**Теоретический часть:**

Android – это операционная система, предназначенная для смартфонов, планшетов, электронных книг, умных часов, цифровых проигрывателей, ноутбуков, очков Google Glass и телевизоров. Android работает на ядре Linux и собственной реализации JVM (виртуальной машины Java) от Google.

Для разработки на платформу Android можно использовать как кроссплатформенные среды разработки такие как Unity, так и специализированные такие как Android Studio.

Android Studio – это интегрированная среда разработки для создания приложений под операционную систему Android. Android Studio основан на программном обеспечении IntelliJ IDEA от компании JetBrains.

Android Studio обычно используют для создания приложений и некоторых простых или игр. Для создания игр лучше подходит среда разработки Unity.

В комплекте с Android Studio устанавливается эмулятор для запуска виртуальных устройств (Android Virtual Device). Android Studio позволяет настраивать для AVG множество параметров, например такие как: разрешение экрана, объем памяти, наличие камер, версия Android и другие. При установке эмулятора устанавливаются набор шаблонов, эмулирующих популярные устройства, работающие на операционной системе Android.

**Практический часть:**

Запустим Android studio



Рис.1. Окно выбора проекта.

Создадим новый проект, нажав на кнопку “Create New Project”

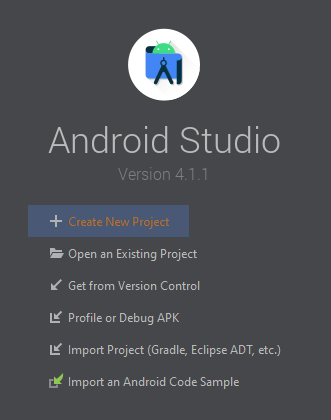


Рис. 2. Кнопка создания нового проекта.

При создании проекта android studio предлагает нам много пресетов для создания проекта. Выберем “No Activity” и продолжим.

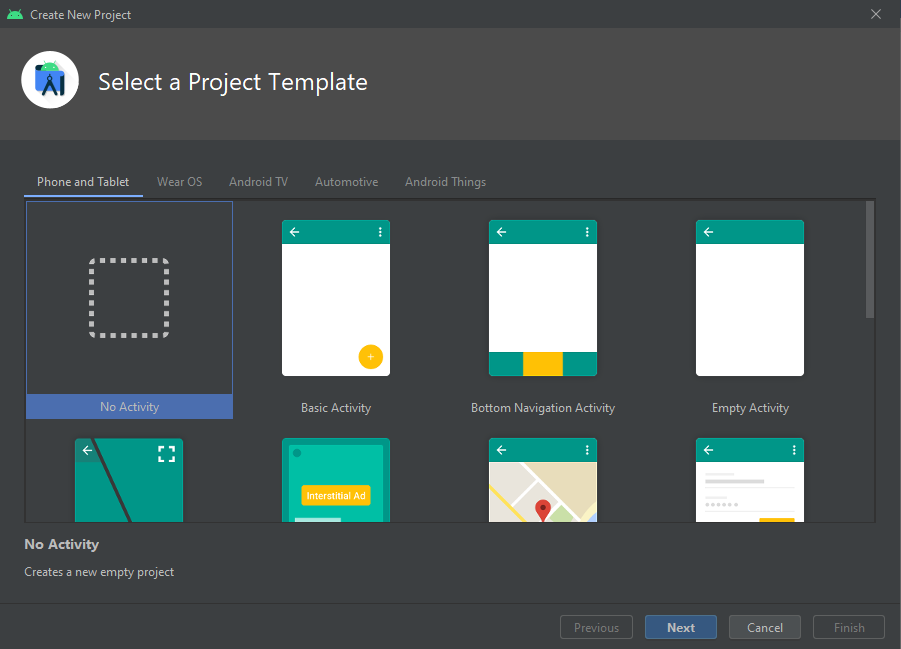


Рис. 3. Окно выбора шаблона приложения.

Настроим название проекта и его расположение, завершим создание проекта.

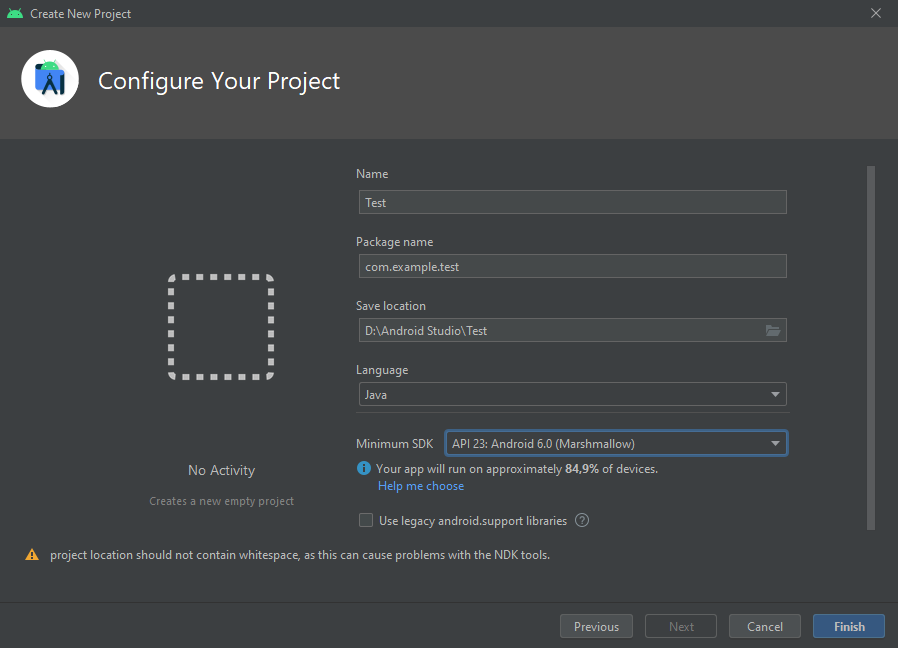


Рис. 4. Окно настройки проекта.

После создания проекта нам откроется окно Android Studio. Рассмотрим его интерфейс:

1. Верхняя навигационная панель



Рис. 5. Верхняя навигационная панель.

В пункте меню “File” можно открыть настройки или создать новый / открыть уже существующий проект.

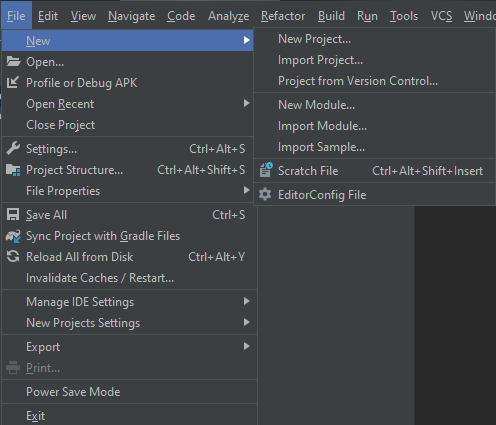


Рис. 6. Кнопка в верхнем меню, для создание нового проекта.

В пункте меню “Tools” можно открыть окно настройки AVG (эмуляторы устройств, работающих на android) и SDK (пакеты и библиотеки).

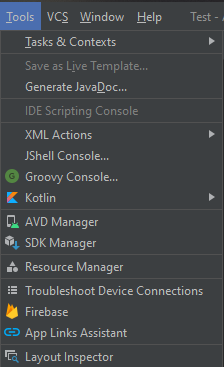


Рис. 7. Раздел меню с пунктами настройки проекта.

В окне настройки SDK можно управлять и скачивать стандартные библиотеки от Google. Такие как различные версии Android, Google Play сервисы, эмулятор android устройств и драйвера для него.

В окне настройки AVG можно управлять и создавать виртуальные устройства, которые будут открываться в эмуляторе.

* Откроем окно настройки AVG

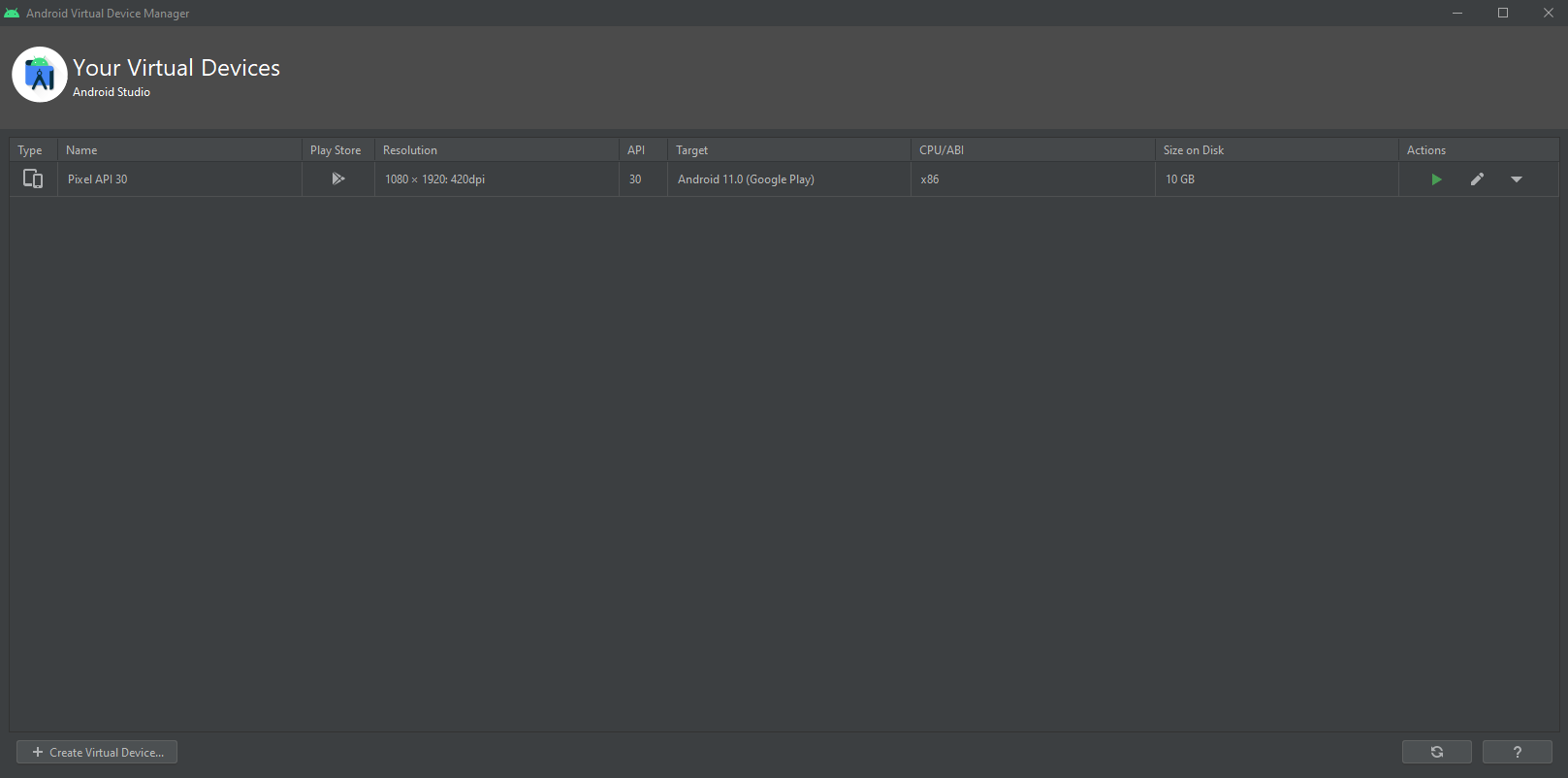


Рис. 8. Окно настроек AVG.

* Если здесь нет устройств, то создадим новое, для этого нажмем кнопку “Create Virtual Device”. Выберем категорию “Phone” и модель Pixel.

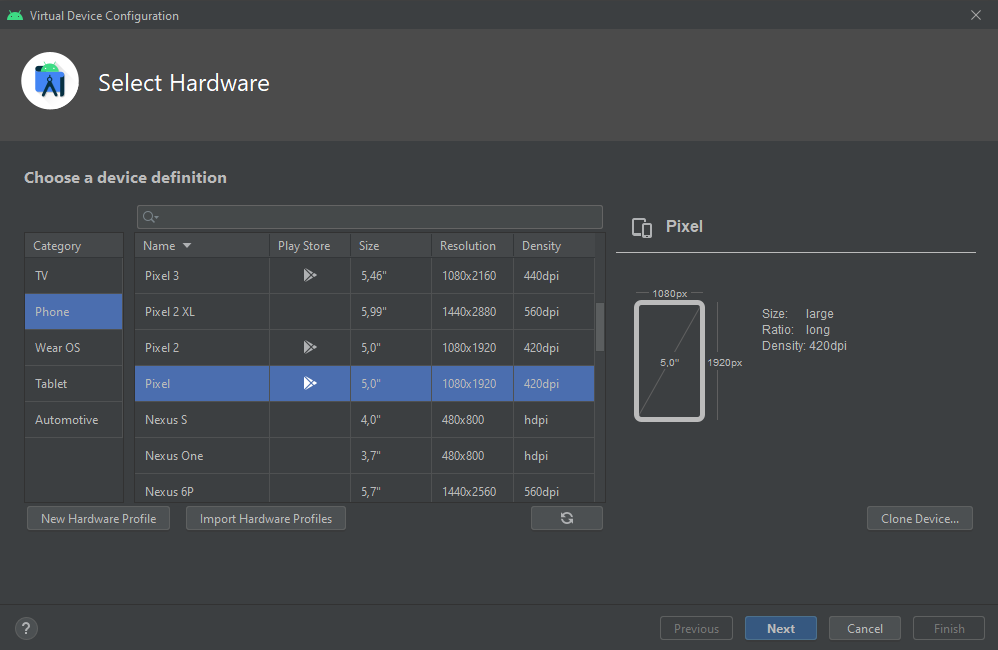


Рис. 9. Выбор устройства.

* Скачаем и выберем последнюю версию Android.

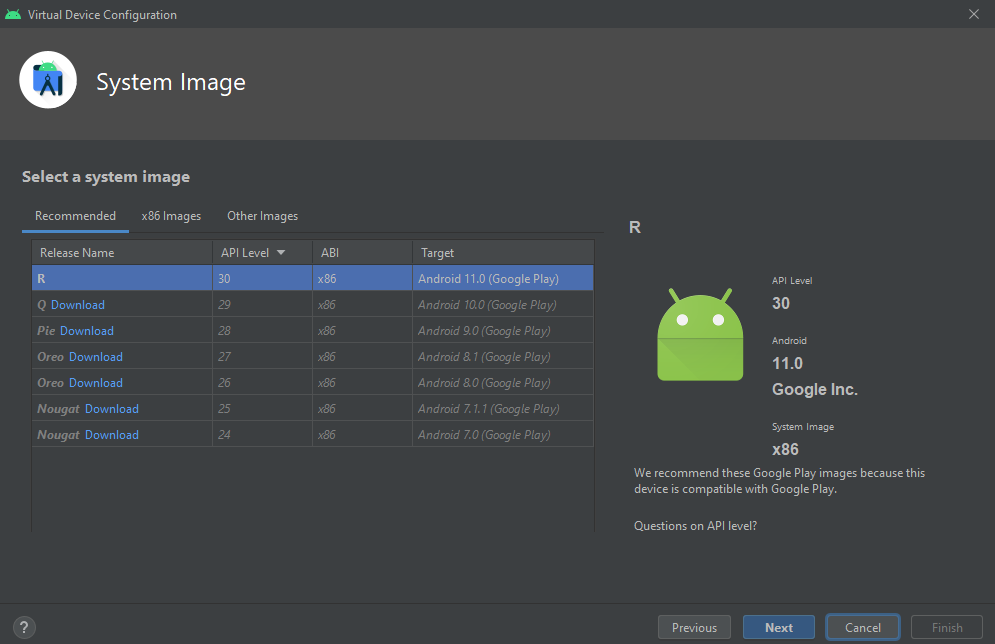


Рис. 10. Выбор версии OS.

* Назовем новое виртуальное устройство и закончим.



Рис. 11. Окно настройки виртуального устройства.

1. Панель управления запуском приложения



Рис. 12. Панель управления запуском приложения.

При помощи данной панели можно запустить приложение на эмуляторе или устройстве, подключенное через usb провод к компьютеру, а также разрешившее режим отладки.

Рассмотрим кнопки на этой панели:

* Иконка молотка – позволяет собрать apk файл проекта.
* Раскрывающийся список с выбранным элементом “Pixel API 30” – позволяет выбрать на каком виртуальном/подключенном устройстве будет запускаться приложение.
* Иконка запуска – собирает, устанавливает и запускает приложение на выбранном устройстве.
* Иконка жука – позволяет запустить приложение в режиме отладки.

1. Окно иерархии проекта

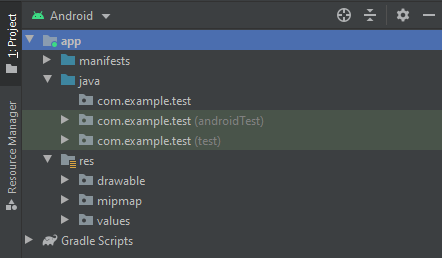


Рис. 13. Окно иерархии проекта

При помощи этого окна можно создавать новые файлы и управлять уже существующими. Изучите разнообразие файлов, которое можно создать, нажав ПКМ по иерархии и в контекстном меню выбрав пункт “new”.

Создадим новый экран. Для этого нажмем правый кнопкой мыши по окну иерархии в любом месте. В появившемся контекстном меню выберем пункт “New” => “Activity” => “Empty Activity”.

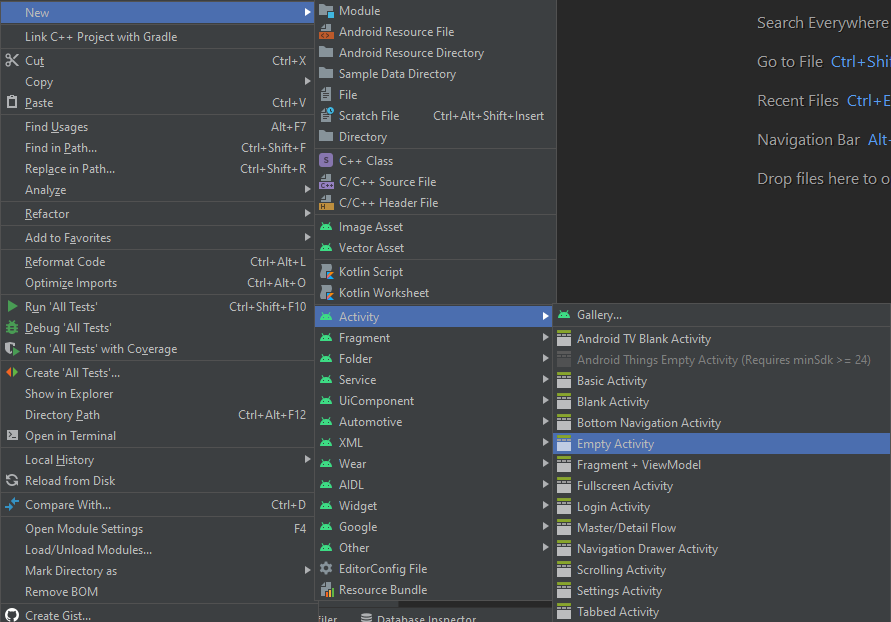


Рис. 14. Пункт выпадающего меню для создания нового экрана.

В появившемся окне называем новый экран и ставим галочку около пункта “Launcher Activity”, это покажет android устройству, что данный экран является стартовым для приложения. Нажимаем кнопку готово.

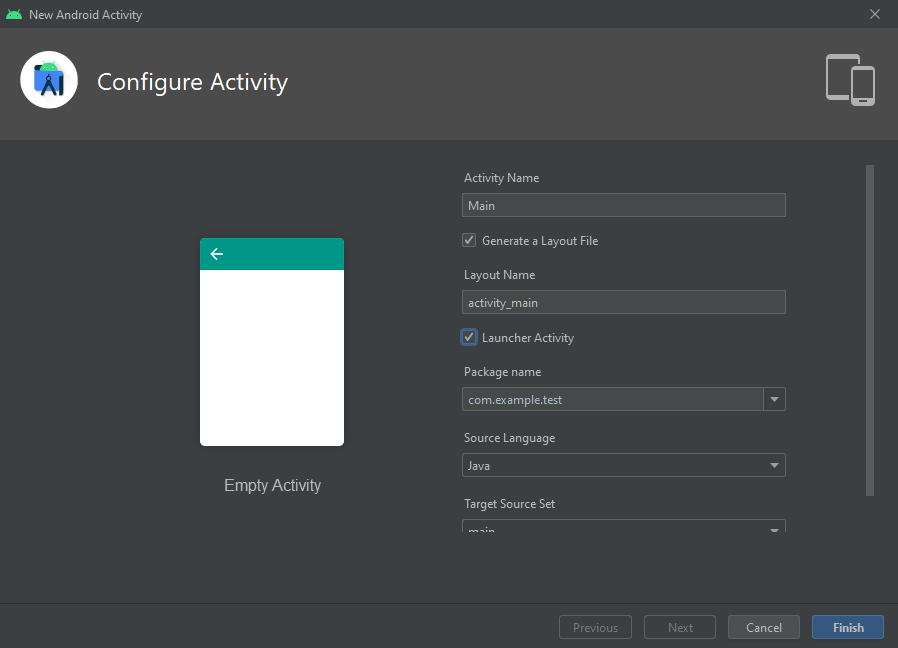


Рис. 15. Окно настройки нового экрана.

В проекте после этого появилось 2 файла: Java класс Main и XML файл activity\_main. Java класс будет хранить всю логику связанную с данным экраном, а XML файл будет хранить информацию об элементах интерфейса и их расположению на экране. Подробнее разберем их в следующих методичках.

В иерархии проекта можно переключать фильтр отображаемых объектов. Для этого нажмем на раскрывающийся список с выбранном элементов “Android”.

Раздел “Android” фильтрует иерархию так, чтобы разработчику было удобно просматривать проект. В иерархии остается файл манифеста, директория с тестами приложения, директория с Java классами, директория с ресурсами приложения и настройки сборки проекта.

Раздел “Project” не фильтрует иерархию и показывает все директории проекта. Это может пригодится если необходимо создать дополнительные директории или открыть другие файлы проекта.

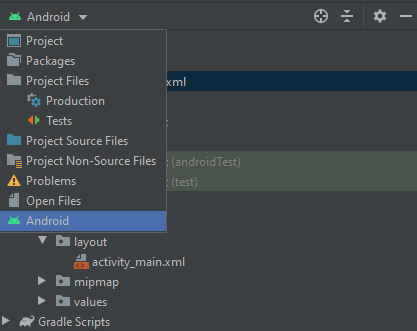


Рис. 16. Выпадающий список с фильтрами иерархии.

Сравните папки и файлы, которые отображаются с различными фильтрами. Также посмотрите манифест проекта, файлы в папках раздела “res” и “gradle scripts”. Опишите что находится в каждом из разделов.

**Теоретическая часть:**

Activity – это экраны приложения. Activity состоит из двух взаимосвязанных файла: XML файла и Java класса. Java классы разберем в следующий лабораторной работе.

XML – расширяемый язык разметки. В Android Studio он используется для хранения информации об элементах экрана, их расположению, смещениям, цвету и прочее. В проекте все XML файлы находятся в директории “res” по различным папкам, которые делят их по их назначению. Например в папке “layout” хранятся экраны приложения, в папке “values” различные значения: цвета, строки, числа и пр.

В XML элементы располагаются на экране в контейнерах (Layout). Они сами настраивают расположение элементов внутри себя. Расмотрим основные контейнеры в Android Studio:

* ConstraintLayout – элементы в этом контейнере привязываются к другим элементам или краям самого контейнера. Также можно настроить отступы от них.

Для этого используются свойства:

* + layout\_constraintBottom\_toBottomOf – привязка нижний части объекта
  + layout\_constraintEnd\_toEndOf toBottomOf – привязка правой части объекта
  + layout\_constraintStart\_toStartOf– привязка левой части объекта
  + layout\_constraintTop\_toTopOf– привязка верхний части объекта

Названия свойств могут отличать от данных из-за типа привязки. Лучшим способом обозначить привязку будет перетянуть белую точку на объекте к другому объекту и выбрать тип привязки, если требуется.

* LinearLayout – элементы в этом контейнере располагаются последовательно вертикально/горизонтально.

Для обозначения ориентации используеться свойство “orientation”. Оно поддерживает значения “horizontal” и “vertical”.

* ScrollView – контейнер, который позволяет листать его содержимое. Листать содержимое можно и в редакторе при помощи колесика мыши.
* RelativeLayout – элементы в этом контейнере располагаются в зависимости от настроенных на них отступах (padding и margin). Отступы можно задавать как для всех сторон при помощи одного свойства, так и отдельно для каждый стороны/плоскости (вертикально и горизонтально). Для этого используются соответствующие приписки к свойствам.

Расмотрим стандартные элементы XML для Android Studio:

* Button – кнопка
* ImageView – картинка

Для задания картинки нужно импортировать ее в проект и указать на нее ссылку в свойстве “src”.

* TextView – текст
* EditText – поле для ввода текста

У всех элементов XML и контейнеров есть базовые свойства, такие как:

* layout\_width – ширина элемента
* layout\_height – высота элемента

Данные свойства нужно прописывать для каждого элемента. Они принимают в себя 3 значения:

* “wrap\_content” – по размеру содержимого
* “match\_parent” – по размеру родителя
* Точное значения в dp, они используются вместо обозначения в пикселях, для стандартизации картинки на разных устройствах, потому что размер экрана в пикселях может меняться от устройства к устройству, а размер экрана в dp будет всегда одинаковый на экранах одного размера (маленький, средний, большой и т.д.)

Также у некоторых элементов могут быть свойства:

* text – задает текст внутри элемента
* textAlignment – выравнивает текст внутри элемента
* textSize – задает размер текста в sp
* layout\_gravity – выравнивает элемент внутри родителя
* fontFamily – задает шрифт текста
* foreground – задает верхний слой для элемента, можно указать как цвет, так и картинку
* background – задает фон для элемента, можно указать как цвет, так и картинку
* textColor – задает цвет текста
* textStyle – стиль текста (полужирный (bold) и курсив (italic))

**Практическая часть**

При создании Activity в первой лабораторной работе автоматически создался пустой XML файл. Откроем его. Рассмотрим интерфейс редактирования XML файлов.



Рис. 1. Интерфейс редактирования XML файлов.

В левом нижнем углу находится иерархия экрана. При добавлении новых элементов на экран, там будут появляться эти элементы. Если был добавлен контейнер, то он создаст подкатегорию, в которую можно будет поместить другие элементы или контейнеры. Иерархия доступна только в режиме “Design”.

В правом верхнем углу можно выбрать режим редактирования XML файла. При создании автоматически открывается режим “Design”. В данном режиме можно создавать экран приложения перетаскивая на него элементы интерфейса.

Также можно включить экран отображения кода XML файла. Для этого выберем режим “Code”. В этом режиме откроется только код XML файла, что не удобно при создании экрана, ведь не видно результата редактирования.

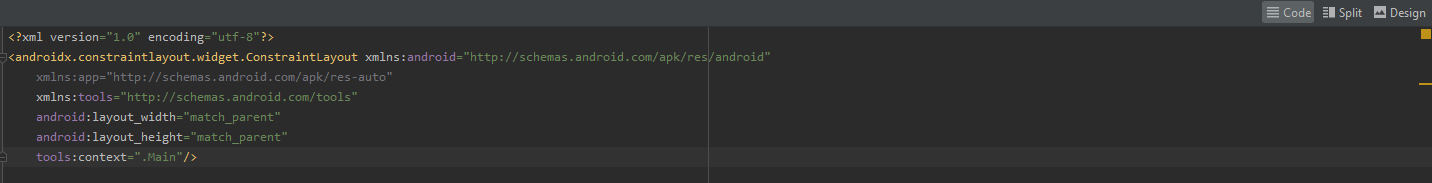


Рис. 2. Режим редактирования кода.

Чтобы видеть результат редактирования и изменять код, выберем режим “Split”. Данный режим предпочтителен для верстки/настройки экрана, но делать это можно и при помощи свойств и режима “Design”.

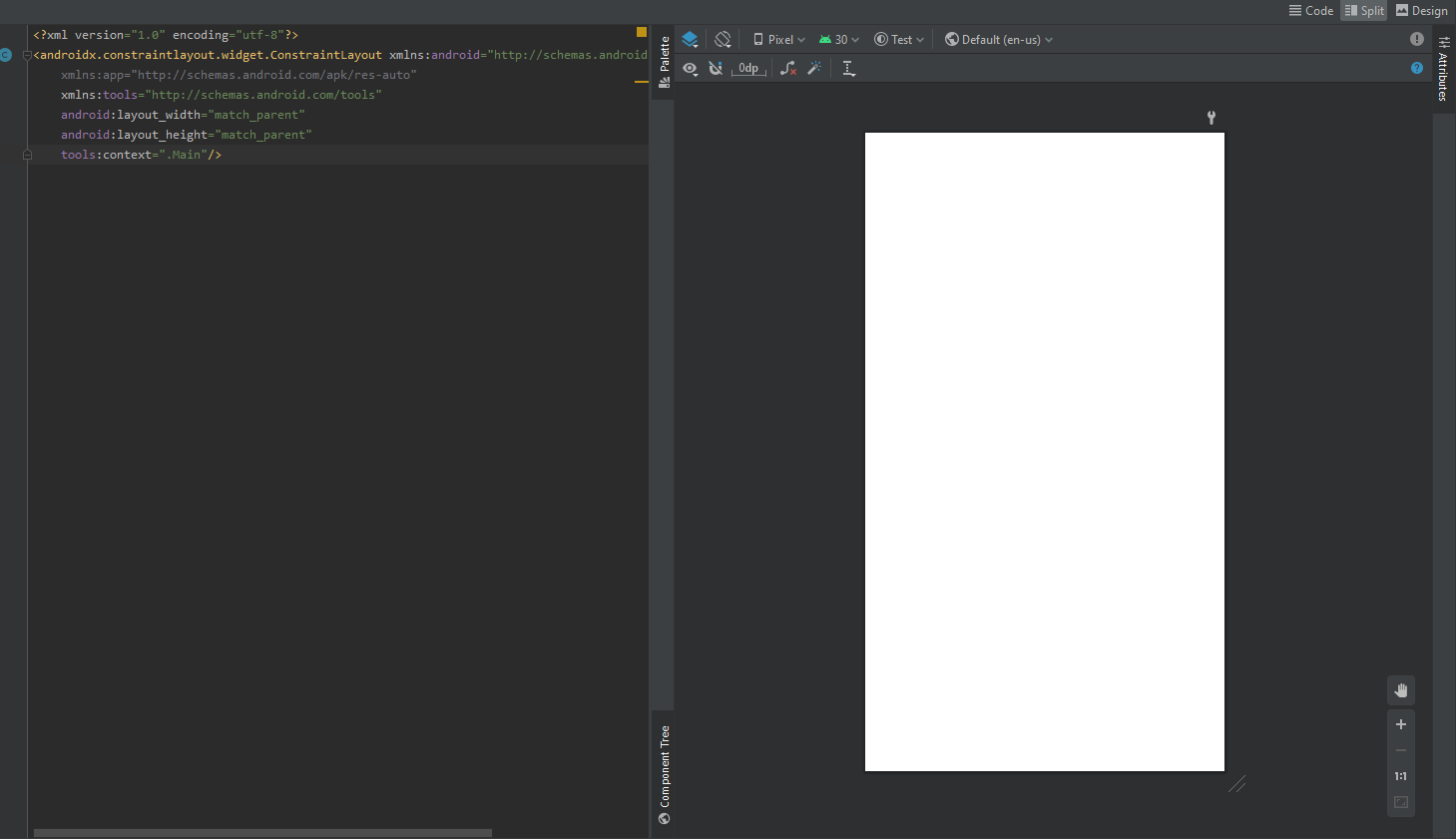


Рис. 3. Режим редактирования кода с экраном.

Рассмотрим элементы интерфейса, для этого выберем режим “Design” и перетащим на экран элементы “Button”, “TextView”, “Edit Text” и “ImageView”. При попытки перетащить ImageView на экран Android Studio спрашивает какое изображение выбрать для отображения. Для загрузки нового изображения нажмем на кнопку “+” в левом верхнем углу и выберем “Image Asset”.

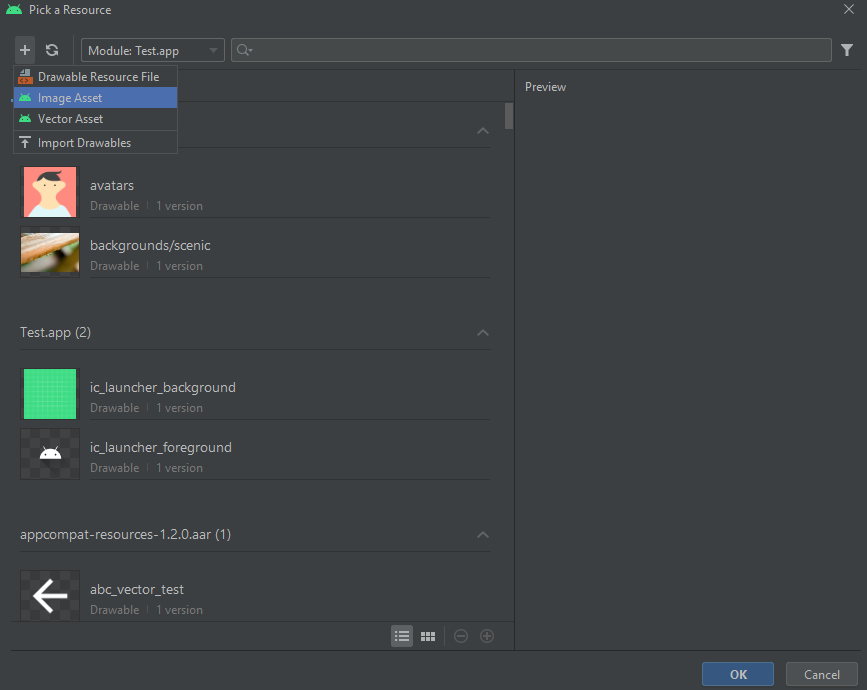


Рис. 4. Добавление новой картинки.

В появившемся окне выставляем параметры:

* Icon Type – Launcher Icons (Legacy only)
* Asset Type – Image
* Path – путь до любой картинки на компьютере, URL путь до картинки в интернете не подойдет для этого способа. Для лабораторной работы можно использовать как любую картинку на компьютере, так и скачать из интернета. Но стоит учитывать, что подобным образом импортируются изображения больше похожие на иконки, большие изображения или изображения, которые некорректно отображаются в окне предпросмотра, можно импортировать в проект просто перетащив их в папку “drawable” в директории “res”. При таком импорте изображений нужно убедиться, что android studio импортирует картинку именно в папку “drawable”, а не в производную от нее.
* Name
* Shape – форма картинки

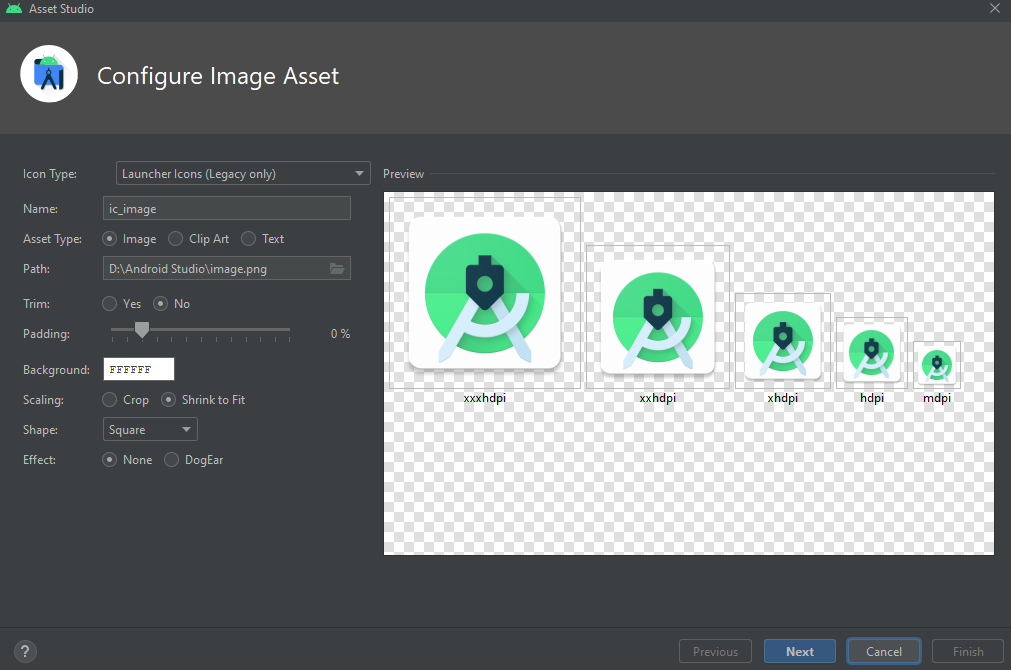


Рис. 5. Настройка картинки.

После этого нажимаем Next и Finish. После этого в папку “minimap” в иерархии “res/minimap” добавилась еще одна папка “ic\_image”, но при попытке ее открыть нам откроется сама картинка. Это происходит из-за того, что Android Studio при добавлении картинки в проект создал несколько ее копий в разных размерах изображений: hdpi, mdpi, xhdpi, xxhdpi, xxxhdpi. Это позволяет правильно отображать элементы на различных устройствах, таких как умные часы, смартфоны, планшеты и телевизоры.

Вернемся к добавлению картинки на экран. В окне добавления ImageView выберем Mip Map и картинку, которую только что добавили.

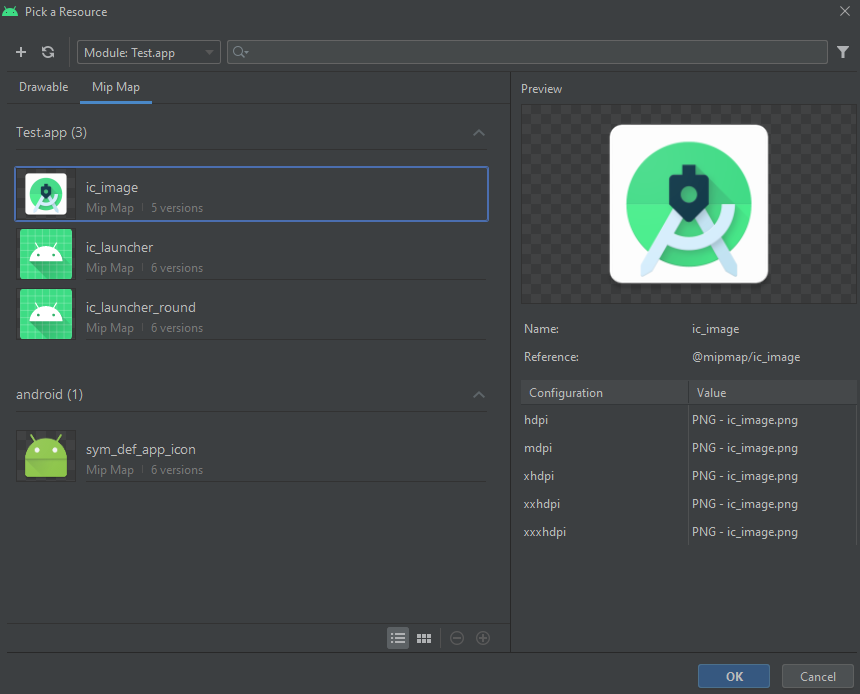


Рис. 6. Добавление картинки.

Рассмотрим различия контейнеров для элементов. По умолчанию первый контейнер для элементов это ConstaintLayout. Для размещения в нем элементов их нужно привязывать к другим элементам этого контейнера или к краям самого контейнера. Для этого выберем любой элемент и перетянем его край к объекту, к которому мы его хотим привязать.

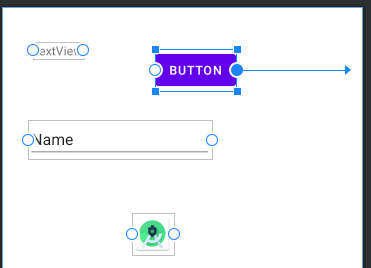


Рис. 7. Привязка элемента к краю контейнера.

Также можно расположить элементы на экране, а потом в окне “Свойств” (справа) выставить все связи автоматически. Для этого нужно нажимать на синие значки по сторонам квадрата.

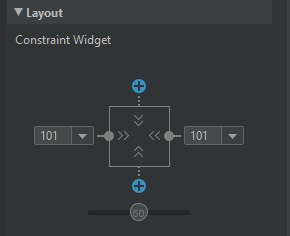


Рис. 9. Автоматическая привязка элемента.

Данный способ позволяет быстро создать дизайн приложения для конкретного телефона. Но при изменении разрешения экрана, дизайн может показывается некорректно.

Изменим в коде XML файла строку “androidx.constraintlayout.widget. ConstraintLayout” на “RelativeLayout”. Все элементы, которые мы располагали на экране до этого, переместились в левый верхний угол. Для того, чтобы располагать элементы в этом контейнере нужно использовать отступы.

Для их настройки перейдем в код XML файла. Для всех элементов настроим Padding и Margin, так чтобы элементы не располагались друг в друге. Таже можно использовать свойство “layout\_alignParent” для привязки элемента к определенной стороне контейнера. Например “layout\_alignParentBottom = “true”” привяжет элемент к низу контейнера.

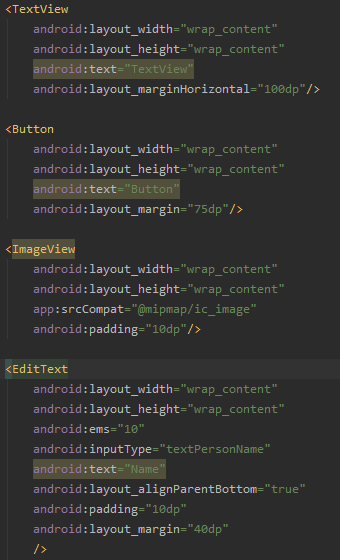


Рис. 10 – 11. Пример использования RelativeLayout.

Данный способ обычно комбинируют с использованием LinearLayout. Для этого удалим все элементы в иерархии дизайна экрана (окно слева снизу). И добавим туда контейнеры LinearLayout таким же образом что и на рисунке 12.

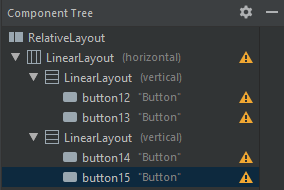


Рис. 12. Структура дизайна экрана.

Если посмотреть на экран, то мы увидим только 2 кнопки вместо 4. Это происходит из-за того, что кнопка растягивается на всю ширину. Чтобы это исправить, откроем режим редактирования кодаXML файла. Найдем все теги кнопок и изменим параметр “layout\_width = “match\_parent”” на “wrap\_content”. Тоже самое сделаем для всех контейнеров, кроме первого.

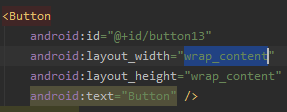


Рис. 13. Настройка контейнеров.

В итоге все 4 кнопки должны образовать квадрат 2x2. Настроим padding и margin для всех кнопок и контейнеров.

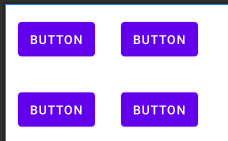


Рис. 14. Дизайн приложения.

С помощью комбинации этих способов можно делать дизайн интерфейса приложений для всех устройств и разрешений, гибко настраивая его.

Создадим новый проект, в котором начнем реализовывать прототип проекта, сделанный на прошлых лабораторных работах. Посмотрим на прототип проекта в figma.

 (этот файл можно скачать, если открыть его через браузер)

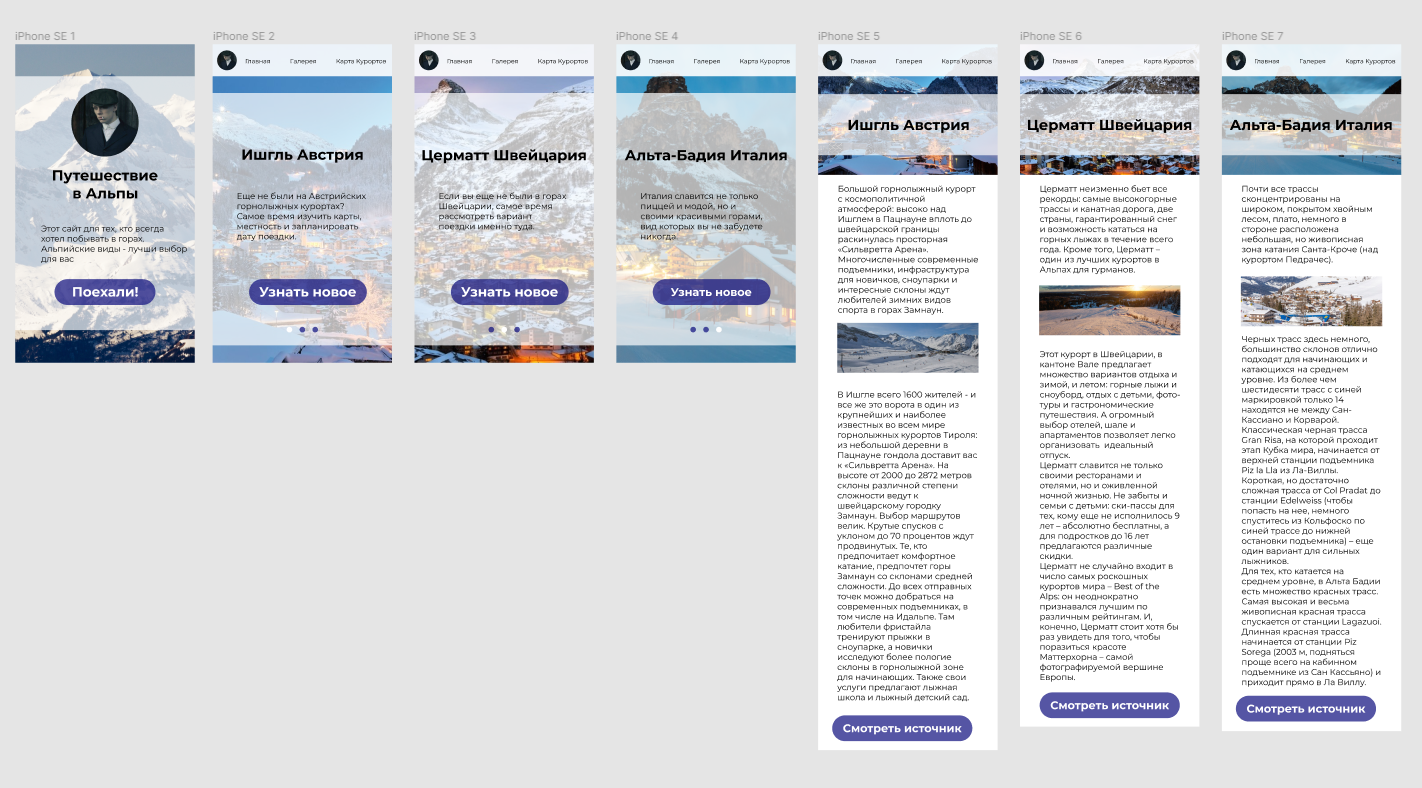


Рис. 15. Проект

Создадим 3 Activity. Начнем делать первый экран. На стартовом экране есть 2 картинки, копируем их в качестве png (ПКМ по элементу с картинкой в иерархии > Copy/Paste as > Copy as PNG). После этого изображение будет в буфере обмена, но вставить в проект его не получиться, для этого его нужно сохранить в виде файла. Сделать это можно при помощи любого графического редактора, например paint или gimp. Фон на круглом фото профиля можно обрезать в android studio, для этого нужно выбрать круг в форме изображения при его импорте.

Импорт изображений необязательно делать через создание Image View, можно вызвать окно добавления изображения через ПКМ по иерархии проекта > New > Image Asset.

Заменим основной контейнер на RelativeLayout.

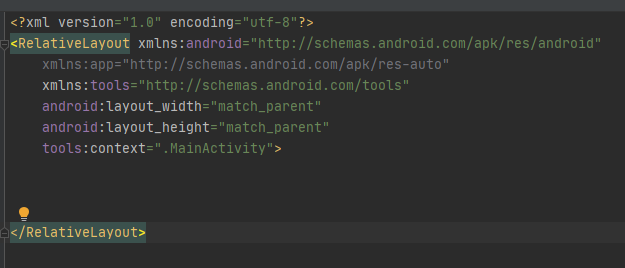


Рис. 18. XML код пустого экрана.

Чтобы добавить фон напишем свойство background для тега RelativeLayout. Укажем в нем путь до фона, который мы импортировали. Если начать писать название файла, то android studio сам подскажет, где он находиться.

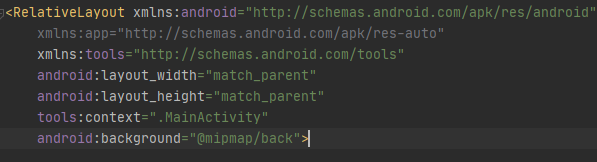


Рис. 19. XML код добавления фона

Для создания серого полупрозрачного фона, создадим элемент LinearLayout и выберем для него цвет фона. Для этого зайдем в figma и найдем его. Получается цвет #B3FFFFFF по aRGB. Также можно нажать на иконку цвета слева от свойства background, когда там выбран цвет и настроить его в открывшемся окне.

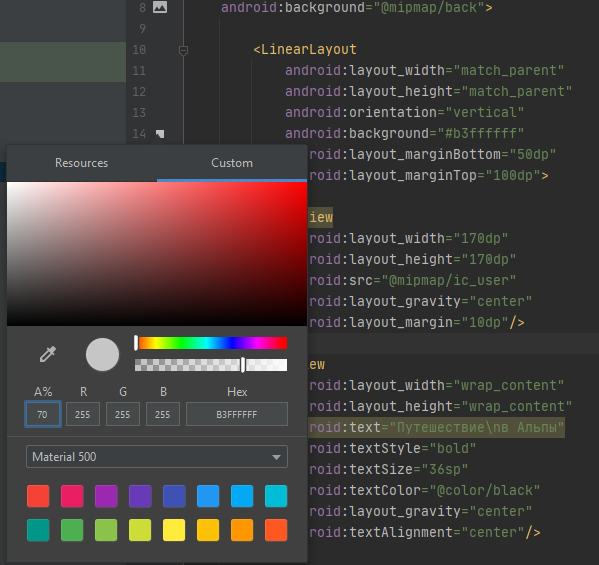


Рис. 20. Окно настройки цвета.

Вставляем внутрь LinearLayout картинку профиля. Выставляем размер картинке в 170dp и располагаем ее по центру с отступом во все стороны.

Так как LinearLayout выставляет все дочерние элементы друг за другом, то просто после картинки вставляем текст и настраиваем его. После чего также настраиваем кнопку. В итоге получаем готовый экран



Рис. 21. Готовый экран

Цвет в превью может отличатся от реального. Для просмотра реального цвета запустим приложение нажав на зеленую стрелочку “Run app”

В готовом приложении будет полоска с названием приложения сверху экрана, чтобы убрать ее нам нужно поменять стиль приложения в манифесте, для этого дважды кликаем на папку манифеста в инспекторе и в поле android:theme пишем “ @style/Theme.AppCompat.NoActionBar”

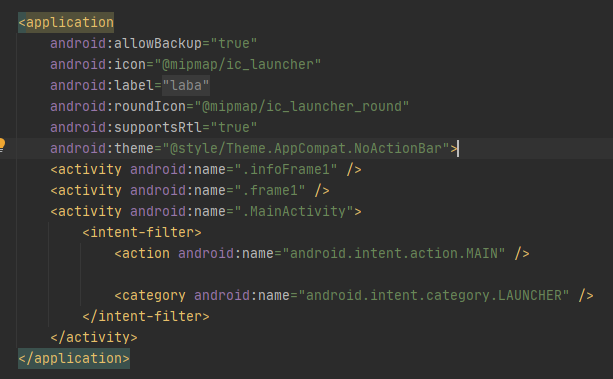


Рис. 22. Нужная строка в манифесте.

По аналогии делаем 2 оставшихся экрана, но в экране с возможностью скролла, после основного контейнера делаем контейнер Scroll View. Для создания swipe меню импортируем SVG круги. Для их импорта нужно выбрать Vector Asset вместо Image Asset. При скачивании SVG файлов браузер откроет их код. Чтобы сохранить их нужно создать текстовый файл, вставить в него содержимое SVG файла и сохранить как файл с форматом SVG. (эти файлы также можно скачать)



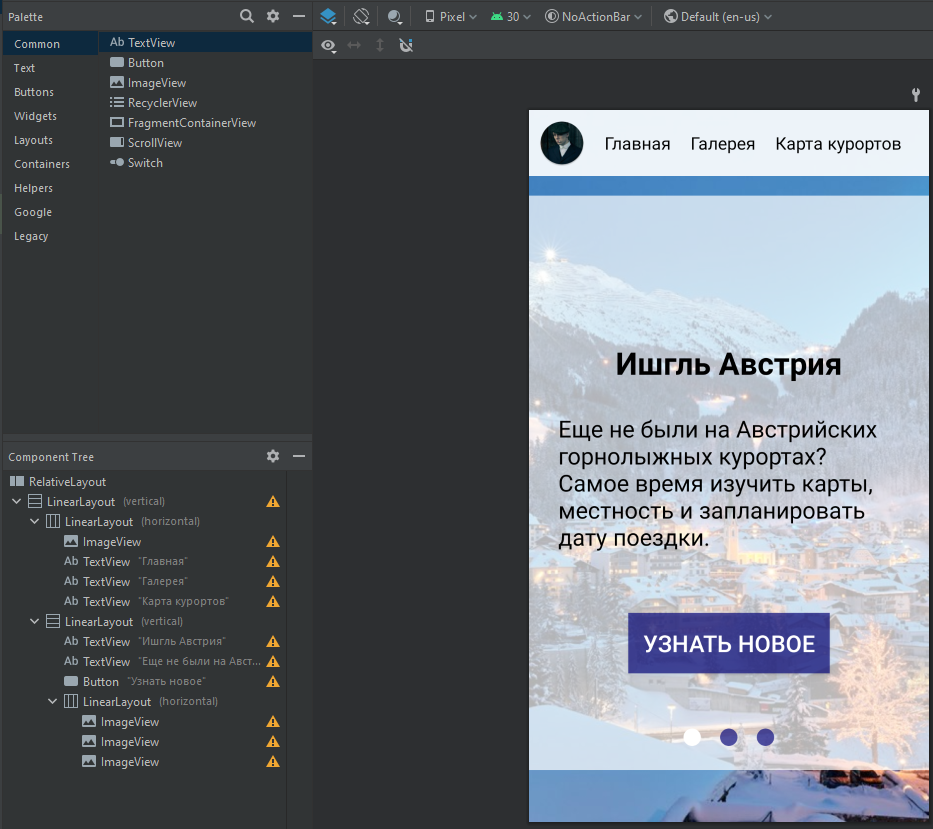


Рис. 23. Первый экран.

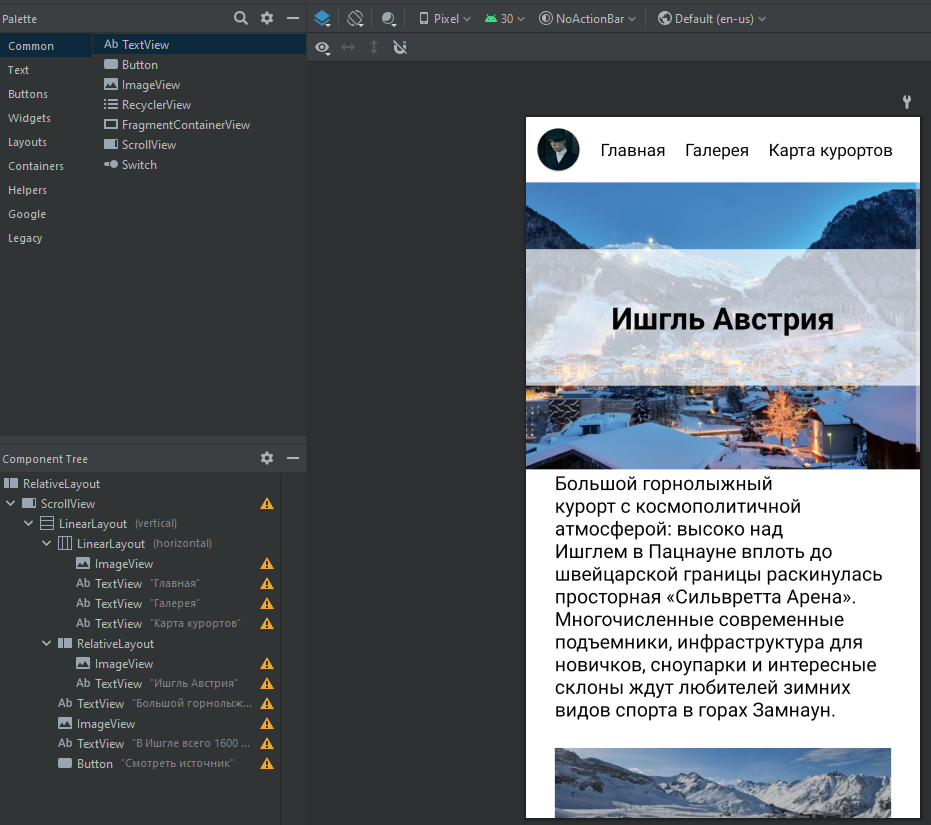


Рис. 24. Второй экран.

Для создания остальных экранов создадим новые Activity. Заменим основной контейнер на RelativeLayout и скопируем все остальные элементы из уже готовых экранов. После чего поменяем текст и картинки в получившихся копиях.