

**КУРСОВОЙ ПРОЕКТ**

Тема:«Мобильная игра-аналог «Змейка» »

Студентка группы №3ИСП11-6 Карбан Наталия Анатольевна­

Специальность: 09.02.07 Информационные системы и программирование

Руководитель /А.Н.Виеру/

Допустить к защите:

Председатель ПЦК (КМК) / /

Оценка Дата

г.Москва

2020г.

ОГЛАВЛЕНИЕ

Введение……………………........................................................................

Глава I.Теоретические аспекты разработки мобильных приложений……

1.1 Понятие информационной системы и Android Studio…….…….…….

1.2 Особенности Android Studio……………………………………………..

1.3 Жизненный цикл мобильного приложения……………………………..

Выводы……………..………………………………………………………

Глава II. Разработка мобильного приложения «Twilight star»…

2.1 Анализ и расстановка целей для предприятия…………………………

2.2.Анализ и сравнение программ для разработки мобильных

приложений……………………………………………………………….

2.3. Разработка мобильного приложеня……………………………………

Заключение......................................................................................................

ВВЕДЕНИЕ:

С развитием технологий компьютеры все больше и больше вливаются в повседневную жизнь человека. Персональные компьютеры - это уже не просто машины для математических вычислений, их общедоступность значительно расширила область применения. И, конечно же, здесь не обошлось без игр.

Змейка тренирует внимательность, мелкую моторику и реакцию. Позволяет на некоторое время отвлечься от внешних проблем и немного расслабиться.

Жанр игры: аркада, экшен, инди, казуальная игра, реиграбельность.

Игра «Twilight star» краткое описание: Карта выполнена в формате шахматной доски. По карте ползает синяя змея, задача которой является поимка звезд. Новые звезды генерируются через 2 секунды, либо когда их поймает змея. Управление ведется с помощью скольжения пальца по экраную

Карта имеет невидимые границы, о которых сообщается в обучении, которое проходит пользователь в начале игры. Программа игры позволяет звездам генерироваться внутри границ, с целью усложнения игрового процесса для пользователя.

Снизу экрана находится два счетчика:

* Счетчик звезд(обновляется после каждой пойманной звезды)
* Счетчик рекордов (обновляется после окончания игры)

Игра заканчивается при столкновении змейки со стеной или своим телом. По окончанию игры выводится информация о конце и рекорде пользователя.

ЦЕЛЬ:

* систематизация и закрепление полученных теоретических знаний и практических умений по общепрофессиональным и специальным дисциплинам специальности “Информационные системы и программирование”;
* углубление теоретических знаний в соответствии с заданной темой;
* формирование умений применять теоретические знания при решении поставленных вопросов;
* формирование умений использовать справочную, нормативную и правовую документацию;
* развитие творческой инициативы, самостоятельности, ответственности и организованности;
* изучение основ игровых механик и их реализация в курсовом проекте;

Для достижения цели работы должны быть решены следующие задачи:

* осуществить постановку задачи, выделить требования к приложению;
* изучить современные средства разработки мобильных приложений для Android;
* определить требования и спроектировать мобильное приложение;
* реализовать и протестировать мобильное приложение.

Структура и объем работы:

Работа состоит из введения, двух глав, заключения и библиографического списка. Объем работы составляет 29 страницы, объем библиографии – 25, объем приложений – 8 страниц.

Глава I. ТЕОРЕТИЧЕСКИЕ АСПЕКТЫ РАЗРАБОТКИ МОБИЛЬНЫХ

ПРИЛОЖЕНИЙ

* 1. Понятие информационной системы и AndroidStudio

Информационная система (ИС)— система, предназначенная для хранения, поиска и обработки информации, и соответствующие организационные ресурсы (человеческие, технические, финансовые и т. д.), которые обеспечивают и распространяют информацию (ISO/IEC 2382:2015).

По масштабу информационные системы подразделяются:

* корпоративные;
* групповые;
* одиночные.

По способу организации групповые и корпоративные информационные системы подразделяются на классы:

* системы на основе архитектуры файл-сервер;
* системы на основе архитектуры клиент-сервер;
* системы на основе многоуровневой архитектуры;
* системы на основе Интернет/интранет - технологий.

По сфере применения информационные системы составляют четыре группы:

* системы обработки транзакций;
* системы принятия решений;
* информационно-справочные системы;
* офисные информационные системы.

AndroidStudio — среда разработки под популярную операционную систему Андроид. Программное обеспечение вышло в 2013 году и развивается по сегодня. В каждой новой версии AndroidStudio разработчик добавляет увеличивает функционал, оптимизирует процессы и другое. В комплекте с IDE идет эмулятор, проверяющий корректную работу уже написанных утилит, приложений на разных конфигурациях. Прямой конкурент самой популярной утилиты для создания софта под Android – Eclipse.

AndroidStudio превосходит конкурента по многим параметрам, к которым можно отнести:

* гибкость среды разработки;
* больший набор функций;
* процесс разработки, который подстраивается под разработчика.

Во время создания приложений и утилит для операционной системы Android, пользователь программного обеспечения может наблюдать за изменениями в проекте, в режиме реального времен

AndroidStudio – универсальная среда разработки, так как позволяет оптимизировать работу будущих приложения для работы не только на смартфонах, но и на планшетах, портативных ПК, которые работают на основе рассматриваемой операционной системы.

1. Особенности Android Studio

В программу встроен эмулятор, позволяющий проверить корректную работу приложения на устройствах с разными экранами, с различными соотношениями сторон. Особенно актуальной эта функция стала после входа в тренды смартфонов, в которых установлены экраны с соотношением сторон 18:9.

Отличительная особенность эмулятора – просмотр приблизительных показателей производительности при запуске приложения на самых популярных устройствах.

Среда разработки для приложений Android Studio последней версии стала по-настоящему удобной даже для начинающих разработчиков. В программе реализованы все современные средства для упаковки кода, его маркировки. Востребованная многими создателями ПО функция Drag-n-Drop, облегчающая перенос компонентов в среду разработки непосредственно.

Локализация приложений становится существенно проще с функцией SDK, которая также входит в перечень достоинств Android Studio.

* Связь с целевой аудиторией приложения после его релиза поможет реализовать инструмент от компании Google – GoogleCloudMessaging.
* среда разработки поддерживает работу с несколькими языками программирования, к которым относятся самые популярные – C/C++, Java.
* редактор кода, с которым удобно работать;
* позволяет разрабатывать приложения не только для смартфонов/планшетов, а и для портативных ПК, приставок для телевизоров Android TV, устройств AndroidWear, новомодных мобильных устройств с необычным соотношением сторон экрана;
* тестирование корректности работы новых игр, утилит, их производительности на той или иной системе, происходит непосредственно в эмуляторе;
* рефакторинг уже готового кода;
* достаточно большая библиотека с готовыми шаблонами и компонентами для разработки ПО;
* разработка приложения для Android N – самой последней версии операционной системы;
* предварительная проверка уже созданного приложения на предмет ошибок в нем;
* большой набор средств инструментов для тестирования каждого элемента приложения, игры;
* для неопытных/начинающих разработчиков специально создано руководство по использованию AndroidStudio, размещенное на официальном сайте утилиты.

## Недостатки/спорные моменты

Несмотря на наличие встроенного Android-эмулятора в самой среде разработки, с тестированием новоразработанного приложения могут возникнуть трудности. Так, для его запуска необходима достаточно внушительная по производительности аппаратная основа ПК, на котором планируется тестирование.

Еще один недостаток - это невозможность написать серверные проекты на языке Java для ПК, Android устройств.

## Выводы

ПО для разработки утилит и приложения на Android действительно создает приятное первое впечатление богатый набор инструментов, гибкость в разработке, возможности тестирования, поддержка нескольких языков программирования и встроенный эмулятор делают утилиту одной из лучших в своей нише.

1. Жизненный цикл мобильного приложения

Ключевым компонентом для создания визуального интерфейса в приложении Android является activity (активность). Нередко activity ассоциируется с отдельным экраном или окном приложения, а переключение между окнами будет происходить как перемещение от одной activity к другой. Приложение может иметь одну или несколько activity.

Все приложения Android имеют строго определенный системой жизненный цикл. При запуске пользователем приложения система дает этому приложению высокий приоритет. Каждое приложение запускается в виде отдельного процесса, что позволяет системе давать одним процессам более высокой приоритет, в отличие от других. После прекращения работы с приложением, система освобождает все связанные ресурсы и переводит приложение в разряд низкоприоритетного и закрывает его.

Все объекты activity, которые есть в приложении, управляются системой в виде стека activity, который называется backstack. При запуске новой activity она помещается поверх стека и выводится на экран устройства, пока не появится новая activity. Когда текущая activity заканчивает свою работу (например, пользователь уходит из приложения), то она удаляется из стека, и возобновляет работу та activity, которая ранее была второй в стеке.

Все объекты activity представляют собой объекты класса android.app.Activity, которая содержит базовую функциональность для всех activity. При переходе активности от одного состояния к другому, она получает уведомления через защищенные методы:

* protected void onCreate(Bundle savedInstanceState);
  + protectedvoidonStart();
  + protectedvoidonRestart();
  + protectedvoidonResume();
  + protectedvoidonPause();
  + protectedvoidonStop();
  + protectedvoidonDestroy()

Схематично взаимосвязь между всеми этими обратными вызовами можно представить следующим образом (Рис.1.1):

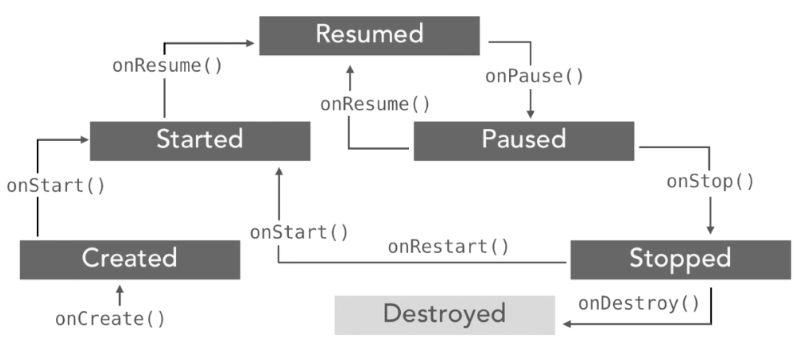


Рис.1.1. Взаимосвязи между активностями AndroidStudio

Из перечисленных методов в классе обязательно должен быть метод onCreate(), которая задаёт начальную установку параметров при инициализации активности. Вторым по популярности является метод onPause(), используемый для сохранения пользовательских настроек активности и подготовиться к прекращению взаимодействия с пользователем.

В целом переход между состояниями activity можно выразить следующей схемой (Рис.1.2):

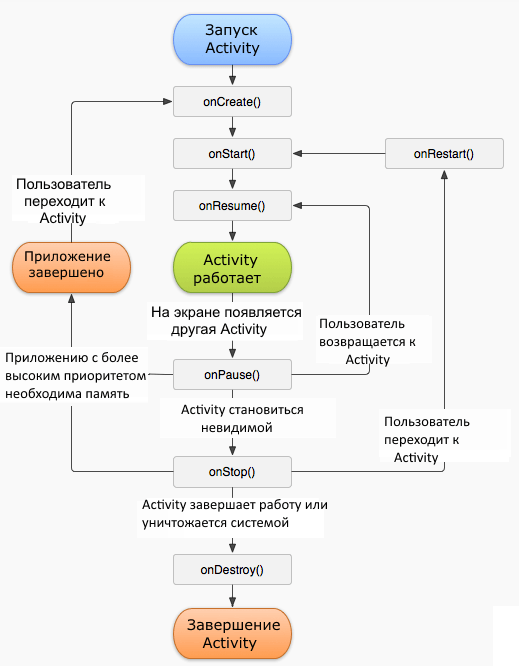


Рис.1.2. Переход между состояниями activity

Семь перечисленных методов определяют весь жизненный цикл активности. Есть три вложенных цикла, которые можно отслеживать в классе активности:

* полное время жизни (entirelifetime) — время с момента первого вызова метода onCreate() до вызова onDestroy(). Активность делает всю начальную установку своего глобального состояния в методе onCreate() и освобождает все остающиеся ресурсы в onDestroy(). Например, если активность порождает дополнительный поток, выполняющийся в фоновом режиме, можно создать этот поток в методе onCreate() и затем остановить поток в методе onDestroy();
* видимое время жизни (visiblelifetime) — время между вызовом метода onStart() и вызовом onStop(). В это время пользователь может видеть окно активности на экране, хотя окно может не быть на переднем плане и может не взаимодействовать с пользователем. Между этими двумя методами вы можете поддерживать в коде ресурсы, которые необходимы, чтобы отображать активность пользователю;
* активное время жизни (foregroundlifetime) — время между вызовами onResume() и onPause(). В это время окно активности находится на переднем плане и взаимодействует с пользователем. Активность в процессе работы приложения может часто переходить между состояниями *active* и *paused*, поэтому код в этих двух методах должен быть или небольшим по объему (чтобы не замедлять работу приложения во время выполнения), или порождать дополнительные потоки, если требуется выполнение задач, занимающих длительное время.

Можно написать код с заглушками для методов внутри Активности, которые обрабатывают изменения состояний. Комментарии к каждой такой заглушке описывают действия, которые нужно учитывать при обработке этих событий.

Выводы по главе 1.

В первой главе были рассмотрены понятие информационной системы (ИС), классификация информационных систем по масштабу, по способу организации и сфере применения. Так же была освещена среда разработки Android Studio, ее особенности, достоинства и недостатки. В последней подглаве рассказано о жизненном цикле приложения.

Глава II. РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ «ЗМЕЙКА»

2.1 Конструкции, функции, методы, использованные для создания приложения.

2.1.1 Класс BitMap

* [Битовая карта](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D1%82%D0%BE%D0%B2%D0%B0%D1%8F_%D0%BA%D0%B0%D1%80%D1%82%D0%B0) (битовый массив, битовый вектор) — структура данных.
* [Битовое изображение](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D1%82%D0%BE%D0%B2%D0%BE%D0%B5_%D0%B8%D0%B7%D0%BE%D0%B1%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5) — применение этой структуры в компьютерной графике.
* [Бинарное изображение](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D0%BD%D0%B0%D1%80%D0%BD%D0%BE%D0%B5_%D0%B8%D0%B7%D0%BE%D0%B1%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5) (двухуровневое, двоичное) — разновидность цифровых растровых изображений, когда каждый пиксель может представлять только один из двух цветов.
* [BMP](https://ru.wikipedia.org/wiki/BMP) — формат файлов растровой графики.

2.1.2 Класс SoundPoll

* Используют для проигрывания маленьких аудиоклипов. SoundPool может играть несколько звуков одновременно. Звуки также можно повторять (зациклить). В этом случае для остановки проигрывания нужно явно вызывать метод остановки.

2.1.3 Класс Handler

* Handler - это механизм, который позволяет работать с очередью сообщений. Он привязан к конкретному потоку (thread) и работает с его очередью. Handler умеет помещатьсообщения в очередь. При этом он ставит самого себя в качестве получателяэтого сообщения. И когда приходит время, система достает сообщение из очереди и отправляет его адресату (т.е. в Handler) на обработку.
* Позволяет реализоватьотложенное повремени выполнение кода
* Позволяет выполнение кода не в своем потоке.

|  |  |  |  |
| --- | --- | --- | --- |
| Название среды разработки | Разработчик | Платформы | Плюсы |
| 1 | 2 | 3 | 4 |
| Net Beans | Oracle | Windows, Mac и Linux | Бесплатная;  легкая интеграция с программным обеспечением для управления версиями;  открытый исходный код; |
| 1 | 2 | 3 | 4 |
| IntelliJ IDEA | JetBrains | Windows, Mac и Linux | Бесплатно для студентов, также существует версия с открытым исходным кодом — тоже бесплатная; |
| AndroidStudio | [Google](https://ru.wikipedia.org/wiki/Google_(%D0%BA%D0%BE%D0%BC%D0%BF%D0%B0%D0%BD%D0%B8%D1%8F)) | GNU/Linux, [macOS](https://ru.wikipedia.org/wiki/MacOS) и [Microsoft Windows](https://ru.wikipedia.org/wiki/Windows) | гибкость среды разработки;  больший набор функций;  процесс разработки, который подстраивается под разработчика;  во время создания приложений и утилит для операционной системы Android, пользователь программного обеспечения может наблюдать за изменениями в проекте, в режиме реального времени; |

2.2 Анализ и сравнение программ для разработки мобильных приложений

IDE (Integrated Development Environment) – это интегрированная, единая среда разработки, которая используется разработчиками для создания различного программного обеспечения.

IDE представляет собой комплекс из нескольких инструментов, а именно: текстового редактора, компилятора либо интерпретатора, средств автоматизации сборки и отладчика. Помимо этого, IDE может содержать инструменты для интеграции с системами управления версиями и другие полезные утилиты.

Есть IDE, которые предназначены для работы только с одним языком программирования, однако большинство современных IDE позволяет работать сразу с несколькими.

Java IDE - это программное обеспечение, которое содержит все необходимые инструменты, библиотеки и другие ресурсы, необходимые для программирования на Java.

Какая среда разработки подойдет конкретно для моего проекта? В этом вопросе надо основываться на трех факторах:

* язык разработки;
* простота использования;
* на каких платформах работает;
* стоимость.

Таблица 1. Сравнение сред разработки

Из вышеперечисленных пунктов следует, что IDE следует выбирать, исходя из возможностей и желаемого результата. В результате анализа сред разработки Java я пришла к выводу, что для реализации моего проекта наиболее подойдет Android Studio из-за преимуществ:

* среда разработки поддержиает работу с несколькими языками программирования, к которым относятся самые популярные – C/C++, Java.
* редактор кода, с которым удобно работать;
* позволяет разрабатывать приложения не только для смартфонов/планшетов, а и для портативных ПК, приставок для телевизоров Android TV, устройств AndroidWear, новомодных мобильных устройств с необычным соотношением сторон экрана;
* тестирование корректности работы новых игр, утилит, их производительности на той или иной системе, происходит непосредственно в эмуляторе;
* рефакторинг уже готового кода;

После выбора среды разработки для мобильного приложения перейдем к непосредственной разработке мобильного приложения «Twilight star».

2.3 Разработка мобильного приложения

Проанализировав средства разработки мобильных приложений и языки программирования, мы приступаем к разработке нашего приложения.

1. Сначала создается проект в Android Studio, представленный на рисунке 2.3:

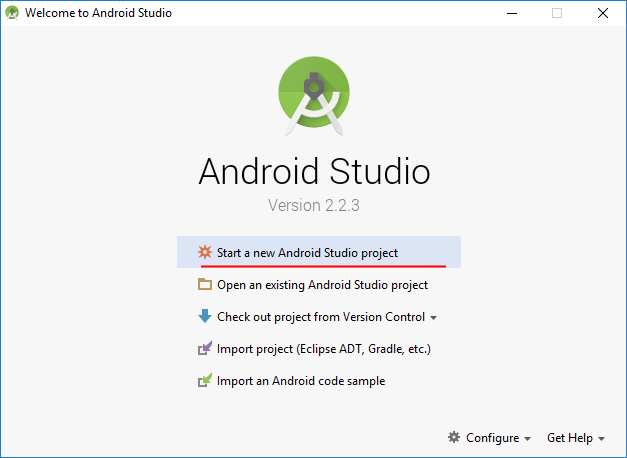


Рис. 2.3. Начальный экран при запуске приложения

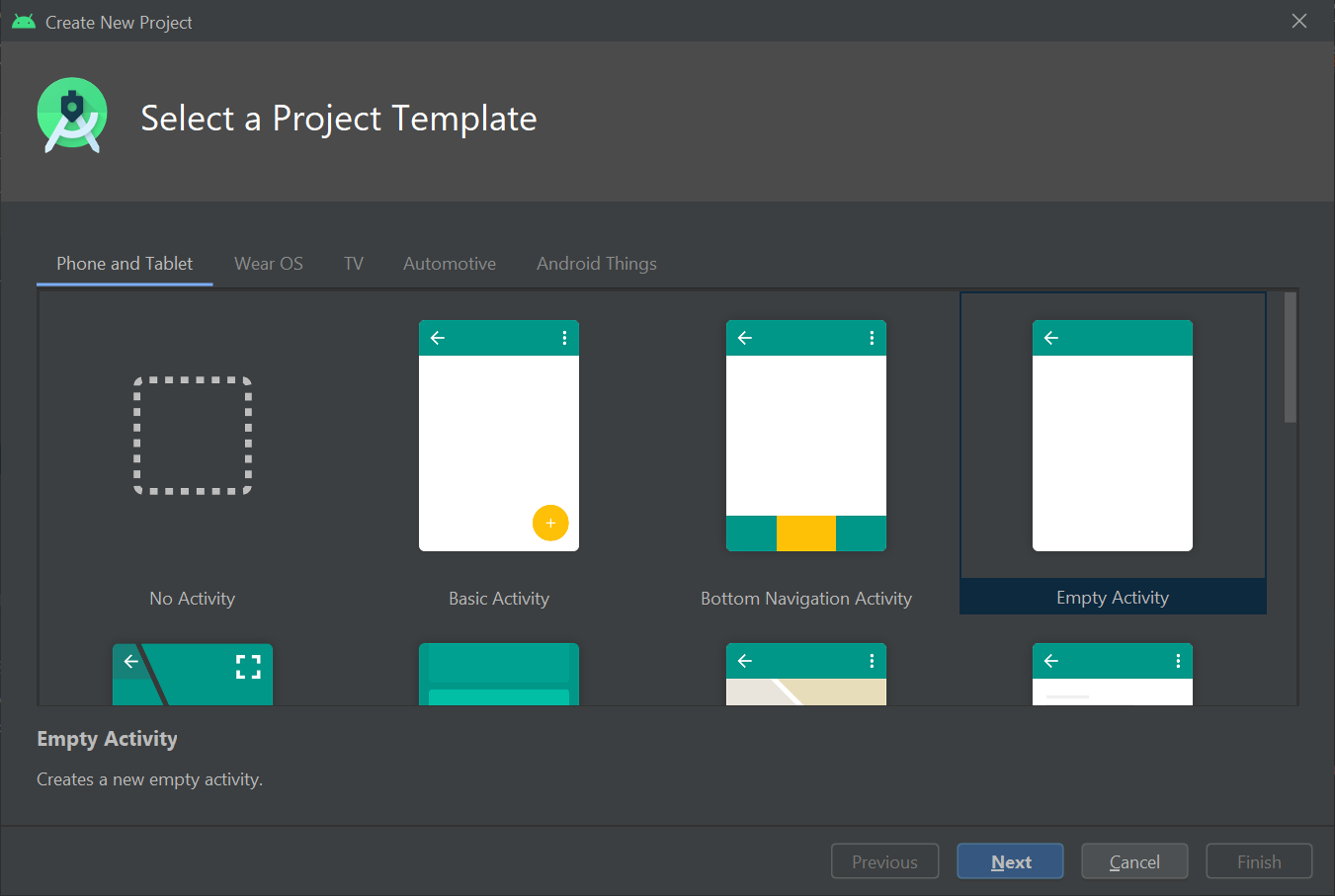
1. Затем выбирается нужный вариант проекта, выборка представлена на рисунке 2.4;

Рис. 2.4. Выбор доступных базовых надстроек

1. Далее указывается название проекта и папка, в которой он будет сохранен (Рис.2.5).

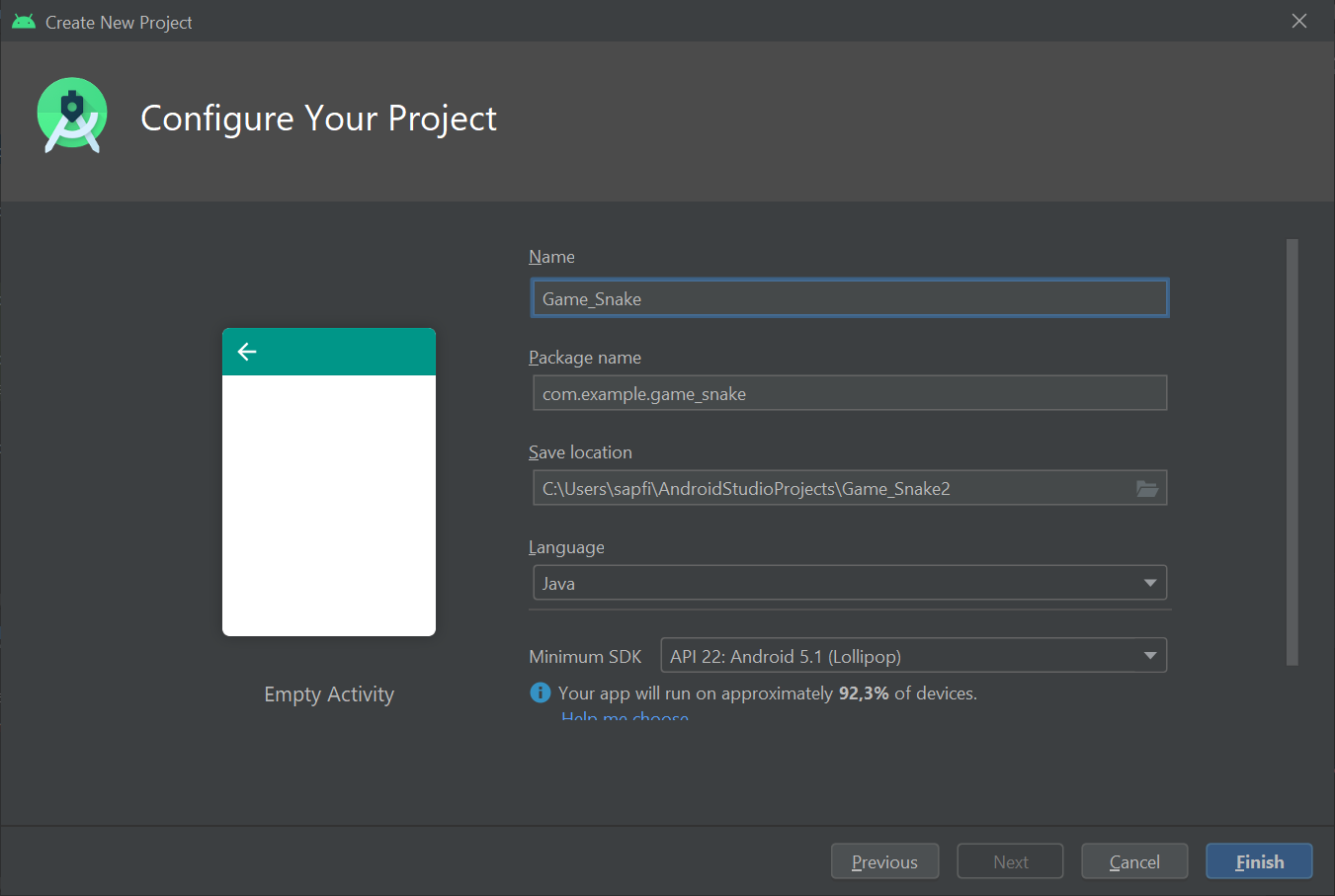


Рис. 2.5. Создание проекта и добавление в папку

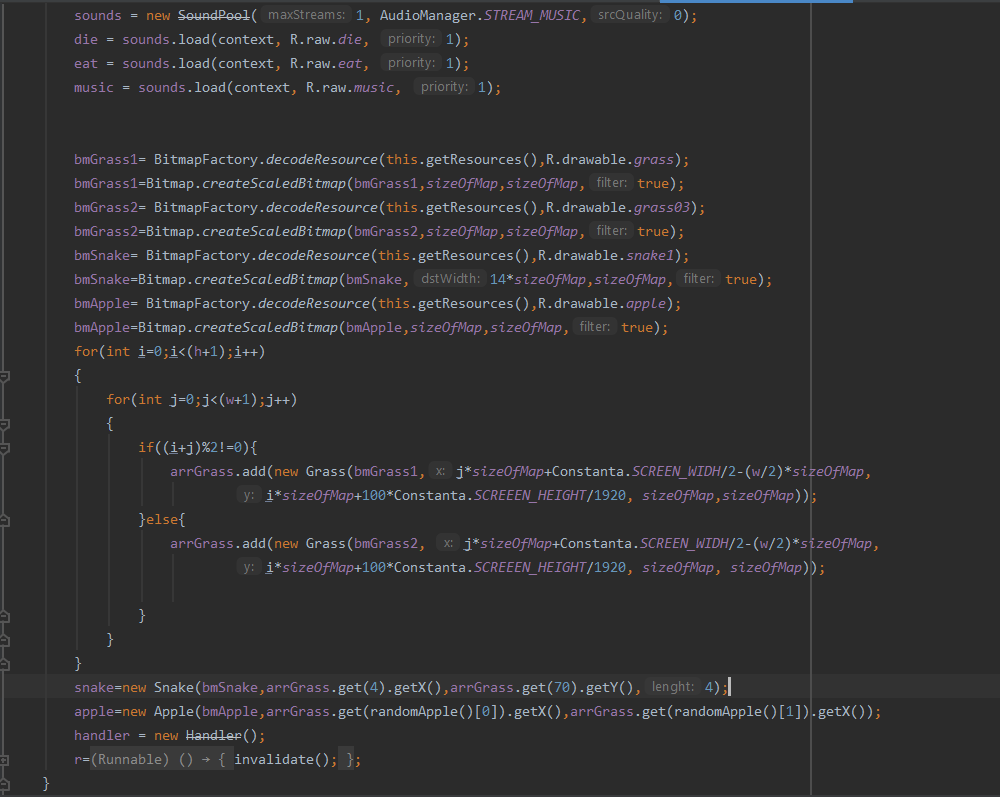
После этого начинается создание приложения. (Рис.2.6).

Рис. 2.6. Генерация поля, звезды и змейки. Объявление многопотока на передвижение змейки. Объявление звукового сопровождения

1. Далее прописывается необходимый код для работы приложения.

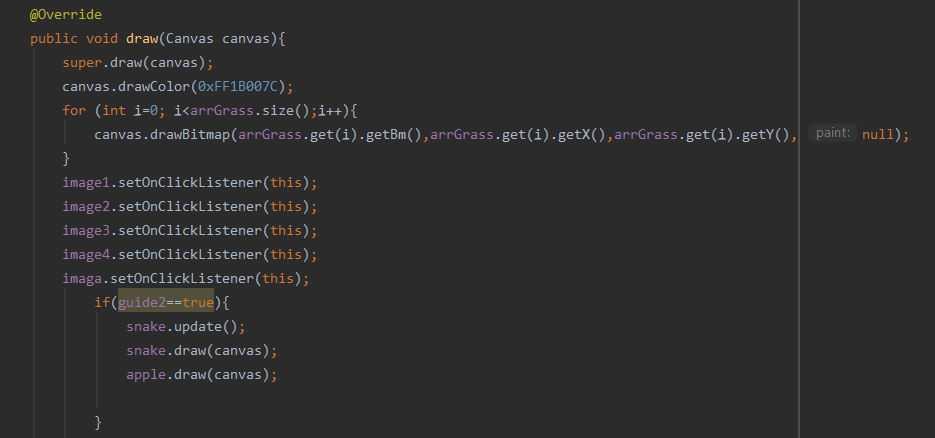


Рис. 2.7. Прорисовка поля, добавление звезд и змейки

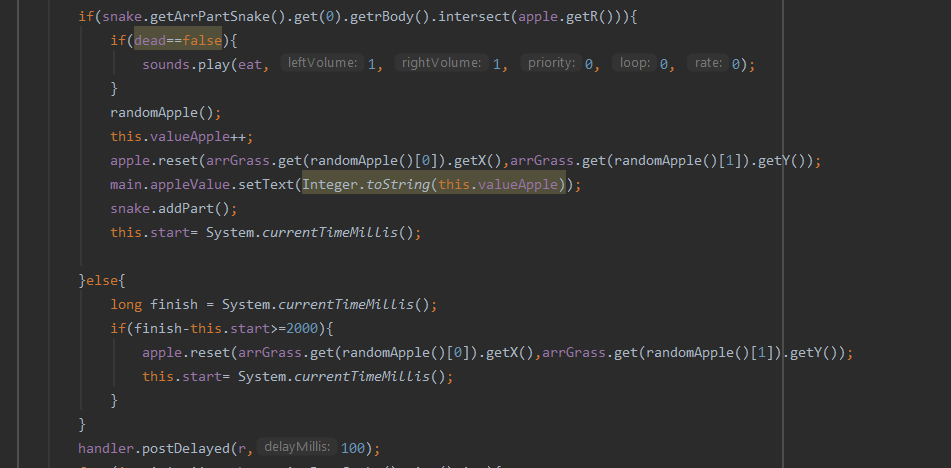


Рис. 2.8. Условия генерации звезд и подачи звука

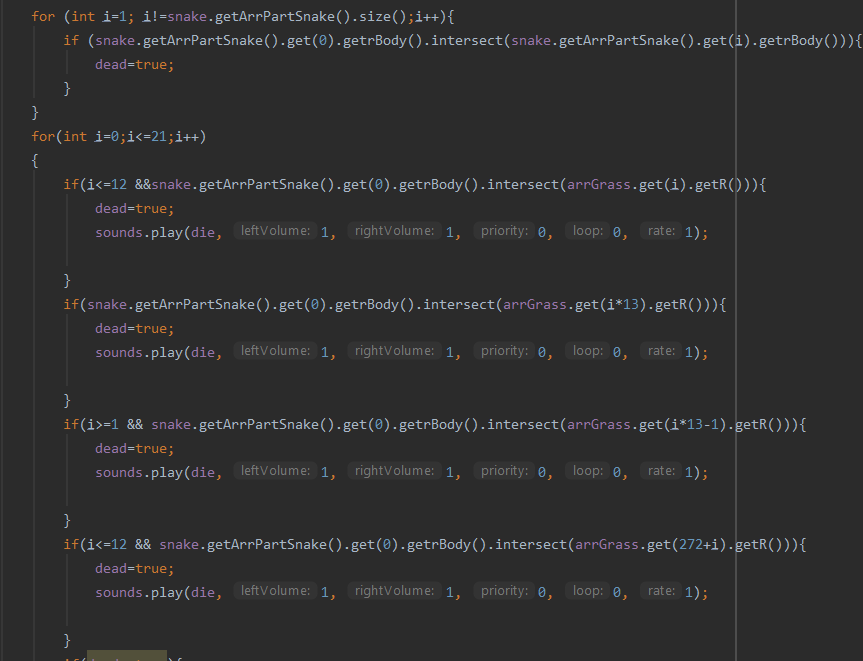
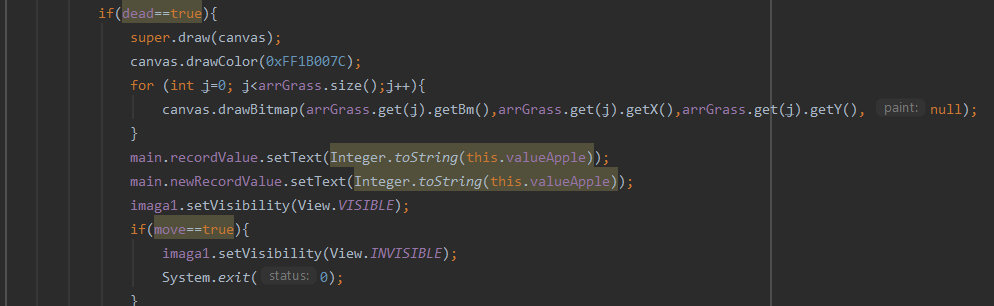


Рис. 2.9. Условия смерти змейки и подачи звука

Рис. 2.10. Реакция на смерть змейки и условие выхода из приложения

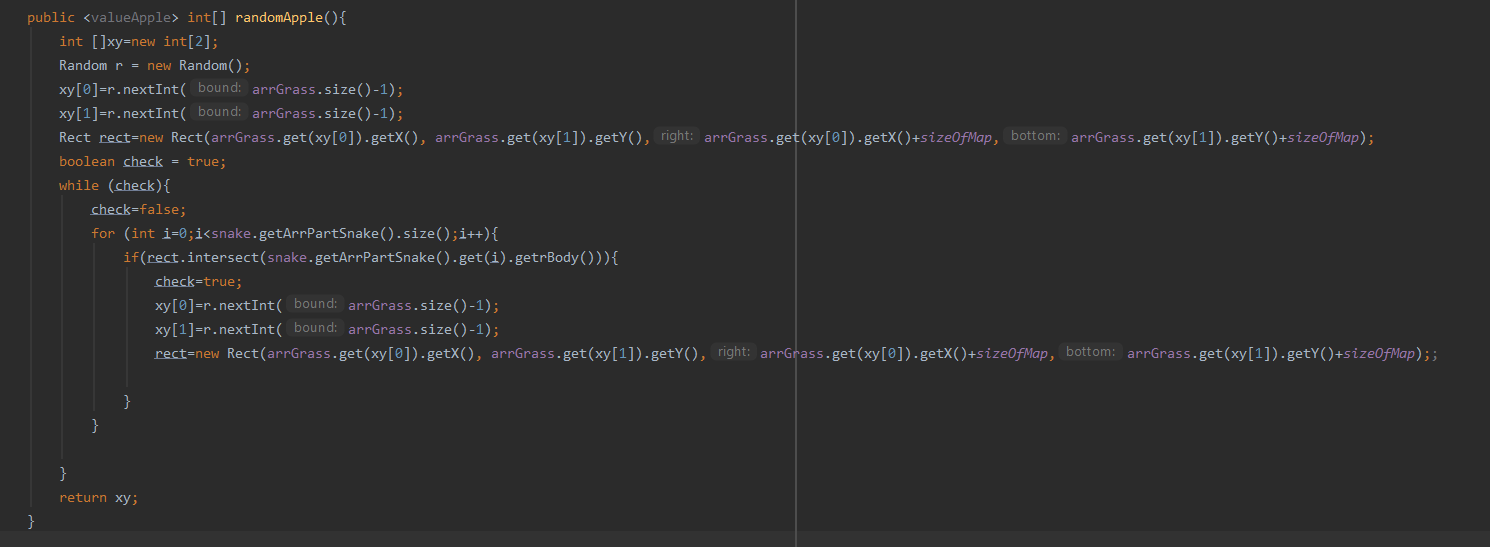


Рис. 2.11. Рандомайзер для звезд

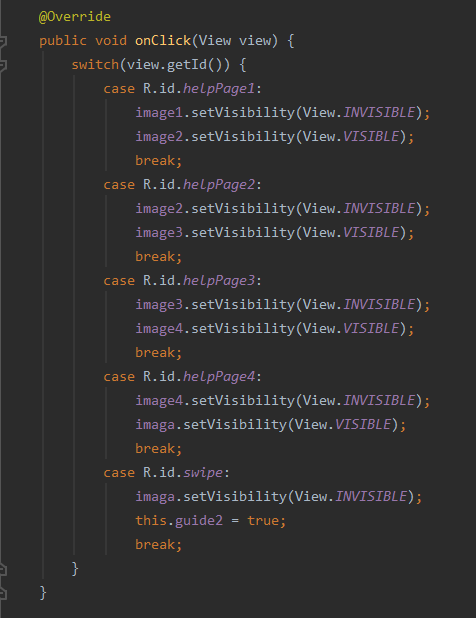


Рис. 2.12. Метод для пролистывания обучающего меню

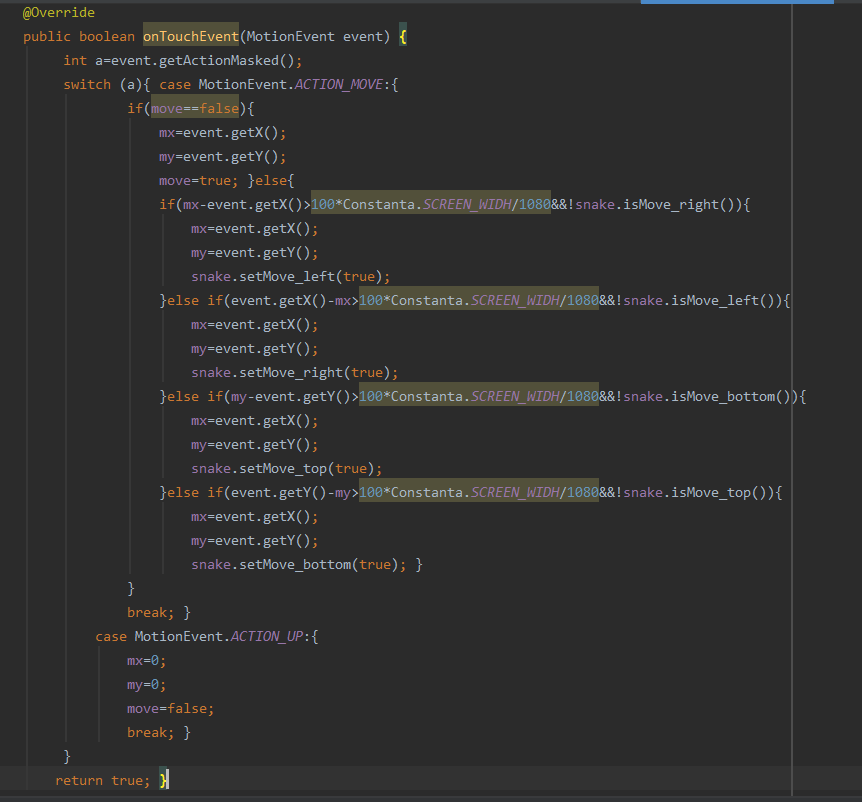


Рис. 2.13. Обработчик нажатий

1. Функционал приложения

Так как данная игра отличается от классической змейки некоторым набором правил, программа, перед началом игры, выводит пользователю обучающее меню, в котором говорится о 4х правилах:

* «Лови звезды и расти» - цель игры ловить звезды
* «Бойся стен» - стены в данной игре – не являются прямой границей карты. Они прилегают к ранице карты имея ширину в одну клетку.
* «Хитрые звезды» - звезды, как им и положено, мерцают. А значит они то появляются, то исчезают, меняя свое местоположение. Их «хитрость» заключается в особенности прятаться в тех самых невидимых стенах.
* «Не сотвори себе капкан» - змейка может умереть не только врезавшись в стену, но и в свой хвост.





Рис. 2.21. Обучающее меню Twilight star

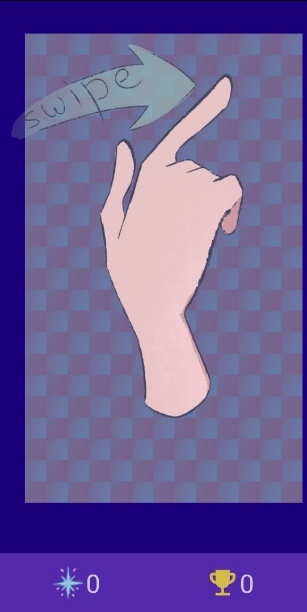


Рис. 2.21. Начальный экран перед началом самой игры, являющийся последней обучающейся частью, говорящей о стиле управления. (нужно не просто нажимать на экран, а проводить по нему пальцем, чтобы змейка поворачивалась)

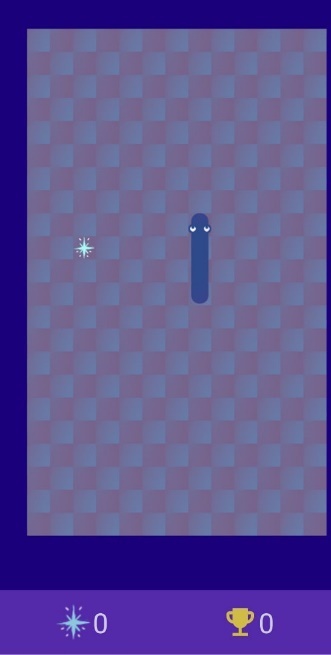


Рис. 2.22. Начало игры

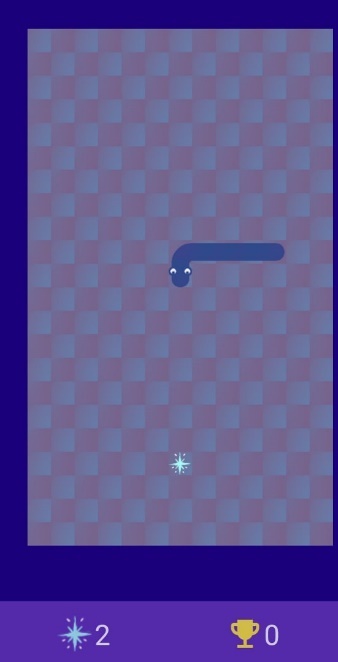


Рис. 2.23. Демонстрация роста змейки и смены положения звезды



Рис. 2.23. Экран окончания игры

1. Получаем готовое приложение, предназначенное для развлечения пользователей. Рекомендованное разрешение экрана HD (1080x1920: 420dpi). Минимальная версия Android – 10.0

Таким образом приложение «Twilight star» полностью закончено и готово к эксплуатации в реальных условиях.

Выводы по главе 2.

Во второй главе был проведен анализ и сравнение программ для разработки мобильных приложений, предоставлены код и скрины работающего приложения «Twilight star», описан его функционал и указаны минимальные требования к ресурсам

ЗАКЛЮЧЕНИЕ

В рамках работы было разработано мобильное приложение для Android. Для достижения данной цели были решены следующие задачи:

1. осуществлена постановка задачи, выделены требования к приложению;
2. изучены современные средства разработки мобильных приложений для Android;
3. определены требования и спроектировано мобильное приложение;

реализовано и протестировано мобильное приложение. Все поставленные задачи были решены, цель достигнута.

1. Разработанное приложение имеет перспективы дальнейшего развития. С учетом желаний и требований пользователей перспективы дальнейшего развития мобильного приложения могут быть следующими:

* добавление возможности добавления новых уровней
* добавление возможности настраивания сложности(скорости змейки после каждой съеденной звезды)
* добавление многопользовательского режима
* добавление листа рейтингов пользователя/пользователей
* добавление квестов

Приложение работает верно, за исключением сохранения прошлого рейтинга. В будущем планируется устранить данную проблему.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Арлоу Дж., Нейштадт А. «Разработка мобильных приложений. Практический объектно-ориентированный анализ и проектирование. 2-е издание.» – СПб.: Издательство «Питер», 2016.
2. Блох Джошуа «Java: эффективное программирование (3-е издание)» – Мск.: Издательство «Эксмо», 2019.

Бурнет Э. «Привет, Android!. 2-е издание». – М.: Издательство «Символ-Плюс», 2017.

1. Дарвин Ян Ф. «Android. Сборник рецептов». 2-е издание. – СПб.: Издательство «Питер», 2018 год.
2. Дейтел П., Дейтел Х., Уолд А. Android для разработчиков. 3-е издание. – СПб.: Издательство «Питер», 2016.
3. Колесниченко Д. М. «Программирование для Android 5» – Мск.: Издательство «Символ-Плюс», 2017.
4. Федотенко М.В. «Разработка мобильных приложений. Первые шаги». – СПб.: Издательство «Питер», 2017.
5. Маркотт Итан «Отзывчивый веб-дизайн» – Мск.: Издательство «Символ-Плюс», 2020.
6. Унгер Расс, Чендлер Кэролайн «UX-дизайн. Практическое руководство по проектированию опыта взаимодействия» – Мск.: Издательство «Эксмо», 2017
7. Филлипс Билл «Android. Программирование для профессионалов» – Мск.: Издательство «Питер», 2019.
8. Эккель Брюс «Философия Java (4-е полное издание)». – СПб.: Издательство «Питер», 2018.

Электронные-ресурсы

1. Архитектура Android-приложений. Часть II – архитектурные стили и шаблоны. [Электронный ресурс] URL: https://habrahabr.ru/post/140655/
2. Основы создания приложений. [Электронный ресурс] URL: https://developer.android.com/guide/components/fundamentals.html
3. Download Android Studio and SDK Tools | Android Studio. [Электронный ресурс] URL: https://developer.android.com/studio/index.html
4. Eclipse - The Eclipse Foundation open sourse community website.

[Электронный ресурс] URL: https://www.eclipse.org/downloads/

1. IntelliJ IDEA the Java IDE – JetBrains. [Электронный ресурс] URL:

https://www.jetbrains.com/idea/