Практическая работа №1

1. **Предметная область** – разработка мобильной игры-головоломки "Переливашка".
2. **Название ИС**– "Переливашка".
3. **Цель ИС** – создание мобильного приложения для платформы Android, которое предоставляет пользователям увлекательную и интерактивную головоломку, где необходимо переливать жидкости между сосудами для достижения цели.
4. **Анализ осуществимости ИС** – разработка игры осуществима, так как не требует сложных технологий или инновационных решений. Основные задачи включают создание игрового интерфейса, реализацию механики переливания и разработку уровней с постепенным увеличением сложности. Аналоги проекта:

|  |  |  |  |
| --- | --- | --- | --- |
| Приложение | Преимущества | Недостатки | Вывод |
| Water Sort Puzzle | - Простой и интуитивно понятный интерфейс.  - Большое количество уровней с постепенным увеличением сложности.  - Успокаивающая цветовая гамма и приятная анимация. | - Ограниченное количество жизней и необходимость ожидания для их восстановления.  - Иногда уровни могут быть слишком однообразными. | - Необходимо добавить разнообразие в уровни, чтобы игра не становилась монотонной.  - Убрать ограничения на количество попыток, чтобы игра была более доступной и менее раздражающей. |
| Learn Morse Code | - Лёгкое введение в основы.  - Практика перевода текста в сигналы. | - Нет вариативности в обучении.  - Нет функции воспроизведения вибраций или световых сигналов. | - Внедрить разнообразные способы практики (ввод текста, аудио, свет). - Добавить адаптивное обучение (разные уровни сложности). |
| Morse Trainer | - Фокус на тренировке скорости (CW-режим). | - Сложный для новичков интерфейс.  - Сосредоточен только на практической части, мало обучающего контента. | - Сделать обучение более дружественным для новичков. - Включить пошаговое обучение с простыми примерами. - Сбалансировать обучение и тренировку. |

**Таким образом проект должен содержать:**

* **Современный и удобный дизайн**: интуитивно понятный интерфейс с минималистичным дизайном и приятной цветовой гаммой.
* **Игровые элементы**: механика переливания жидкостей, уровни с постепенным увеличением сложности, система подсказок и возможность отмены ходов.
* **Персонализированное обучение**: адаптивная система уровней, которая подстраивается под навыки игрока.
* **Поддержка мультимодальных сигналов**: использование звуковых и визуальных эффектов для улучшения игрового опыта.
* **Баланс между обучением и практикой**: постепенное усложнение уровней, чтобы игроки могли развивать свои навыки.

1. Размещение и пользователи ИС – клиентская часть хранится на устройствах пользователей, а также в магазинах приложений. Данные, включая прогресс пользователя, хранятся локально. Серверная часть. Пользователями ИС являются:

* Любители головоломок и казуальных игр.
* Люди, которые хотят развивать логическое мышление и концентрацию.
* Образовательные учреждения, которые могут использовать игру для развития навыков решения задач у студентов.

1. **Комплекс технических средств ИТ:**

**Компьютерная техника:**

* Мощные компьютеры для разработки и тестирования.
* Тестовые устройства (смартфоны и планшеты на Android).
* Облачные серверы (опционально для хранения резервных копий или аналитики).

**Коммуникационная техника:**

* Интернет для публикации приложения в Google Play.
* Мобильные устройства для тестирования и использования.

**Организационная техника:**

* Проекторы для презентации проекта.
* Оргтехника для документирования и планирования.

**Оперативная полиграфия:**

* Принтеры для печати инструкций и рекламных материалов.
* ПО для дизайна (например, Adobe Photoshop, Figma) для создания графики и макетов.

1. **Системное ПО ИТ:**

**Клиентская часть:**

* ОС Android предоставляет инструменты для взаимодействия с сенсорами устройства и API для работы с графикой и звуком.

**Средства разработки:**

* Android SDK и Android Studio – основная платформа для разработки под Android.
* Язык программирования: Java или Kotlin.

**Средства тестирования:**

* VirtualBox для создания виртуальных машин на этапе разработки и тестирования.
* Эмуляторы Android для тестирования на различных устройствах.

**Системы обновления ПО:**

* Google Play Console предоставляет механизмы автоматического обновления приложений.

Основные особенности проекта "Переливашка":

* Интуитивно понятный интерфейс.
* Постепенное увеличение сложности уровней.
* Отсутствие ограничений на количество попыток.
* Минимальное количество рекламы.
* Возможность отмены ходов для большего удобства.
* Подсказки для сложных уровней.
* Яркая и привлекательная графика.

Практическая работа №2

|  |  |  |
| --- | --- | --- |
| № | Раздел | Содержание |
| 1 | Общие сведения | 1. Наименование системы: Мобильная игра-головоломка "Переливашка" (рабочее название: "Переливашка"). 2. Основание для разработки: Приложение создается на основании инициативы разработчика в целях удовлетворения потребностей пользователей, заинтересованных в увлекательных и развивающих головоломках. Игра направлена на развитие логического мышления, концентрации и навыков решения задач, а также предоставляет возможность расслабления и отдыха. 3. Наименование и условное обозначение темы разработки:    * Наименование: Разработка мобильной игры-головоломки "Переливашка".    * Условное обозначение: "Переливашка v1.0". 4. Разработчик и заказчик:    * Разработчик: Груздев Д.Д.    * Заказчик: Горелов А.С. 5. Плановые сроки начала и завершения работ:    * Начало разработки: 10.15.2024.    * Завершение разработки: 10.31.2024. |
| 2 | Назначение и цели | 1. Назначение системы: Игра "Переливашка" предназначена для пользователей, которые хотят развивать логическое мышление, концентрацию и навыки решения задач через увлекательную головоломку. Приложение предоставляет возможность переливать жидкости между сосудами, решая задачи различной сложности, а также включает элементы геймификации для повышения мотивации и вовлеченности игроков. 2. Цели разработки:  * Создать современное, удобное и доступное мобильное приложение для любителей головоломок. * Обеспечить кроссплатформенность (доступность на Android и iOS). * Включить элементы геймификации, такие как система достижений, рейтинги и награды, для повышения мотивации пользователей. * Внедрить функциональность адаптивного обучения, чтобы уровни подстраивались под навыки игрока. * Реализовать систему отслеживания прогресса, чтобы пользователи могли видеть свои успехи и улучшения. |
| 3 | Требования к системе | 1. Требования к функциональности:   Игровая механика:   * Переливание жидкостей между сосудами для достижения цели (например, сортировка по цветам). * Постепенное увеличение сложности уровней с добавлением новых элементов (например, трубок, воронок, ограниченного количества ходов).   Система геймификации:   * Уровни, достижения и награды за выполнение заданий. * Челленджи (например, пройти уровень за минимальное количество ходов).   Трекер прогресса:   * Фиксация достижений пользователя (пройденные уровни, количество ходов, время прохождения).   Подсказки и помощь:   * Возможность использовать подсказки для сложных уровней. * Отмена ходов для исправления ошибок.   Мультимодальные сигналы:   * Использование звуковых и визуальных эффектов для улучшения игрового опыта.  1. Требования к техническому исполнению:  * Платформы: Android (версии от 8.0 и выше) * Языки программирования: Kotlin/Java для Android.  1. Производительность:  * Загрузка интерфейса: не более 2 секунд. * Обработка пользовательского ввода: мгновенная реакция на действия игрока (например, переливание жидкостей).  1. Надежность:  * Сохранение данных: прогресс пользователя (пройденные уровни, достижения) сохраняется локально на устройстве. * Восстановление данных: возможность восстановления прогресса при переустановке приложения.  1. Условия эксплуатации:   Мобильные устройства:   * Минимальная оперативная память: 2 ГБ. * Поддержка устройств с разными размерами экранов (от 4,7 дюймов и выше).  1. Интерфейс:   Удобный и современный пользовательский интерфейс:   * Адаптация под разные размеры экранов. * Поддержка тёмной и светлой тем.   + Интуитивно понятное управление: * Простое взаимодействие с сосудами и жидкостями (тапы, свайпы).   Визуальные эффекты:   * Яркая и привлекательная графика. |
| 4 | Состав работ | 1. Разработка проектной документации. 2. Создание прототипа интерфейса. 3. Реализация клиентской части приложения. 4. Тестирование приложения (ручное и автоматизированное). 5. Публикация в Google Play Store и Apple App Store. 6. Поддержка и обновление приложения. |
| 5 | Порядок приемки | 1. Этапы контроля:    * Проверка прототипа интерфейса.    * Тестирование отдельных модулей (функциональное тестирование).    * Интеграционное тестирование.    * Финальное тестирование перед релизом. 2. Критерии приемки:    * Корректная работа всех функций, указанных в ТЗ.    * Полное соответствие дизайна утверждённому макету.    * Успешная публикация в магазинах приложений. |
| 6 | Требования к документации | 1. Руководство пользователя (на русском и английском языках). 2. Документация для разработчиков. |
| 7 | Дополнительные требования | 1. Поддержка локализации (русский, английский языки, с возможностью добавления других). 2. Возможность интеграции с внешними сервисами (например, авторизация через Google). 3. Планы по развитию приложения (внедрение новых функций на следующих этапах). |

Практическая работа №3

Задание №1.

Функциональные требования:

1. Обучение азбуке Морзе

* Отображение правил игры и механики (как переливать жидкости между сосудами);
* Аудио- и визуальная демонстрация процесса переливания (анимации, звуковые эффекты);
* Практические задания на решение различных уровней сложности.

1. Практика и тренировка

* Задачи по решению головоломок с переливанием жидкостей (разные комбинации и объемы);
* Игровые режимы для тренировки скорости и точности (викторины, челленджи);
* Возможность работы с реальными сценариями: создание собственных уровней и задач.

1. Система прогресса и геймификация

* Трекер прогресса пользователя (завершенные уровни, правильные решения);
* Награды за достижения (бэджи, уровни);
* Рейтинги и сравнение результатов с другими пользователями.

1. Функции управления и настройки

* Выбор режимов обучения (новичок, продвинутый, свободная практика).
* Настройки интерфейса (смена тем: тёмная, светлая).
* Регулировка сложности задач (для адаптации под уровень пользователя).
* Возможность отключения или настройки звука и анимаций.

1. Дополнительные функции

* Генератор уровней: создание пользовательских задач и их решение.
* Локализация: поддержка русского и английского языков (с возможностью добавления новых).

Эксплуатационные требования:

1. Производительность

* Загрузка основного интерфейса не должна превышать 2 секунд на большинстве поддерживаемых устройств.
* Воспроизведение звуков и анимаций должно происходить без задержек.
* Игровые действия и решения должны обрабатываться в реальном времени.

1. Надежность

* Локальное сохранение прогресса на устройстве.
* Резервное копирование данных в облаке с возможностью восстановления.
* Приложение должно корректно завершать работу в случае непредвиденных ошибок, с возможностью восстановления после перезапуска.

1. Удобство

* Поддержка устройств с диагональю экрана от 4 до 12 дюймов (смартфоны, планшеты).
* Простота навигации: интуитивно понятный интерфейс, минимальное количество действий для запуска игры.
* Голосовая и визуальная обратная связь при выполнении действий пользователем.

1. Адаптивность

* Автоматическая адаптация интерфейса под размеры экрана устройства.
* Адаптивное обучение: для новичков — упрощенные задания с подсказками, для опытных пользователей — повышение уровня сложности.

1. Совместимость

* Совместимость с ОС Android (версии 8.0 и выше).
* Поддержка сенсорных экранов и мультимедийных функций устройств.
* Работа с устаревшими устройствами (минимальная оперативная память — 2 ГБ).

Эскизный проект

|  |
| --- |
| **Методы решения и выбор языка программирования**  1. **Платформа разработки**    * **Инструмент:** Android Studio — официальная среда разработки для Android-приложений, предоставляющая удобный инструментальный набор и эмуляторы для тестирования.    * **Тип приложения:** Нативное приложение (под Android). 2. **Язык программирования**    * **Kotlin:** Современный язык разработки для Android с лаконичным синтаксисом. Полная поддержка Android API. Удобная работа с многопоточностью (например, для работы с анимациями и звуком). Высокая производительность и безопасность (Null-Safe).   Спецификация процессов   1. Основные функциональные блоки 2. Обучение: Интерактивные уроки по основам игры "Переливашка". Визуализация различных уровней сложности и правил игры. Режим практики: возможность играть с подсказками и без них. 3. Геймификация: Викторины и мини-игры для проверки знаний о правилах и механике игры. Награды и достижения за успешное выполнение заданий. Трекер прогресса (количество завершенных уровней, время на выполнение задач). 4. Настройки: Регулировка сложности игры (количество стаканов, объем жидкости). Переключение звуковых эффектов и анимаций. Управление профилем (сброс прогресса, изменение имени пользователя). 5. Работа с мультимедиа: Воспроизведение звуковых эффектов (наливание, переливание). Анимация стаканов и жидкости для визуализации процесса. Вибрация для тактильной обратной связи при успешном или неуспешном выполнении задания. 6. Архитектурный подход: приложение использует компонентную архитектуру с внедрением зависимостей (DI) для упрощения поддержки и тестирования: 7. Model: бизнес-логика, обработка данных .    * ViewModel: управление состоянием данных для интерфейса.    * View: отображение данных и обработка взаимодействия пользователя. 8. Основные процессы  * Игровой процесс: Игрок выбирает сосуд и переливает жидкость в другой сосуд. Приложение симулирует физику переливания (например, уровень жидкости изменяется в зависимости от объёма).Игрок достигает цели уровня (например, сортировка жидкостей по цветам).Система проверяет выполнение условий уровня и начисляет очки. * Система событий: случайно генерируются игровые события (например, добавление новых сосудов, изменение цвета жидкостей, ограничение количества ходов). Приложение применяет изменения в игровом поле (например, добавляет новые элементы или усложняет условия). Игрок адаптируется к изменениям и ищет новые стратегии для решения задачи. Очки начисляются с учётом сложности ситуации (например, бонусы за прохождение уровня за минимальное количество ходов). * Сохранение прогресса: при паузе или завершении игры приложение сохраняет состояние в локальном хранилище (прогресс уровней, количество очков, достижения). При повторном запуске игры пользователь может продолжить с того же места.  1. Диаграмма структуры процессов – отображает основные потоки данных между компонентами приложения.      1. Навигационная диаграмма – отображает переходы между экранами приложения |

Задание №2.

1. Интерфейс пользователя (UI):

* Главное меню с выбором уровней
* Игровое поле с сосудами
* Отображение объема жидкости в каждом сосуде
* Индикатор прогресса/ходов
* Кнопки управления (отмена хода, рестарт)

1. Модуль игровой логики:

* Управление состоянием сосудов
* Расчет возможных переливаний
* Проверка достижения цели уровня
* Подсчет количества ходов
* Сохранение/загрузка состояния игры

1. Модуль уровней:

* Генерация начальных состояний сосудов
* Хранение целевых состояний
* Проверка решаемости уровня
* Система прогрессии сложности

1. Модуль анимации:

* Визуальные эффекты при успешном завершении
* Анимация заполнения/опустошения сосудов

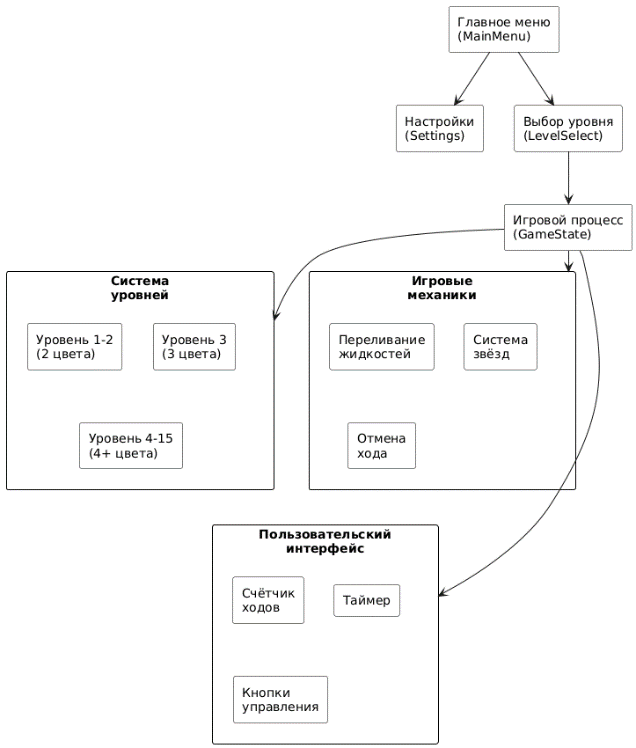
1. Модуль настроек:

* Настройка сложности
* Включение/выключение звуков
* Сброс прогресса игры
* Выбор визуальной темы

1. Система достижений:

* Отслеживание рекордов по уровням
* Подсчет общего прогресса
* Разблокировка новых уровней
* Статистика игрока

1. Структурная карта Константайна – отображает иерархическую организацию экранов и модулей приложения. Она помогает визуализировать функциональную структуру и показать, как пользователь взаимодействует с системой.



1. Структурная карта Джексона – предназначена для отображения иерархической структуры процессов приложения, включая порядок выполнения операций. Этот подход акцентирует внимание на последовательности действий и взаимосвязях между компонентами системы.



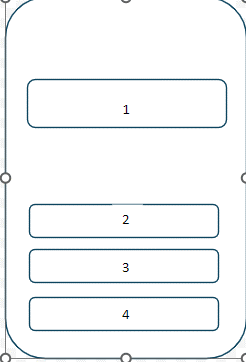
Практическая работа №4

Задание №1.

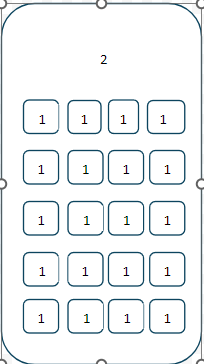
Логотип приложения:

Макеты приложения:

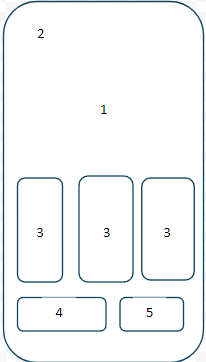
* Главное меню



* Экран выбора уровня



* Экран игры



Лабораторная работа №1

Задание №1 – Диаграмма использования.

Диаграмма вариантов использования (Use Case Diagram) — это визуальное представление взаимодействия пользователей (акторов) с системой. Она помогает определить ключевые функции приложения, связи между ними и их зависимость друг от друга. В данном проекте диаграмма построена с учетом ассоциаций, включения (include), расширения (extend) и наследования (generalization).

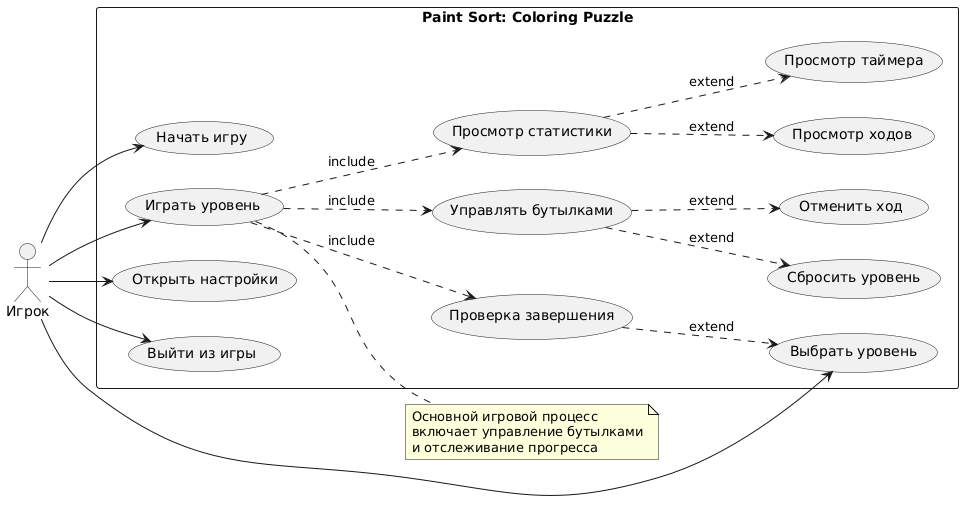
Акторы представляют собой внешние сущности, взаимодействующие с системой. В данной диаграмме он один – Пользователь (основной актор). Это конечный пользователь приложения, который хочет изучить азбуку Морзе, практиковаться, проходить викторины, отслеживать прогресс и настраивать приложение. Он является центральной фигурой приложения, так как все функции созданы именно для его взаимодействия с системой.

Основные функции приложения представлены отдельными вариантами использования (Use Cases). Они являются ключевыми модулями системы, необходимыми для полноценной работы приложения.

* Изучение азбуки Морзе – Это основной модуль обучения, где пользователь знакомится с теорией и проходит практические упражнения. Без этого компонента приложение теряет свою образовательную ценность. Система подмодулей позволяет более гибко структурировать процесс обучения. Включает (include): Просмотр теории – пользователю нужно сначала ознакомиться с азбукой Морзе; Практика перевода – обязательный этап обучения, где пользователь применяет знания. Расширяет (extend): Изучение сложных уровней – доступно только после освоения базового уровня.
* Прохождение викторины – Этот вариант использования нужен для проверки знаний пользователя. Он включает в себя два варианта заданий: Распознавание сигналов – пользователь слышит (или видит) сигнал и должен определить букву; Генерация сообщений – пользователь должен перевести текст в азбуку Морзе или наоборот. Обучение без проверки знаний неэффективно, поэтому викторина служит способом оценки прогресса. Эти модули являются необходимыми частями процесса прохождения викторины. Также происходит наследование от Изучение азбуки Морзе, так как пользователь должен сначала изучить теорию, чтобы успешно проходить викторину.
* Просмотр прогресса – В этом разделе пользователь может анализировать свои достижения. Разделен на три подмодуля: Просмотр достижений – показывает награды и уровни; Просмотр статистики – количественные показатели успехов пользователя; Просмотр рейтинга – положение пользователя среди других.
* Настройка приложения – Этот модуль предоставляет пользователю возможность изменить параметры приложения. Включает в себя три функции: Управление сигналами – настройка звука, вибрации и визуальных эффектов; Выбор темы интерфейса – кастомизация внешнего вида приложения; Сброс прогресса – возможность начать обучение заново. Персонализация интерфейса и управление функционалом делают использование приложения удобнее.

В данной диаграмме вариантов использования тщательно продуманы следующие аспекты:

* Все ключевые функции приложения выделены в отдельные use cases.
* Логические связи между ними учитывают зависимость действий пользователя (например, без изучения теории невозможно пройти викторину).
* Использованы различные виды связей (include, extend, generalization), чтобы показать модульную структуру и зависимость функций.
* Диаграмма помогает разработчику понять, как должен работать пользовательский интерфейс и какие обязательные этапы предусмотрены в логике приложения.



Задание №2 – Диаграмма последовательностей.

Диаграмма последовательностей (Sequence Diagram) предназначена для отображения динамического взаимодействия между пользователем и системой в процессе изучения азбуки Морзе. Данный тип диаграммы иллюстрирует порядок вызова операций, передачу сообщений и зависимости между объектами системы.

Основные цели диаграммы:

* Определение последовательности действий, которые выполняет пользователь при работе с приложением.
* Демонстрация обмена сообщениями между компонентами системы.
* Выявление ключевых операций и их взаимосвязей.
* Подготовка к реализации логики взаимодействия компонентов в коде.

В диаграмме присутствуют следующие акторы и объекты:

|  |  |  |
| --- | --- | --- |
| Элемент | Тип | Назначение |
| Пользователь | Актор | Представляет конечного пользователя, который взаимодействует с приложением. |
| Главный экран | Объект | Основной интерфейс приложения, с которого начинается работа. |
| Раздел обучения | Объект | Интерфейс, отвечающий за доступ к учебным материалам и упражнениям. |
| Теория | Объект | Раздел с теоретической информацией по азбуке Морзе. |
| Практика | Объект | Раздел с интерактивными упражнениями для закрепления материала. |

Диаграмма отражает следующий процесс:

1. Запуск приложения:

* Пользователь открывает приложение, инициируя взаимодействие с системой.
* Главный экран загружается и отображает интерфейс.
* Данный шаг необходим, так как без него невозможно продолжить выполнение дальнейших операций.

1. Переход в раздел "Обучение":

* Пользователь выбирает соответствующий раздел на главном экране.
* Главный экран передает запрос на открытие меню обучения.
* Система загружает и отображает список учебных материалов.

1. Изучение теоретического материала:

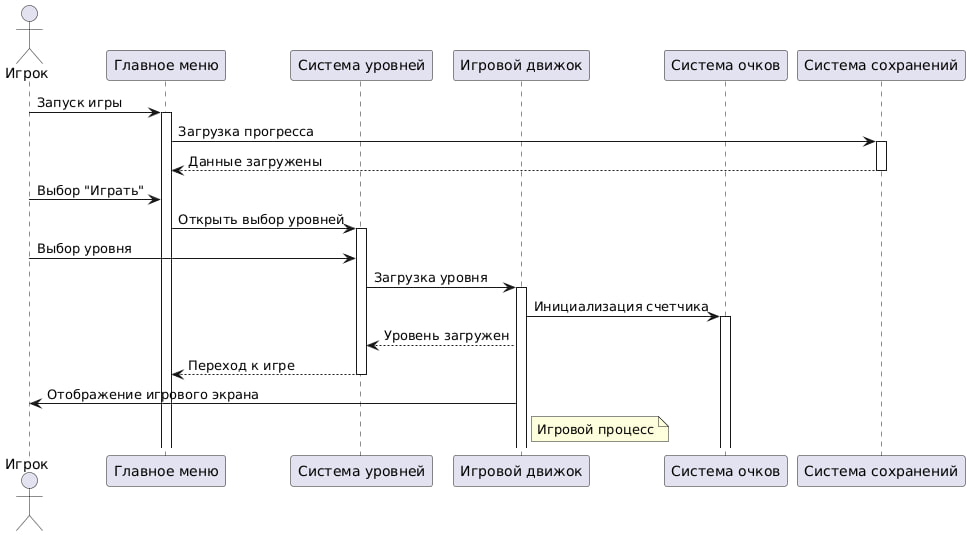
* Пользователь выбирает просмотр теории.
* Раздел обучения передает запрос на модуль "Теория".
* Загружаются учебные материалы, которые затем отображаются пользователю.
* По завершении просмотра пользователь выходит из раздела, и управление возвращается в меню обучения.

1. Переход к практическим упражнениям:

* Пользователь выбирает "Практика".
* Раздел обучения вызывает модуль "Практика".
* Пользователь выполняет упражнение (например, переводит текст в сигналы).
* По завершении задания система оценивает результат и предоставляет обратную связь.

1. Завершение работы:

* Пользователь получает результат.
* При необходимости возвращается в раздел обучения или выходит из приложения.



Лабораторная работа №2

Задание №1 – Диаграмма кооперации.

Диаграмма кооперации представляет собой один из видов диаграмм взаимодействий, используемых в объектно-ориентированном моделировании. Она демонстрирует, каким образом объекты системы взаимодействуют между собой в рамках определенного сценария, а также последовательность передачи сообщений.

Целью данной диаграммы является описание процесса выполнения пользователем практического упражнения в мобильном приложении для изучения азбуки Морзе. Диаграмма демонстрирует последовательность шагов, участвующие в процессе объекты и связи между ними.

Диаграмма кооперации включает следующие объекты:

1. Пользователь – инициатор взаимодействия, который выполняет действия в приложении, такие как выбор режима практики, ввод ответа и получение обратной связи.
2. Главный экран – точка входа в приложение, с которой начинается взаимодействие пользователя с системой. Этот объект обеспечивает навигацию к разделу "Обучение".
3. Раздел обучения – отвечает за выбор режима обучения и передачу управления в нужный модуль, в данном случае – в практику.
4. Практика – ключевой объект в данном сценарии, содержащий логику выполнения упражнения. Он управляет отображением заданий, принимает пользовательский ввод и передает его на проверку.
5. Проверка результата – объект, ответственный за обработку введенного пользователем ответа и его сравнение с эталоном.

Последовательность взаимодействий отражает процесс выполнения упражнения:

Этап 1: Запуск сценария

* Шаг 1: Пользователь инициирует взаимодействие с системой, выбирая в главном меню пункт "Обучение".
* Главный экран является центральным элементом навигации, с которого пользователь получает доступ к различным разделам. Без этого шага сценарий не сможет начаться.

Этап 2: Переход к практике

* Шаг 2: Главный экран передает управление объекту "Раздел обучения", который предлагает пользователю выбрать режим (теория, практика и т. д.).
* Шаг 3: Пользователь выбирает "Практику", после чего система загружает упражнение и передает управление соответствующему объекту.
* Разделение логики на несколько экранов обеспечивает гибкость системы и масштабируемость приложения. Это позволяет легко добавлять новые режимы обучения.

Этап 3: Выполнение упражнения

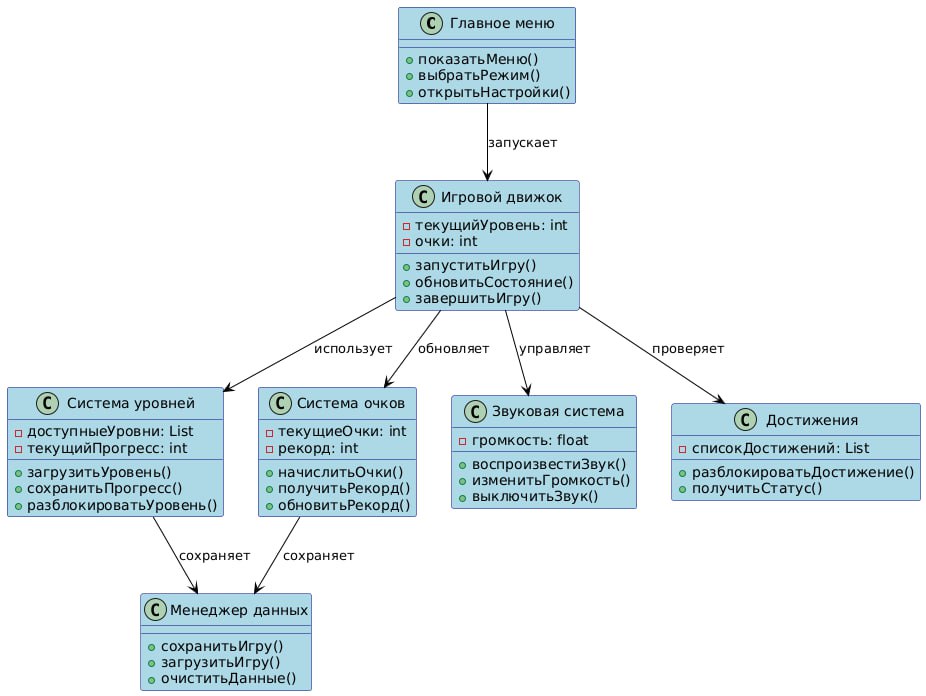
* Шаг 4: Объект "Практика" загружает задание и отображает его пользователю.
* Шаг 5: Пользователь вводит свой ответ.
* Данный шаг фиксирует действие пользователя, являясь критически важным для сценария. Без него невозможно проверить знание пользователя.

Этап 4: Проверка результата

* Шаг 6: Объект "Практика" передает введенный ответ в "Проверку результата".
* Шаг 7: Объект "Проверка результата" анализирует введенный ответ и сравнивает его с эталоном, после чего возвращает оценку точности.
* Зачем этот элемент? Проверка знаний – основная цель сценария. Обособление этого процесса в отдельный объект позволяет легко изменять алгоритмы проверки и расширять систему.

Этап 5: Завершение сценария

* Шаг 8: После проверки объект "Практика" передает пользователю обратную связь (правильный ли ответ, допущенные ошибки и т. д.).
* Зачем этот элемент? Этот этап завершает сценарий и предоставляет пользователю информацию о его успехах, что важно для обучения.



Задание №2 – Диаграмма развертывания.

Диаграмма развертывания представляет собой блок-схему процесса установки и начала использования мобильного приложения для изучения азбуки Морзе. Она демонстрирует последовательность действий пользователя — от поиска приложения в магазине до начала обучения. Диаграмма построена по линейной структуре, где каждый последующий этап логически вытекает из предыдущего.

1. Начальный этап: «Пользователь заходит в магазин приложений для Android»

* Магазин приложений (например, Google Play) является основным источником получения мобильных приложений на платформе Android.
* Пользователь начинает взаимодействие с системой именно с поиска приложения в этом магазине.

1. Действие: «Пользователь вводит в поисковую строку название приложения»

* Пользователь должен найти нужное приложение среди множества других.

1. Действие: «Пользователь скачивает и устанавливает установочный пакет»

* Любое приложение перед использованием должно быть загружено и установлено.
* Этот этап преобразует программное обеспечение из доступного в магазине в установленное на устройстве.

1. Действие: «Пользователь заходит в приложение»

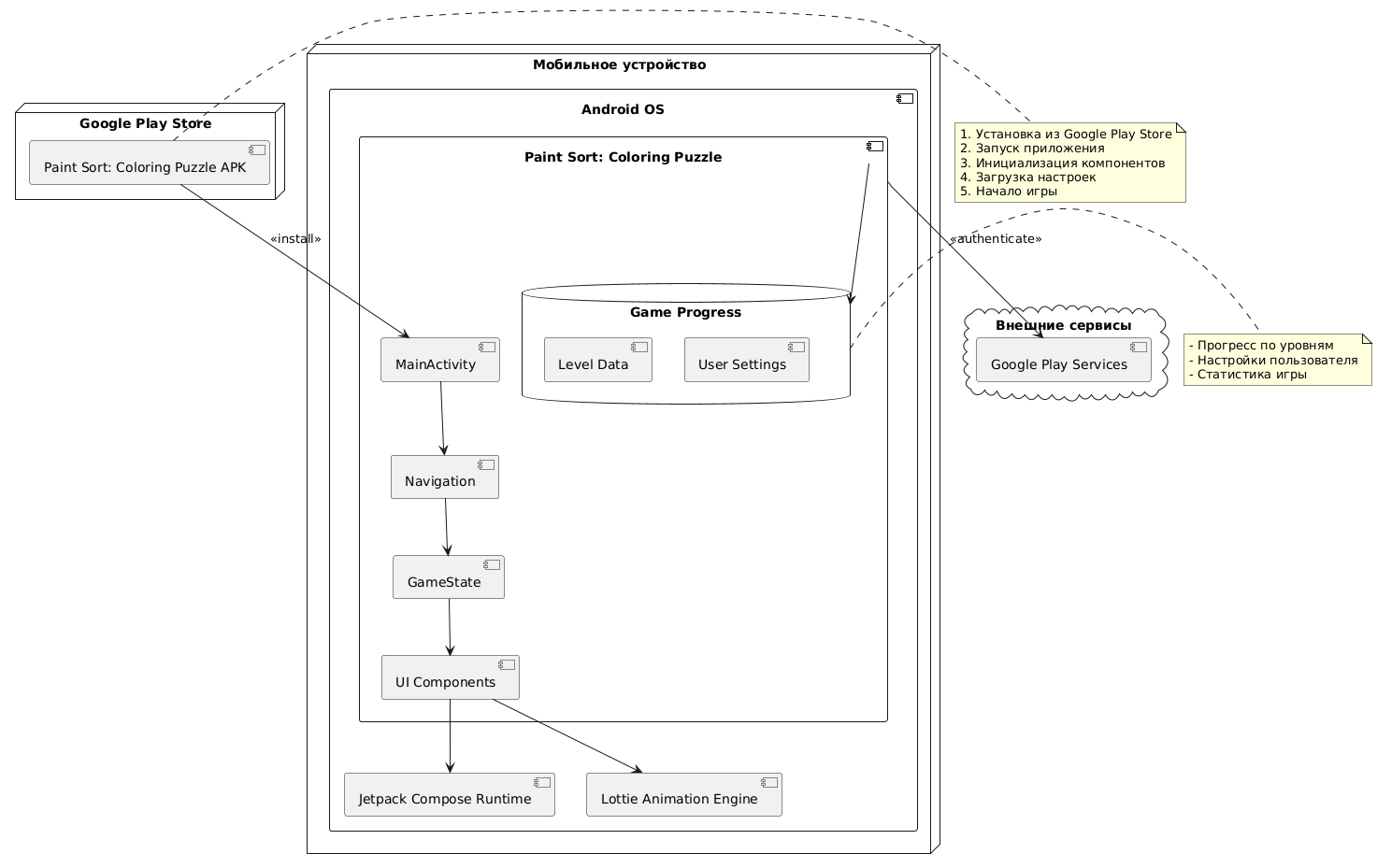
* После установки необходимо запустить приложение, чтобы начать взаимодействие с ним.

1. Действие: «Пользователь настраивает приложение в меню настроек»

* Для более удобного использования пользователь может захотеть изменить настройки (например, выбрать язык, включить звуковые сигналы или вибрацию).
* Некоторые настройки могут быть критически важны для полноценной работы приложения.

1. Завершающий этап: «Пользователь начинает обучение»

* Это конечная цель пользователя — изучение азбуки Морзе.
* Все предыдущие шаги направлены на то, чтобы обеспечить этот результат.



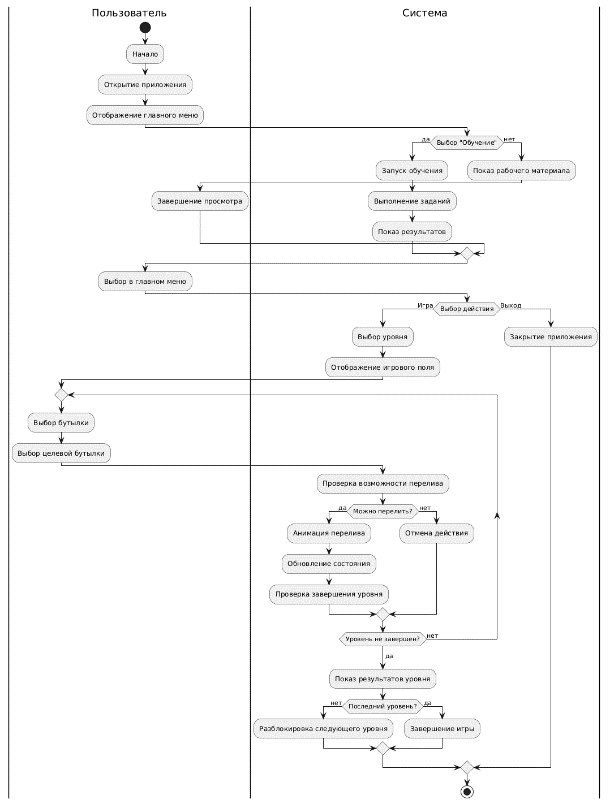
Лабораторная работа №3

Задание – Диаграмма деятельности.

Диаграмма деятельности представляет собой графическое описание последовательности действий пользователя в рамках функционала приложения, направленного на изучение азбуки Морзе. Данная диаграмма отображает логику взаимодействия пользователя с системой, возможные варианты развития событий и принятия решений.

Сценарий: "Процесс обучения пользователя азбуке Морзе"

* Пользователь открывает приложение.
* Выбирает раздел "Обучение".
* Выбирает либо "Теорию", либо "Практику".
* Если выбрана "Теория", отображается учебный материал.
* Если выбрана "Практика", запускается упражнение:Пользователь выполняет задание, после чего получает обратную связь.
* Пользователь может повторить обучение или выйти в главное меню.



Лабораторная работа №4

Задание №1 – Диаграмма состояний.

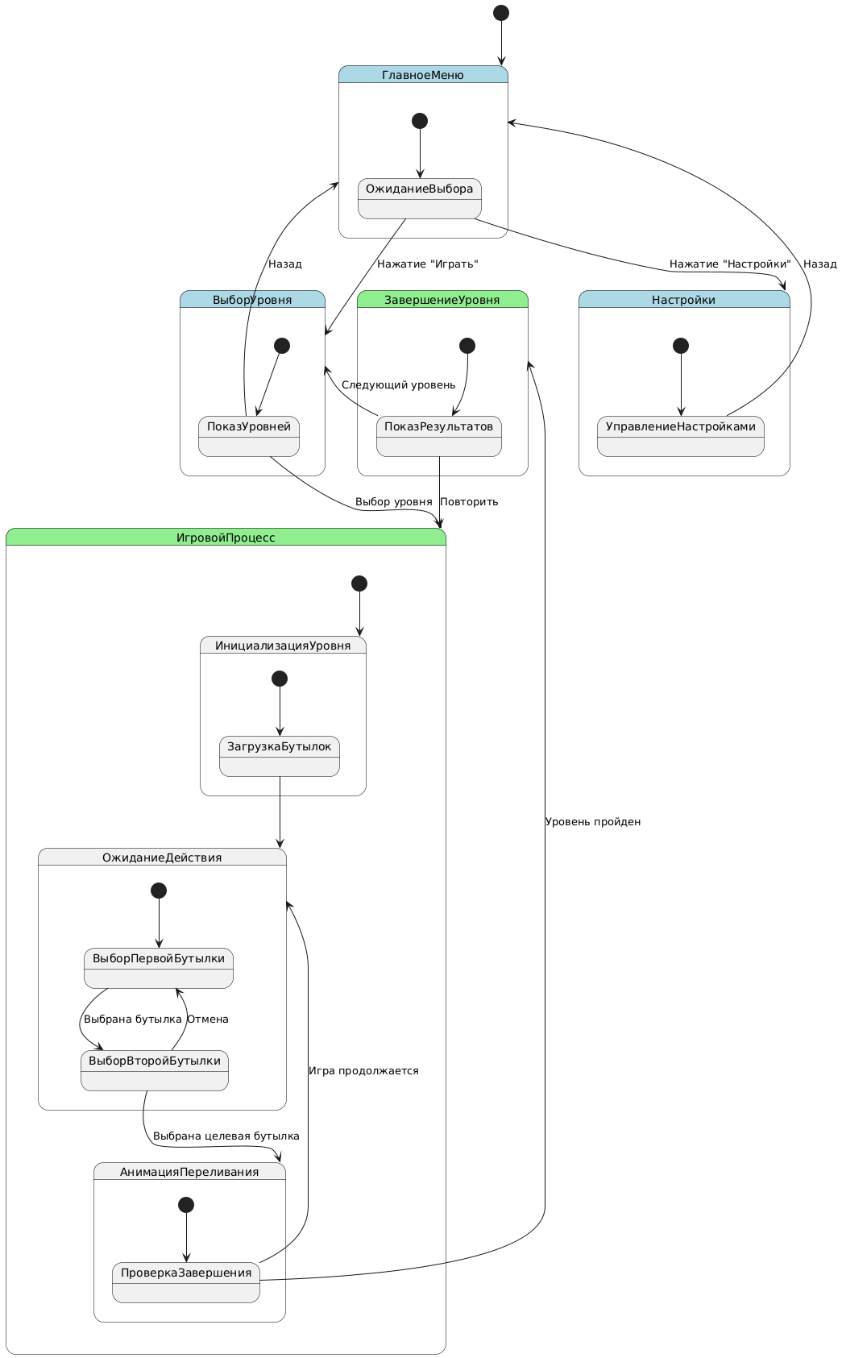
Диаграмма состояний моделирует поведение мобильного приложения для изучения азбуки Морзе в терминах различных состояний системы и переходов между ними. Она демонстрирует, как пользователь взаимодействует с системой и как приложение реагирует на его действия.

Диаграмма включает следующие основные компоненты:

* Состояния – различные режимы работы приложения, в которых оно может находиться.
* Переходы – связи между состояниями, определяющие возможные изменения в ходе работы приложения.
* Начальное и конечное состояния – точки старта и завершения работы системы.

Переходы обозначают действия пользователя, которые ведут к изменению состояния системы.

* Запуск приложения –> Главный экран – приложение загружено, и пользователь видит основное меню.
* Главный экран –> Обучение – пользователь решил изучить азбуку Морзе.
* Обучение –> Просмотр теории – пользователь выбрал теоретическую часть.
* Обучение –> Практика – переход к выполнению заданий.
* Главный экран –> Викторина – пользователь проходит тестирование.
* Прохождение теста –> Просмотр результата – завершен тест, отображаются результаты.
* Главный экран –> Прогресс – пользователь анализирует статистику.
* Прогресс –> Просмотр достижений – пользователь хочет видеть свои достижения.
* Прогресс –> Просмотр статистики – анализирует статистику обучения.
* Главный экран –> Настройки – пользователь решил изменить параметры приложения.
* Настройки –> Изменение параметров – внесение изменений в настройки.
* Главный экран –> Выход из приложения → Конец – пользователь завершает работу с программой.



Задание №2 – Диаграмма классов

Диаграмма классов иллюстрирует архитектуру классов Android-приложения с использованием AppCompatActivity в качестве базового класса. Она показывает взаимосвязи между активностями (Activity), их XML-разметками и ресурсными файлами (colors.xml, strings.xml, style.xml).

1. AppCompatActivity (Базовый класс)

* AppCompatActivity является суперклассом для всех Activity в Android-приложении.
* Он предоставляет базовый функционал, включая поддержку тем, взаимодействие с фрагментами и работу с ActionBar.

1. Активности приложения (Activity) – каждая активность отвечает за отдельный функциональный модуль приложения.

* ExploreActivity – экран исследования/обучения.
* MainActivity – основной экран, вероятно, главное меню приложения.
* PlayActivity – экран с игровыми или практическими элементами.
* ReceivingActivity – экран приема сигналов Морзе.
* SendingActivity – экран отправки сигналов Морзе.

1. XML Layout-файлы – каждая активность связана со своим XML-файлом, который определяет пользовательский интерфейс. Такое разделение логики (Activity) и интерфейса (XML) делает код более гибким и удобным для модификации:

* ExploreActivity –> XML Layout ExploreActivity
* MainActivity –> XML Layout MainActivity
* PlayActivity –> XML Layout PlayActivity
* ReceivingActivity –> XML Layout ReceivingActivity
* SendingActivity –> XML Layout SendingActivity

1. Общие ресурсные файлы – диаграмма также показывает связь интерфейса с тремя ресурсными файлами. Эти файлы позволяют централизованно управлять цветами, текстами и стилями:

* colors.xml – содержит цветовые схемы приложения.
* strings.xml – хранит текстовые строки, используемые в UI.
* style.xml – задает стили и темы для приложения.

