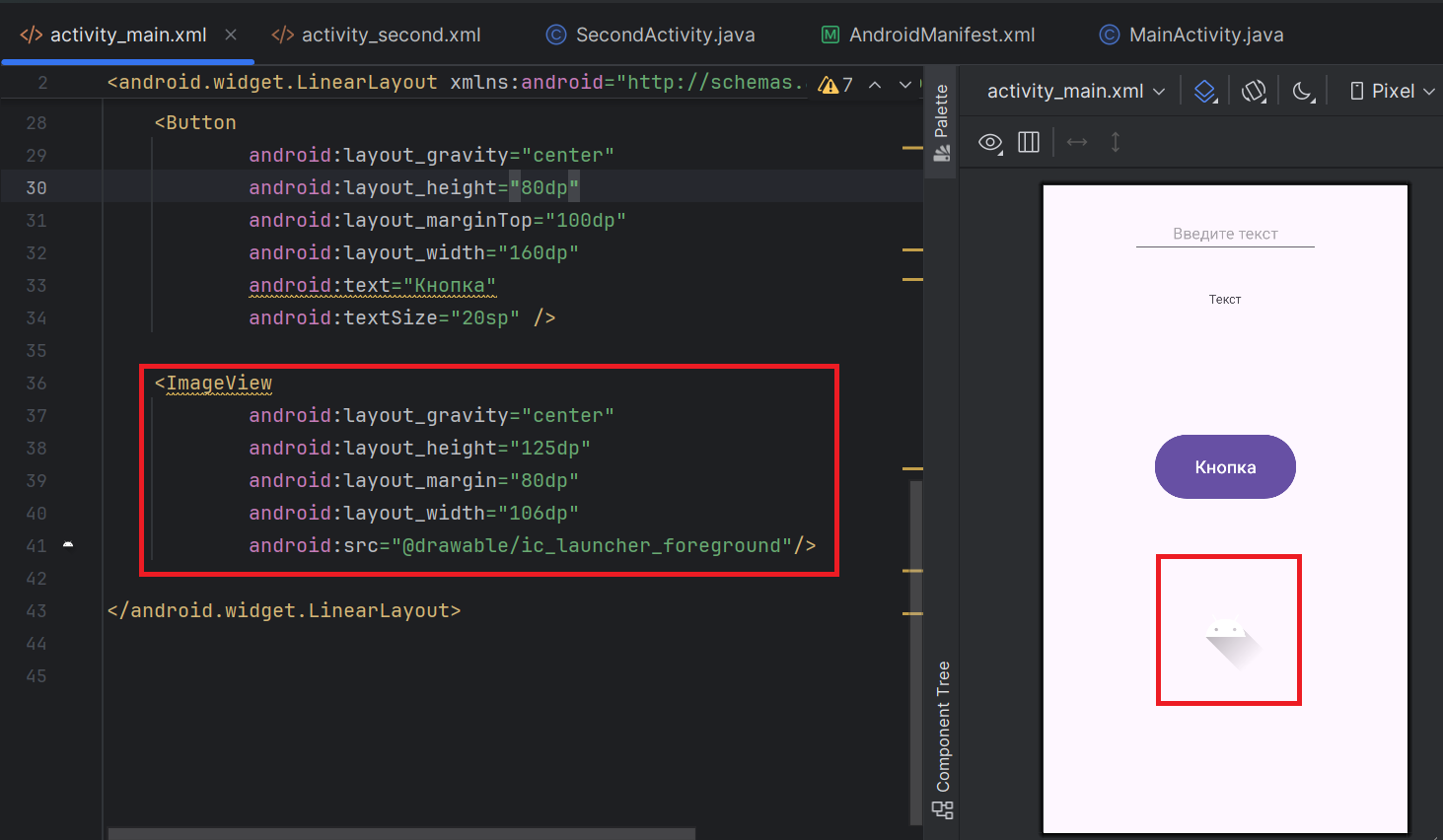


Рисунок 44 – XML код и отображение созданного элемента ImageView

## Результат

После проведенных действий были созданы файлы разметки activity\_main.xml и activity\_second.xml (Рисунки Рисунок 45-Рисунок 46).

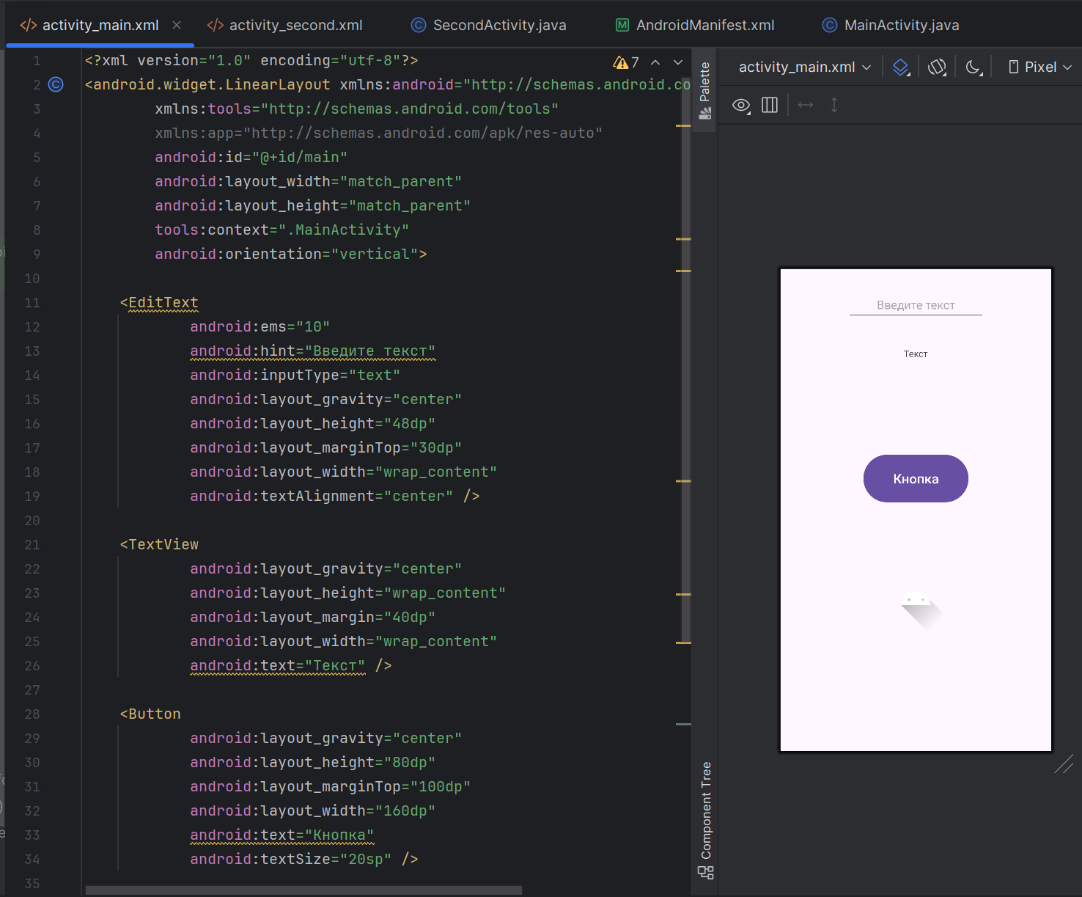


Рисунок 45 – Файл разметки activity\_main.xml

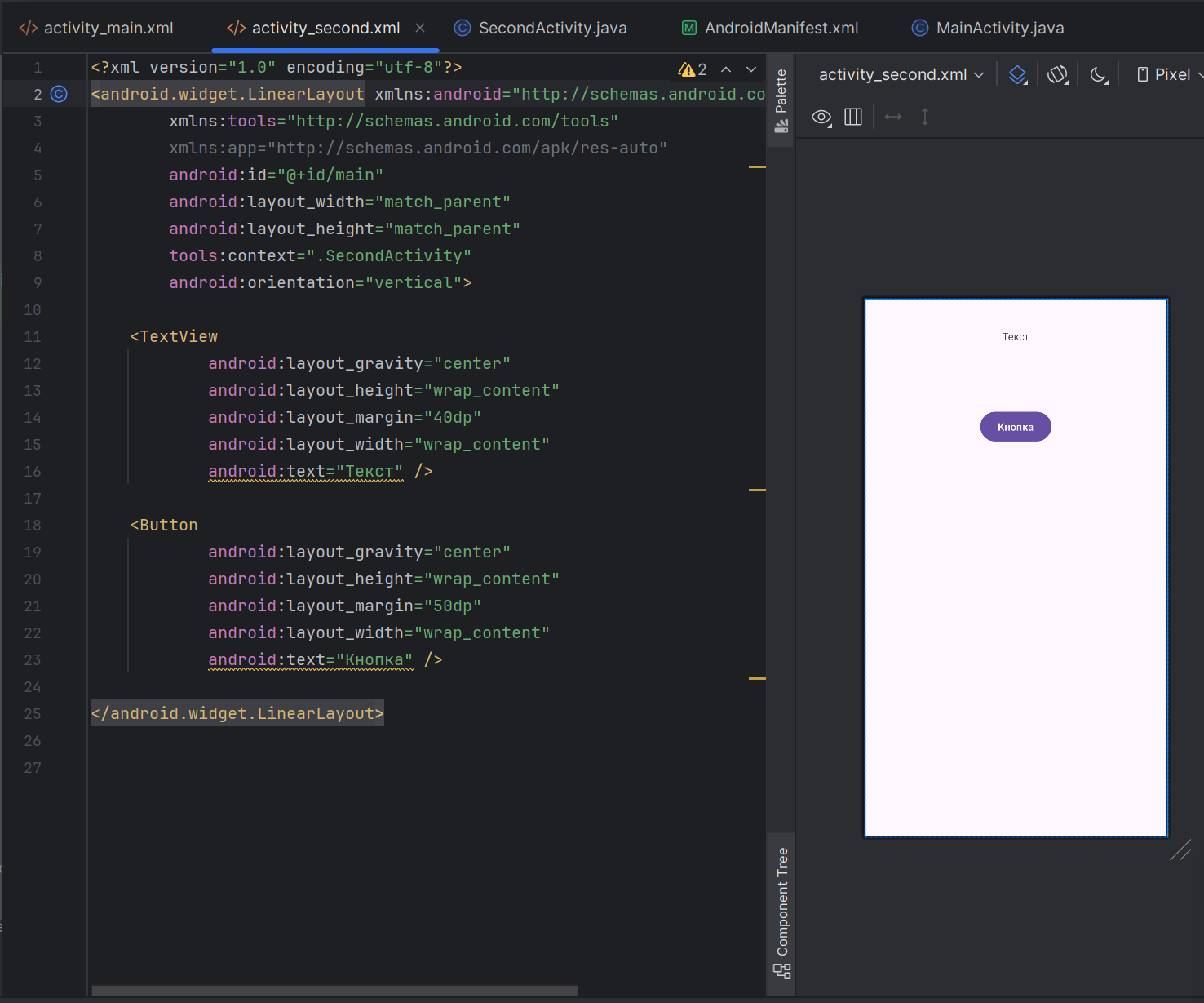


Рисунок 46 – Файл разметки activity\_second.xml

На рисунках Рисунок 47-Рисунок 48 представлено отображение данных файлов разметки на экране физического устройства Xiaomi Redmi Node 9 Pro.

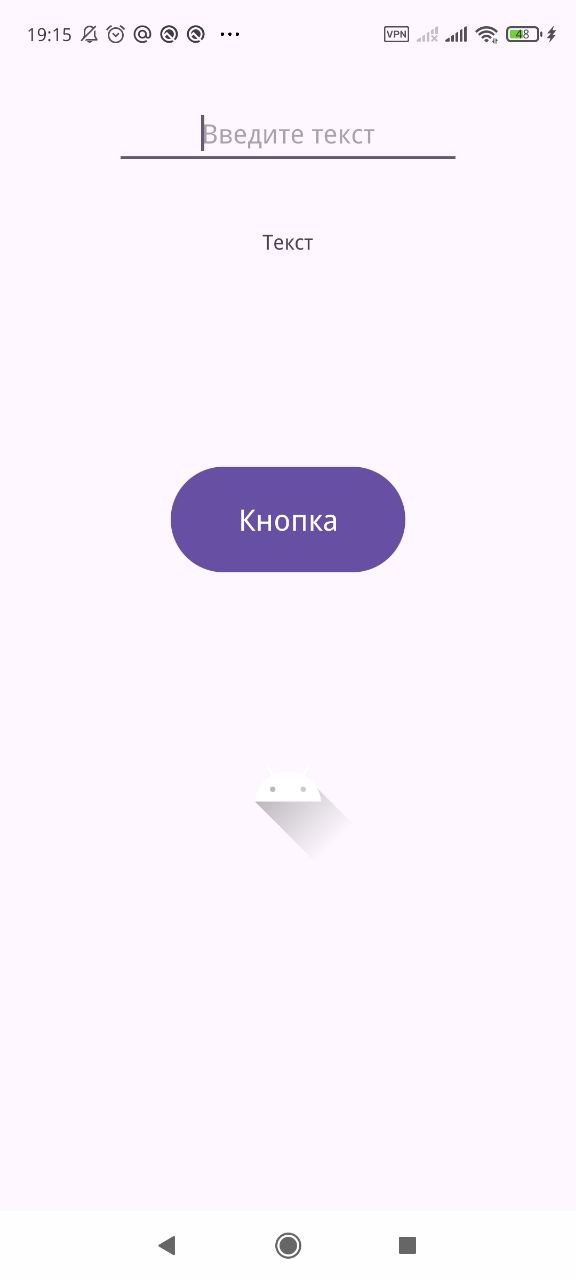


Рисунок 47 – Отображение файла разметки activity\_main.xml на физическом устройстве



Рисунок 48 – Отображение файла разметки activity\_second.xml на физическом устройстве

ЗАКЛЮЧЕНИЕ

В ходе выполнения работы были изучены основы работы с интегрированной средой разработки Android Studio, предназначенной для разработки приложений на устройства с операционной системой Android. С применением наиболее используемых компонентов разметки графического пользовательского интерфейса была проведена верстка двух страниц. Был получен опыт работы с XML-разметкой Android.