**Введение**

На учебную практику по ТРПО была поставлена задача, разработать игровое приложение в жанре 2D-Runner на тему: «Пустынные бега».

Цель учебной практики заключается в знакомстве массового пользователя с игровым приложением, его свойств, вниманием и терпением пользователя.

Создаваемое приложение будет рассчитано на любого рода пользователей. Применить данное приложение смогут не только люди, которые хотят проверить свои навыки, но и просто люди, заинтересовавшиеся в данной теме.

Далее приведем краткое описание разделов пояснительной записки.

Первый раздел носит название “Анализ задачи”. В нем вы сможете ознакомиться с постановкой задачи, которая включает в себя: исследование предметной области поставленной задачи, определение ее организационно-экономической сущности. Также в этом разделе вы сможете узнать о том, как данная задача решается в настоящее время. Все входные и выходные данные тоже будут описаны в первом разделе. В подразделе “Инструменты разработки” будет рассмотрена среда, в которой создается данное игровое приложение. Здесь также будут установлены минимальные и оптимальные требования к аппаратным характеристикам, обеспечивающим правильное функционирование поставленной задачей.

В разделе “Проектирование задачи” будут рассмотрены основные аспекты разработки игрового приложения. Здесь можно будет узнать об организации данных в контексте среды разработки. В данном разделе будет четко описан пользовательский интерфейс, составлены алгоритмы процесса обработки информации, описана разработка системы справочной информации.

“Реализация задачи” – это третий раздел пояснительной записки, в котором описываются все элементы и объекты, которые будут использованы при реализации данного приложения. В этом разделе будут четко описаны функции пользователя и их структура. Здесь можно будет найти таблицу, в которой будет представлена полная аннотация файлов используемых в данном проекте.

Четвертый раздел – “Тестирование”. В нем будет описано полное и функциональное тестирование данной программы, т.е. будет оттестирован каждый пункт меню, каждая операция, которая выполняется приложением. Будут смоделированы все возможные действия пользователя при работе с программой, начиная от запуска до выхода.

В разделе “Применение” будет описано назначение, область применения, среда функционирования игрового приложения. Также в нем будет описано использование справочной системы.

“Заключение” будет содержать краткую формулировку задачи, результаты проделанной работы, описание использованных методов и средств, описание степени автоматизации процессов на различных этапах разработки.

В разделе “Список используемых источников” будет приведен список используемых при разработке источников.

В приложениях к пояснительной записке будет приведен листинг программы с необходимыми комментариями и диаграммы.

**1 Анализ задачи**

* 1. **Постановка задачи**

**1.1.1 Организационно-экономическая сущность задачи**

Темой данного проекта является разработка игрового приложения «Пустынные бега».

Целью разработки данного программного продукта является создание игрового приложения 2D-Runner.

Данное игровое приложение предназначен для людей любого возраста.

Данный программный продукт может быть использован в любое время, не зависимо от того нуждается ли пользователь в данной продукции или он просто хочет ознакомиться с данной предметной областью.

Источники и способы получения данных для данного продукта отсутствуют.

Игровое приложение имеет цифровой аналог в виде браузерной игры.

Sand Racing (Пустынные бега) – Dino Runner;

Игровое приложение является пародией на представленный аналог с изменённой графикой и реализацией.

**1.1.2 Функциональные требования**

Sand Racing(Пустынные бега): геймплей заключается в прохождении забега. Целью игрока заключается в прохождении забега на рекордное расстояние или фиксированного с преодолением препятствий за неограниченное время. За каждый пройденный 1 игровой метр начисляется рекорд и увеличивается скорость игры в несколько раз для сложности. Игроку предоставляются 1 жизни в фиксированных забегах, если же игрок потеряет их, то он начинает заново, если потеряет 1 жизнь, продолжает с того же расстояния. Если же игрок натыкается в препятствие, то игрок умирает, что в последствии приведёт к проигрышу. Игра является бесконечной, а всё набранное игроком расстояние являются показателем рекордного значения. Игра завершается, при проигрыше игрока.

К поставленной задаче были заявлены следующие функциональные требования, который может выполнять пользователь:

- выбор режима игры из списка(уровни или забег);

- играть в выбранный режим;

- просмотр рекорда;

- настройки звука и изображения;

- просмотр информации о разработчике.

**1.1.3 Описание входной, выходной и условно постоянной информации**

Таблица 1 – Функции программы с описание с входной, выходной и условно-постоянной информации

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Категория пользоват  елей | Наименова  ние процесса | Краткое описание алгоритма выполнения процесса | Входная информация | Выходная информация | Условно–постоянная информация |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | Пользователь | выбор режима игры из списка(уровни или забег) | Пользователь может выбрать режим игры | Отсутствует | Отсутствует | Предоставленный доступ к режимам |
| 2 | Пользователь | играть в выбранный режим | Пользователь может играть в выбранный режим игры | Отсутствует | Отсутствует | Отсутствует |
| 3 | Пользователь | просмотр рекорда | Пользователь может просмотреть рекорд в главном меню | Отсутствует | Информация о рекорде | Рекорд |
| 4 | Пользователь | настройки звука/изображения | Изменение звука и изображения | Отсутствует | Изменение значений настроек | Значение громкости, формат разрешения |
| 5 | Пользователь | просмотр информации о разработчике | Пользователь может узнать информацию о разработчике | Отсутствует | Перечень информации о игре | Информация о разработчике |

**1.1.4 Нефункциональные требования**

**Требования к применению:**

Интерфейс должен быть легким, понятным, функциональным и простым в использовании. Он не должен перегружать пользователя ненужной информацией и напрягать глаза яркой цветовой гаммой.

**Требования к производительности:**

Требования к производительности наименьшие. Для повышения производительности игрового приложения и во избежание ошибок должны использоваться не нагружаемые компоненты и ресурсы.

**Требования к реализации:**

Для реализации данного программного продукта может быть использован язык C#, с игровым движком Unity.

**Требования к надежности:**

Для данного продукта решено провести аlfa- и beta-тестирования перед выпуском игрового приложения.

**Требования к интерфейсу:**

При запуске будет проигрываться анимация пробегающего персонажа. В главном меню будет находиться несколько кнопок: «Начать игру» (позволит выбрать режим), «настройки» ( управление настройками звука и изображения), «информация» (информация о игре, механиках, авторах и подобной информации), «выход»(закрывает игровое приложение), «Значок кубка»(отображает текущий рекорд, набранный игроком). Цвета интерфейса будут представлять собой ярко выраженную палитру. Весь интерфейс будет исполнен с использованием атрибутики 80-90-х годов, визуальных ретро-решений.

* 1. **Диаграмма вариантов использования**

Проектирование – один из важных шагов при разработке программы, который очень часто игнорируется начинающими разработчиками. Как результат, у них нет чёткого плана дальнейших действий, и проект может быть отложен.

Обычно при проектировании разработчики изображают систему графически, поскольку человеку легко разобраться в таком представлении. Именно поэтому вместо написания громоздких текстов про каждую возможность будущей программы разработчики строят различные диаграммы для описания своих систем. Это помогает им не забывать, что нужно реализовать в программе, и быстро вводить в курс дела своих коллег.

На диаграмме вариантов использования изображаются:

- акторы — группы лиц или систем, взаимодействующих с системой;

- варианты использования — сервисы, которые наша система предоставляет акторам;

- комментарии;

- отношения между элементами диаграммы.

С диаграммой вариантов использования для игрового приложения «Пустынные бега» можно ознакомиться в Приложении А.

* 1. **Модель данных**

Модель данных — это абстрактное, самодостаточное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь. Эти объекты позволяют моделировать структуру данных, а операторы — поведение данных.

Модели данных служат для проектирования структуры постоянных хранилищ данных, используемых системой. Профайл на языке UML для проектирования базы данных предоставляет разработчикам базы данных набор элементов моделирования, позволяющих разрабатывать подробный макет таблиц в базе данных и моделировать макет физической памяти базы данных. Профайл базы данных на языке UML также предоставляет конструкции для моделирования целостности по ссылкам (ограничений и триггеров), а также хранимых процедур, предназначенных для управления доступом к базе данных.

Модели данных могут создаваться на уровне предприятия, отдела или отдельного приложения. Модели данных на уровне предприятия или отдела могут использоваться для предоставления стандартных определений для ключевых бизнес-сущностей (таких как клиент и сотрудник), которые будут применяться всеми приложениями всего бизнес-процесса или его части. С помощью этих типов Моделей данных можно также определить, какая система предприятия будет "владельцем" данных для конкретной бизнес-сущности и какие другие системы будут пользователями данных.

С моделью данных созданную для игрового приложения «Пустынные бега» можно ознакомиться в Приложение Б.

**2 Проектирование**

**2.1 Выбор стратегии разработки и модели жизненного цикла**

Для разработки игрового приложения следует выбрать стратегию разработки и модель жизненного цикла. Осуществляем выбор посредством составления таблиц:

Таблица 2 – Выбор модели жизненного цикла на основе характеристик требований

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории требований | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Являются ли требования к проекту легко определимыми и реализуемыми? | Да | Да | Да | Нет | Нет | Нет |
| 2. | Могут ли требования быть сформулированы в начале ЖЦ? | Да | Да | Да | Да | Нет | Нет |
| 3. | Часто ли будут изменяться требования на протяжении ЖЦ? | Нет | Нет | Нет | Нет | Да | Да |
| 4. | Нужно ли демонстрировать требования с целью их определения? | Нет | Нет | Да | Нет | Да | Да |
| 5. | Требуется ли проверка концепции программного средства или системы? | Нет | Нет | Да | Нет | Да | Да |
| 6. | Будут ли требования изменяться или уточняться с ростом сложности системы (программного средства) в ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| 7. | Нужно ли реализовать основные требования на ранних этапах разработки? | Нет | Нет | Да | Да | Да | Да |

Вычисления: 2 за каскадную, за 2 V- образную, 5 за RAD, 3 за инкрементную, 5 за быстрого прототипирования и 5 за эволюционную.

Итог: На основе результатов заполнения табл. 3 подходящей является модель быстрого прототипирования и эволюционная.

Таблица 3 – Выбор модели жизненного цикла на основе характеристик команды разработчиков

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории команды разработчиков  проекта | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1. | Являются ли проблемы предметной области проекта новыми для большинства разработчиков? | Нет | Нет | Нет | Нет | Да | Да |
| 2. | Являются ли инструментальные средства, используемые в проекте, новыми для большинства разработчиков? | Да | Да | Нет | Нет | Нет | Да |
| 3. | Изменяются ли роли участников проекта на протяжении ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| 4. | Является ли структура процесса разработки более значимой для разработчиков, чем гибкость? | Да | Да | Нет | Да | Нет | Нет |
| 5. | Важна ли легкость распределения человеческих ресурсов проекта? | Да | Да | Да | Да | Нет | Нет |
| 6. | Приемлет ли команда разработчиков оценки, проверки, стадии разработки? | Да | Да | Нет | Да | Да | Да |

Вычисления: 4 за каскадную, 4 за V-образную, 1 за RAD, 4 за инкрементную, 3 за быстрого прототипирования и 4 за эволюционную.

Итог: На основе результатов заполнения табл. 4 подходящими являются каскадная, V-образная, инкрементная и эволюционная модели.

Таблица 4 – Выбор модели жизненного цикла на основе характеристик коллектива пользователей

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории коллектива пользователей | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 1. | Будет ли присутствие пользователей ограничено в ЖЦ разработки? | Да | Да | Нет | Да | Нет | Да |
| 2. | Будут ли пользователи оценивать текущее состояние программного продукта (системы) в процессе разработки? | Нет | Нет | Нет | Да | Да | Да |
| 3. | Будут ли пользователи вовлечены во все фазы ЖЦ разработки? | Нет | Нет | Да | Нет | Да | Нет |
| 4. | Будет ли заказчик отслеживать ход выполнения проекта? | Нет | Нет | Нет | Нет | Да | Да |

Вычисления: 1 за каскадную, 1 за V-образную, 1 за RAD, 2 за инкрементную, 3 за быстрого прототипирования и 3 за эволюционную.

Итог: На основе результатов заполнения табл. 5 подходящей является модель быстрого прототипирования и эволюционная.

Таблица 5 – Выбор модели жизненного цикла на основе характеристик типа проектов и рисков

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № критерия | Критерии категории типов проекта и рисков | Каскадная | V-образная | RAD | Инкрементная | Быстрого прототипирования | Эволюционная |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 1. | Разрабатывается ли в проекте продукт нового для организации направления? | Нет | Нет | Нет | Да | Да | Да |
| 2. | Будет ли проект являться расширением существующей системы? | Да | Да | Да | Да | Нет | Нет |
| 3. | Будет ли проект крупно- или среднемасштабным? | Нет | Нет | Нет | Да | Да | Да |
| 4. | Ожидается ли длительная эксплуатация продукта? | Да | Да | Нет | Да | Нет | Да |
| 5. | Необходим ли высокий уровень надежности продукта проекта? | Нет | Да | Нет | Да | Нет | Да |
| 6. | Предполагается ли эволюция продукта проекта в течение ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
|  |  |  |  |  |  |  |  |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 7. | Велика ли вероятность изменения системы (продукта) на этапе сопровождения? | Нет | Нет | Нет | Да | Да | Да |
| 8. | Является ли график сжатым? | Нет | Нет | Да | Да | Да | Да |
| 9. | Предполагается ли повторное использование компонентов? | Нет | Нет | Да | Да | Да | Да |
| 10. | Являются ли достаточными ресурсы (время, деньги, инструменты, персонал)? | Нет | Нет | Нет | Нет | Да | Да |

Вычисления: 3 за каскадную, 2 за V-образную, 6 за RAD, 6 за инкрементную, 7 за быстрого прототипирования и 6 за эволюционную.

Итог: На основе результатов заполнения табл. 6 подходящей является модель быстрого прототипирования.

Общий итог: в итоге заполнения табл. 3 – 6 наиболее подходящей является модель быстрого прототипирования и эволюционная.

**2.2 Инструменты разработки**

Для разработки данного проекта будет выбрана среда Unity, так как это самое удобная и доступная среда разработки на данный момент. Unity – больше, чем движок, это среда для разработки компьютерных игр, в которой объединены различные программные средства, используемые при создании ПО – текстовый редактор, компилятор, отладчик и так далее. При этом, благодаря удобству использования, Unity делает создание игр максимально простым и комфортным, а мультиплатформенность движка позволяет игроделам охватить как можно большее количество игровых платформ и операционных систем. В первую очередь, как мы уже упоминали, движок Unity дает возможность разрабатывать игры, не требуя для этого каких-то особых знаний. Здесь используется компонентно-ориентированный подход, в рамках которого разработчик создает объекты (например, главного героя) и к ним добавляет различные компоненты (например, визуальное отображение персонажа и способы управления им). Благодаря удобному Drag & Drop интерфейсу и функциональному графическому редактору движок позволяет рисовать карты и расставлять объекты в реальном времени и сразу же тестировать получившийся результат.

Второе преимущество движка – наличие огромной библиотеки ассетов и плагинов, с помощью которых можно значительно ускорить процесс разработки игры. Их можно импортировать и экспортировать, добавлять в игру целые заготовки – уровни, врагов, паттерны поведения ИИ и так далее. Никакой возни с программированием.

Третья сильная сторона Unity – поддержка огромного количества платформ, технологий, API. Созданные на движке игры можно легко портировать между ОС Windows, Linux, OS X, Android, iOS, на консоли семейств PlayStation, Xbox, Nintendo, на VR- и AR-устройства.

Инструменты, используемые при разработке и написании сопутствующей документации:

Excel – для построения диаграммы Ганта

WEB-ресурс DRAW.IO – будет использоваться для создания графической части и разработки UML-диаграмм.

Figma – будет использована для проектирования интерфейса игрового приложения.

Microsoft Office Word – для написания документации к программному продукту.

Microsoft Office PowerPoint – для создания презентации, которая будет использована на защите проекта.

DrExplain – легкий в использовании и функциональный инструмент, упрощающий создание справочных файлов Windows, печать справочных руководств и документации в целом. Программа имеет интуитивно понятный интерфейс. Все созданные проекты можно сохранить в различных форматах: HTML Help, Winhelp и MS Help 2.0 / Visual Studio Help, Browser-based Help, PDF и Word RTF, а также печатной документации при использовании одного и того же проекта. В основном окне программы содержатся оглавление (в виде древовидного списка) и текстовый редактор. Это дает возможность легко ориентироваться в оглавлении, редактировать или перемещать разделы справки без каких-либо проблем. Также утилита позволяет конвертировать help-файлы из одного формата в другой. Кроме приложений для работы с текстом в данном продукте содержатся утилиты для создания скриншотов и редактирования графических файлов.

Для создания инсталлятора будет использоваться мощное и удобное средство - Smart Install Maker.

Программа обладает удобным и интуитивно понятным интерфейсом, а также полным набором необходимых функций для создания профессиональных инсталляторов с минимальным размером, высокой степенью сжатия файлов и приятным интерфейсом.

Помимо стандартного минимума, Smart Install Maker позволяет редактировать системный реестр и INI-файлы, создавать программные ярлыки, запускать ассоциируемые и исполняемые файлы, регистрировать новые шрифты и ActiveX компоненты, отображать тексты информации и лицензионного соглашения. Также, с помощью этой утилиты, можно создать мульти языковые инсталляторы с поддержкой более 20-ти популярных языков мира.

Adobe IIIustrator - это популярная среди художников программа для рисования. Удобна своей простотой и в то же время большим функционалом. Программа очень сильно распространена в использовании во всем мире и предпочитается рисующими людьми как один из самых удобных инструментов для создания рисунков. В нем можно сохранить файлы картинок во множестве вариантов (.png, .jpg, .bmp и так далее).

Также есть возможность сохранять изображение как файл фотошопа, заносить все в последнюю указанную программу и работать со слоями проекта, ничего не потеряв.

Adobe Photoshop CS6 x64 – редактор для редактирования изображений. Необходим для оформления некоторых элементов интерфейса. Также использовался и для создания элементов программного продукта «Загадки Эйнштейна». Сложен, но удобен своей многофункциольнальностью.

Aseprite – это редактор изображений [с доступным исходным кодом](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BA%D1%80%D1%8B%D1%82%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), предназначенный в первую очередь для рисования и анимации в стиле [пиксель-арт](https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D0%BA%D1%81%D0%B5%D0%BB%D1%8C-%D0%B0%D1%80%D1%82). Он работает на [Windows](https://ru.wikipedia.org/wiki/Windows), [macOS](https://ru.wikipedia.org/wiki/MacOS" \o "MacOS) и [Linux](https://ru.wikipedia.org/wiki/Linux) и содержит различные инструменты для редактирования изображений и анимации, такие как слои, кадры, поддержка тайловых карт, интерфейс командной строки и другие.

Разработка проекта ведется на ноутбуке Honor. У данного ноутбука следующие параметры:

- процессор AMD Ryzen 5 5600H with Radeon Graphics 3.30 GHz;

- объем ОЗУ 16 гб;

- объем места на SDD – 500 гб ;

- ОС – Windows 11.

Как видно разрабатываемое приложение не очень требовательно к аппаратным ресурсам, что, является большим плюсом.

**2.3 Разработка UML-диаграмм**

**2.3.1 Диаграмма вариантов использования**

Проектирование – один из важных шагов при разработке программы, который очень часто игнорируется начинающими разработчиками. Как результат, у них нет чёткого плана дальнейших действий, и проект может быть отложен.

Обычно при проектировании разработчики изображают систему графически, поскольку человеку легко разобраться в таком представлении. Именно поэтому вместо написания громоздких текстов про каждую возможность будущей программы разработчики строят различные диаграммы для описания своих систем. Это помогает им не забывать, что нужно реализовать в программе, и быстро вводить в курс дела своих коллег.

На диаграмме вариантов использования изображаются:

- акторы — группы лиц или систем, взаимодействующих с системой;

- варианты использования — сервисы, которые наша система предоставляет акторам;

- комментарии;

- отношения между элементами диаграммы.

С диаграммой вариантов использования для игрового приложения «Пустынные бега» можно ознакомиться в Приложении А.

**2.3.2 Диаграмма классов**

Диаграмма классов — структурная [диаграмма](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0_(UML)) языка моделирования [UML](https://ru.wikipedia.org/wiki/UML), демонстрирующая общую структуру иерархии [классов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) системы, их коопераций, [атрибутов](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%B5_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B0) (полей), [методов](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_(%D1%8F%D0%B7%D1%8B%D0%BA%D0%B8_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)), интерфейсов и взаимосвязей (отношений) между ними. Широко применяется не только для документирования и визуализации, но также для конструирования посредством прямого или обратного проектирования.

При разработке программного продукта была поставлена задача разработать диаграмму классов, которая показывает общую структуру иерархии [классов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) системы. С данной диаграммой можно ознакомиться в Приложении В.

**2.3.3 Диаграмма объектов**

Диаграмма объектов в языке моделирования [UML](https://ru.wikipedia.org/wiki/UML) предназначена для демонстрации совокупности моделируемых объектов и связей между ними в фиксированный момент времени.

Диаграмма объектов описывает конкретные экземпляры объектов и напрямую соотносится с [диаграммой классов](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%BE%D0%B2), которая даёт общее представление о конфигурации системы. Она используется для документирования структур данных и создания статических снимков состояний объектов принимая во внимание реальные экземпляры или прототипы. Динамику поведения объектов обычно изображают в виде последовательности таких диаграмм.

При разработке программного продукта была поставлена задача разработать диаграмму объектов, которая описывает процесс связей между ними в фиксированный момент времени. С данной диаграммой можно ознакомиться в Приложении Г.

**2.4 Разработка пользовательского интерфейса**

Важным элементом проектирования данного программного продукта является описание внешнего интерфейса разрабатываемого игрового приложения.

Для разработки визуального дизайн использовались сдержанные, мягкие цвета для удобства использования программного продукта.

В ходе разработки был спроектирован дизайн главной страницы игрового приложения «Пустынные бега». Ранее разработанная структура игрового приложения расположена в Приложении А.

Для организации эффективной работы пользователя нужно создать целостный программный продукт данной предметной области, в котором все компоненты будут сгруппированы по функциональному назначению. При этом необходимо обеспечить удобный графический интерфейс пользователя. Игровое приложение должно позволить пользователю решать задачи, затрачивая значительно меньше усилий, чем при работе с разрозненными объектами. Все исходные данные будут разделены на несколько групп.

Прототип – это наглядная модель пользовательского интерфейса. В сущности, это «черновик» созданный на основе представления разработчика о потребностях пользователя. Итоговое отображение программы может отличаться от прототипа. С прототипом главного меню вы можете ознакомиться в Приложении Д.

Так же поставленной задачей на практику была реализация UX/UI интерфейсов. При создании UX/UI интерфейсов были использованы модульные сетки с целью создания пропорционального, понятного интерфейса. Использовались преимущественно оттенки красок 90х годов. Основные разделы доступны с первой страницы. Таким образом был реализован понятный пользовательский интерфейс, созданы макеты под такие устройства как: компьютер. Целью проекта была реализация интерфейса для фронтенд части.

https://www.figma.com/file/2xkRHK18JSwNghR55zEFjY/Untitled?type=design&node-id=0%3A1&mode=design&t=sCpjuUOPhYVGMfo9-1

**2.5 Разработка плана работы над проектом**

Для разработки плана над проектом был использован web-ресурс Wrike. Данное программное обеспечение позволяет отслеживать прогресс и анализировать объём работы. Показывает в какие строки может быть выполнена та или иная задача.

В данном случае Excel позволил быстро составить план разработки проекта, с возможностью отмечать степень его готовности. Таким образом это должно увеличить скорость разработки. План разработки составленный в Excel представлен в Приложении Е.

**3 Реализация**

**3.1 Руководство программиста**

Данный программный продукт был разработан с помощью движка Unity. Данный движок даёт неограниченные возможности в создании программных продуктов, имея обширную базу настроек, библиотек, физики, спрайтов, что позволяет создавать достаточно качественные проекты.

**3.1.1 Создание приложения**

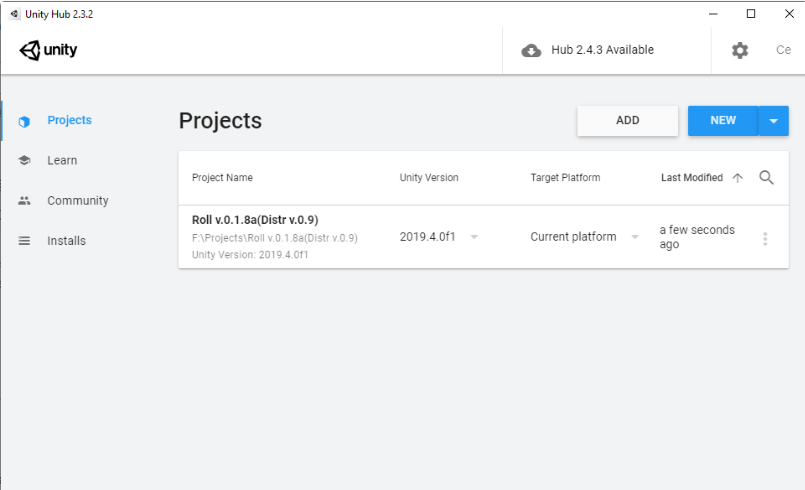
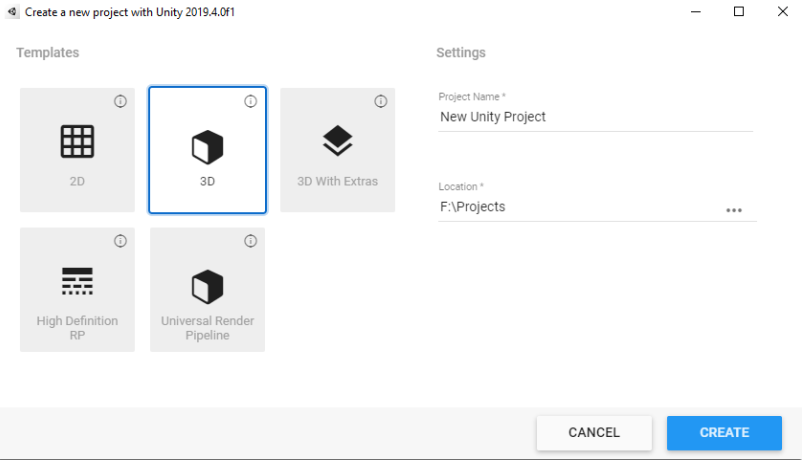
Для того, чтобы создать своей проект в Unity, необходимо сначала, открыть Unity Hub, предварительно его установив, и зарегистрироваться в нём (рисунок 1).

Рисунок 1 – Меню Unity Hub

В основной области находятся поле с проектами и еще правее кнопки «Add» и «New», с выпадающим меню.

Кнопка «Add» отвечает за добавление, уже имеющихся, проектов на вашем компьютере и при нажатии предложит выбрать папку с проектом. Нужная нам кнопка «New» отвечает за создание нового проекта, нажимаем на нее.

Открывается окно с предустановками проекта, здесь можно выбрать один из 5 подготовленных шаблонов для создания игры. Выбираете конечно тот, который подходит под вашу идею. Не забудьте указать имя проекта и папку, где будут храниться все его файлы (рисунок 2).

 Рисунок 2 – Создание проекта

Далее нажимаем кнопку «Create» и дожидаемся окончания процесса инициализации.

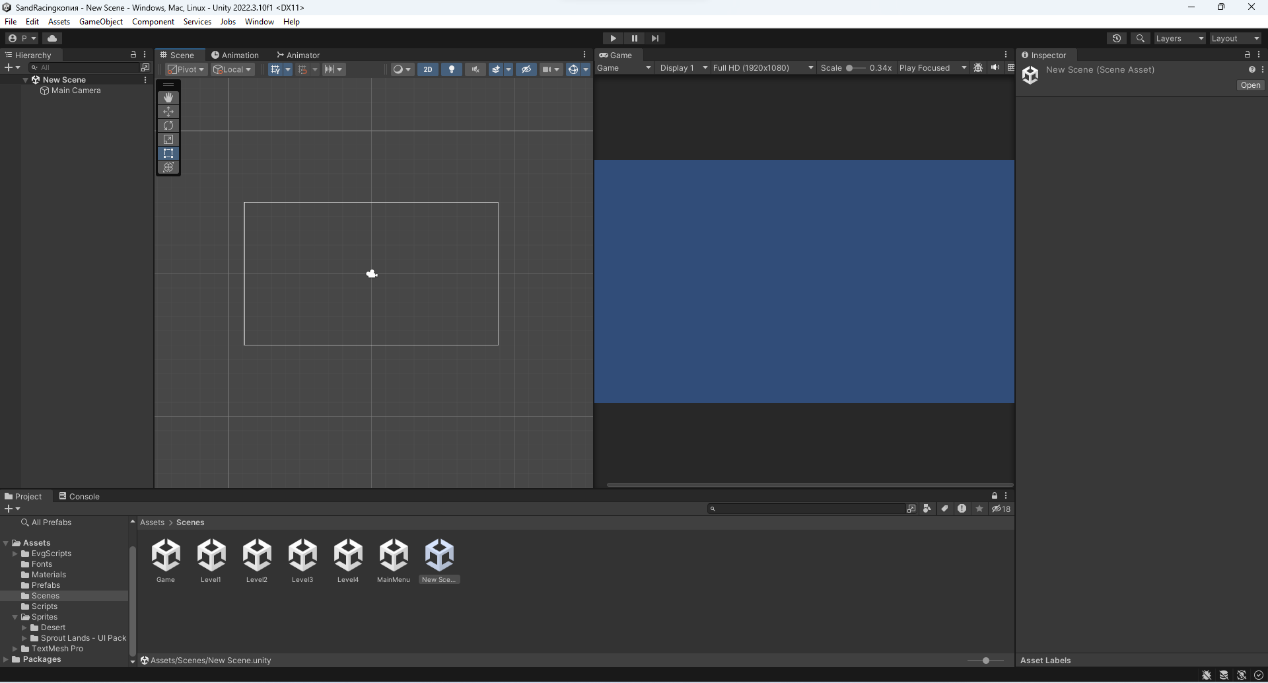
После окончания загрузки вы можете начинать создавать все внутриигровые объекты (рисунок 3).

Рисунок 3 – Созданный проект Unity

**3.1.2 Настройка функциональности и дизайна**

В Unity проекта находиться множество панелей отвечающие за ту или иную функцию. Опишем нужные функции и задачи для создания нашего проекта:

- Hierarchy – иерархия объектов – здесь можно добавлять, копировать, создавать объекты;

- Build Settings – настройки сцен, и установщик;

- Build and Run – создание сборки приложения и запуск его;

- Scene – панель, на которой располагаются все созданные объекты, которые можно редактировать;

- Game – симулятор панели, на которой отображаться расположенные объекты в виде готового проекта;

- Inspector – панель, в которой можно добавлять, изменять или удалять объекты и скрипты;

- Project – иерархия файлов;

- Console - панель отображения запуска и сбоев компонентов проекта;

- Scripts – папка в которой создаются C# скрипты для самого проекта.

**3.1.3 Реализация игрового приложения**

Первым шагом создаём главное меню.

При создании главном меню надо создать все необходимые кнопки, компоненты, задний фон, к ним же прикрепить все необходимые скрипты.

На главном меню расположены 4 кнопки: «Play», «Options», «Info», «Exit» (рисунок 4).



Рисунок 4 – Главное меню

Так же присутствуют 3 панели с выбором режима, настроек и рекорда (рисунок 5), (рисунок 6), (рисунок 7).

 Рисунок 5 – Выбор режима игры



Рисунок 6 – Меню настроек

Рисунок 7 – Панель рекорда

Вторым шагом создаём игровой режим забег с панелью при проигрыше (рисунок 8), (рисунок 9).

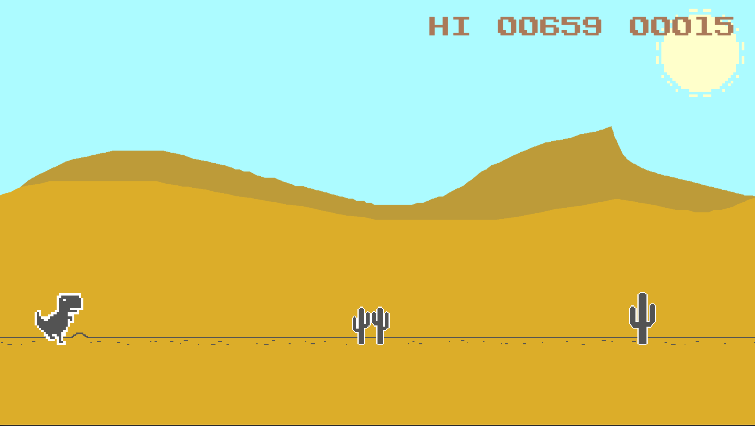


Рисунок 8 – Игровой режим забег

 Рисунок 9 – Игровой режим забег с панелью проигрыша

Третьим шагом создаём 4 уровня игры с панелями при проигрыше. Панели проигрыша не отличаются (рисунок 10-14).

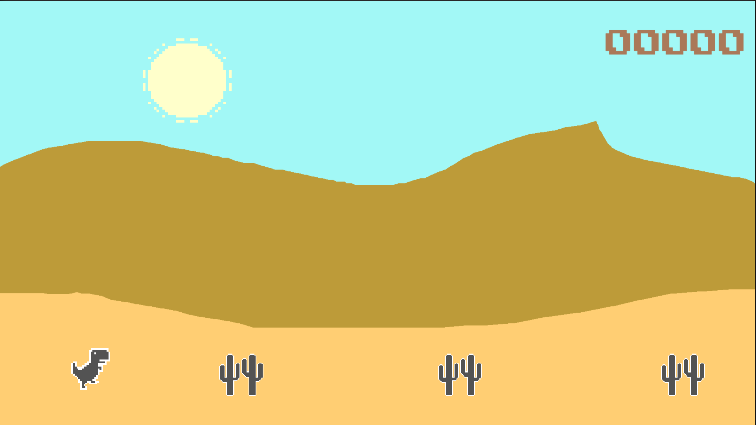
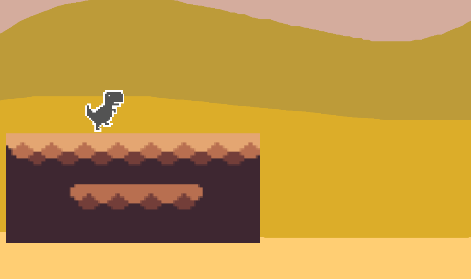


Рисунок 10 – Уровень 1



Рисунок 11 – Панелью проигрыша

Рисунок 12 – Уровень 2

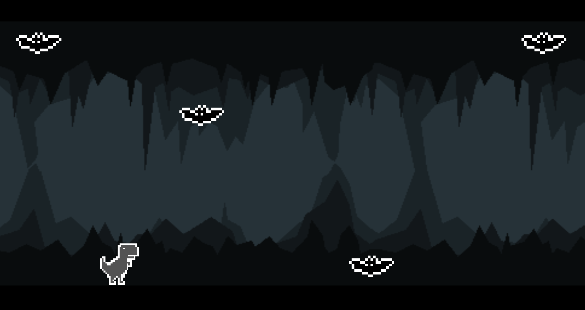


Рисунок 13 – Уровень 3



Рисунок 14 – Уровень 4

Четвёртым шагом создаём загрузочный экран и загрузочную концовку (рисунок 15), (рисунок 16).

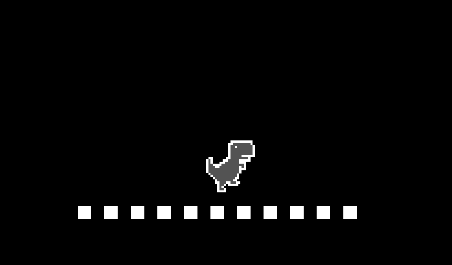


Рисунок 15 – Загрузочный экран

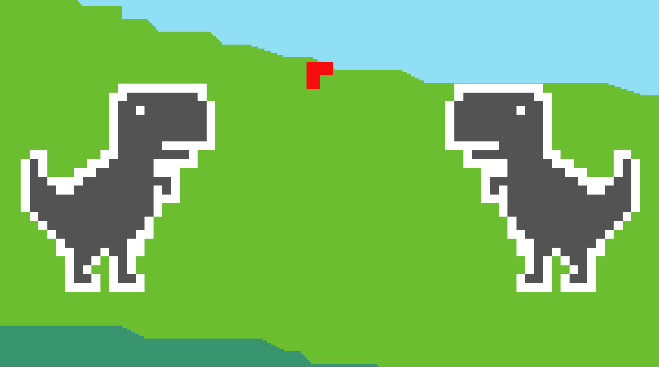


Рисунок 16 – Загрузочная концовка

**3.1.4 Создание билдера проекта**

Для того что бы создать билдер необходимо зайти: File > Build Settings… и выбрать платформу PC, потом добавить название приложения и его иконку. Далее нажимаем на кнопку Build и выбираем место куда будет создан ваш установщик.

**3.1.5 Создание и подключение справочной информации**

Для того, чтобы создать и подключить справочную информацию, необходимо:

1 Установить справочный редактор;

2 Создать файл в расширении .chm;

3 Подключить его с помощью кода в проект.

В рисунке 17 предоставлено изображение справочной системы.

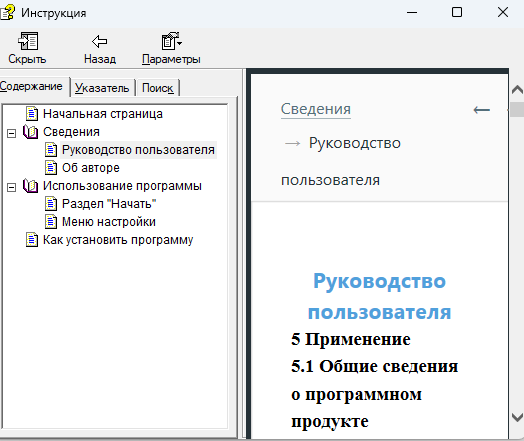


Рисунок 17 – Справочная информация

**4 Тестирование**

При разработке данной программы многие возникающие ошибки и недоработки были исправлены на этапе реализации проекта. После завершения этапа написания программы было проведено тщательно функциональное тестирование. Функциональное тестирование должно гарантировать работу всех элементов программы в автономном режиме. Отчёт о результатах тестирования представлен в таблице 6.

Таблица 6 – Отчёт о результатах тестирования функций для пользователя

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название теста | Действие | Ожидаемый результат | Фактический результат | Результат тестирования |
| 1 | 2 | 3 | 4 | 5 |
| Открытие панели выбор режима | Нажатие на кнопку «Play» | Откроется панель | Открылась панель | Выполнено |
| Открытие панели настройки | Нажатие на кнопку «Options» | Откроется панель | Открылась панель | Выполнено |
| Запуск справки | Нажатие на кнопку «Info» | Откроется справка | Открылась справка | Выполнено |
| Закрытие приложения | Нажатие на кнопку «Exit» | Закроется приложение | Закрылось приложение | Выполнено |
| Открытие панели рекорда | Нажать на кнопку рекорда | Откроется рекорд | Открылся рекорд | Выполнено |
| Вкл/выкл музыку | Смещение ползунка музыки | Вкл/выкл музыка | Вкл/выкл музыка | Выполнено |
| Регулировка громкости | Передвинуть ползунок | Изменится громкость | Изменилась громкость | Выполнено |
| Запуск режима забег | Нажатие на кнопку забег | Запуститься забег | Запустился забег | Выполнено |
| Запуск кнопки уровень 1 | Нажатие на кнопку уровня | Откроется уровень | Открылся уровень | Выполнено |
| Запуск кнопки уровень 2 | Нажатие на кнопку уровня | Откроется уровень | Открылся уровень | Выполнено |
| 1 | 2 | 3 | 4 | 5 |
| Запуск кнопки уровень 3 | Нажатие на кнопку уровня | Откроется уровень | Открылся уровень | Выполнено |
| Запуск кнопки уровень 4 | Нажатие на кнопку уровня | Откроется уровень | Открылся уровень | Выполнено |
| Выход в главное мнею | Нажатие на кнопку главного меню | Выйдет в главное меню | Вышло в главное меню | Выполнено |
| Повторный запуск игры | Нажатие на кнопку повторной игры | Запуститься игра | Запустилась игра | Выполнено |
| Возобновление игры | Нажатие на кнопку возобновления игры | Возобновится игра | Возобновилась игры | Выполнено |
| Открытие меню паузы | Нажатие на кнопку паузы | Остановится игра | Остановилась игра | Выполнено |
| Запуск полного экрана | Нажатие на кнопку полного экрана | Игра измениться на полный и в окне режим | Игра изменилась на полный и в окне режим | Выполнено |
| Смена настроек качества | Выбор качества из низких, средних, высоких настроек | Поменяются настройки качества | Поменялись настройки качества | Выполнено |
| Смена разрешения экрана | Выбор разрешения экрана | Поменяется разрешение экрана | Поменялось разрешение экрана | Выполнено |

В результате проведения тестирования выяснилось, что все ранее оговоренные функции и требования, были разработаны, а также протестированы. Тесты показали, что все функции работают правильно. Следовательно разработанное игровое приложение можно передать заказчику.

В ходе тестирования программного обеспечения продукта на разных устройствах не было выявлено каких-либо ошибок, так как адаптивность игрового приложения была проведена на всех стадиях разработки.

**5 Руководство пользователя**

**5.1 Общие сведения о программном продукте**

Разрабатываемое игровое приложение будет носить название «Пустынные бега».

Данный программный продукт может быть использован только на ПК.

Быстродействие любой программы во многом зависит от характеристик выбранного персонального компьютера: рабочей частоты процессора, объема оперативной памяти и т.д. Несмотря на все реализованные в ней задачи, она легко запускается и функционирует на любых машинах ПК.

Тестирование проводилось на разных классах ЭВМ и работать с данной программой было комфортно. Программа разработана на ПК со следующими характеристиками:

- процессор AMD Ryzen 5 5600H with Radeon Graphics 3.30 GHz;

- объем ОЗУ 16 гб;

- объем места на SDD – 500 гб ;

- ОС – Windows 11.

**5.2 Инсталляция**

Для того, чтобы установить программу необходимо запустить файл Setup.exe. Появится окно установки приложения «Пустынные бега». Затем достаточно следовать приведенной инструкции установки приложения.

**5.3 Выполнение программы**

**5.3.1 Запуск программы**

Данную программу можно запустить различными способами.

Первым из них является запуск с помощью ярлыка на рабочем столе. Необходимо дважды щелкнуть левой кнопкой мыши на ярлыке с названием «Пустынные бега».

Вторым способом является запуск из каталога, в который устанавливалось приложение (по умолчанию C:\Program Files (x86)\Творовский Влад\Пустынные бега).

**5.3.2 Инструкции по работе с программой**

После запуска приложения на экране нас встречает загрузочный экран представленный на рисунке 18.

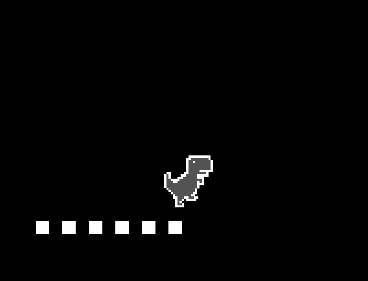


Рисунок 18 – Загрузочный экран

По окончании загрузочного экрана, нас встречает главное меню представленное на рисунке 19.



Рисунок 19 – Главное меню

При нажатии на кнопку «Play» откроется панель с режимами, где можно выбрать режим игры рисунок 20.



Рисунок 20 – Панель с режимами

Среди которых доступно режим забег и 1 левел, чтобы разблокировать левела 2, 3, 4, нужно их пройти.

При нажатии на кнопку забега, мы попадаем в саму бесконечную игру, при смерти которой можно запустить игру заново или вернуться в главное меню (рисунок 21), (рисунок 22).

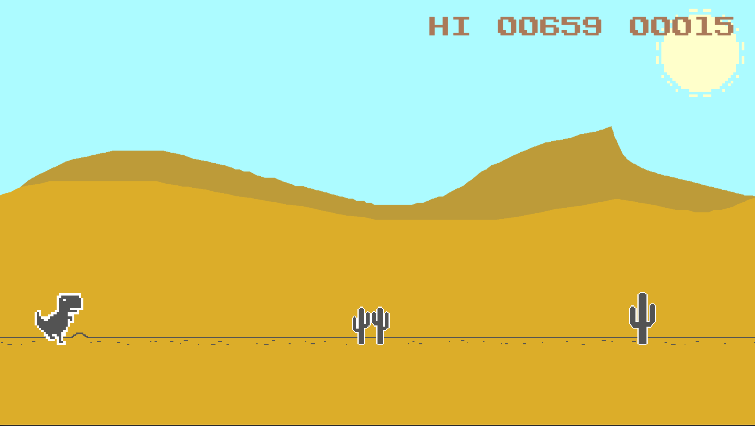
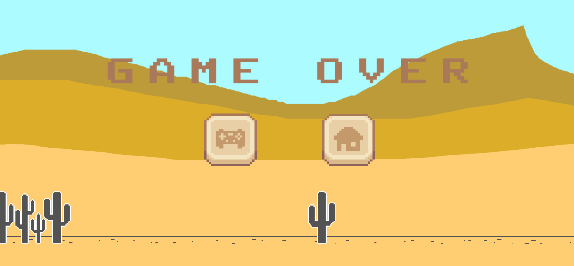


Рисунок 21 – Забег

Рисунок 22 – Меню смерти

Доступно всего 4 уровня, при смерти в которых можно начать заново или вернуться в главное меню. (рисунок 23-27).

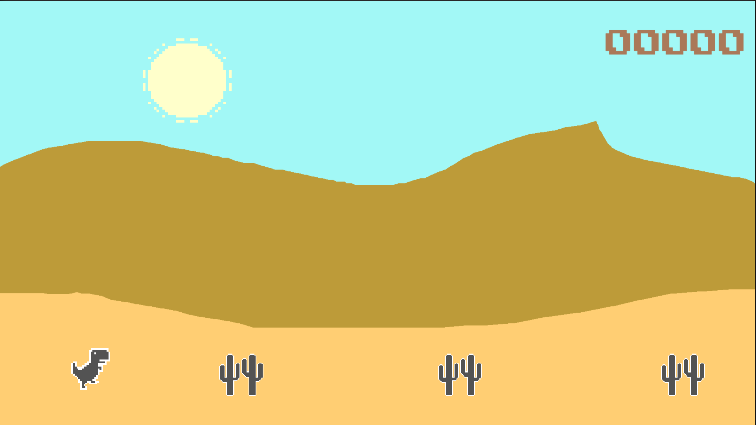
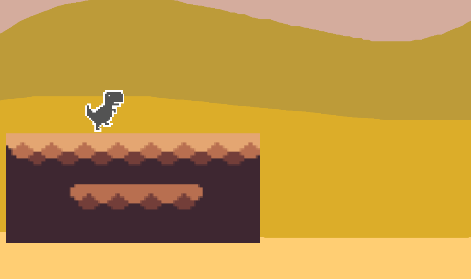


Рисунок 23 – Уровень 1



Рисунок 24 – Панелью проигрыша

Рисунок 25 – Уровень 2

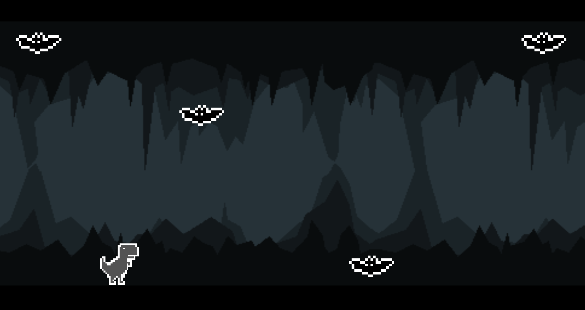


Рисунок 26 – Уровень 3



Рисунок 27 – Уровень 4

При прохождении уровня, отрывается следующий уровень. В конце 4 уровня загружается кат-сцена, после которой переходит на главное меню (рисунок 28).

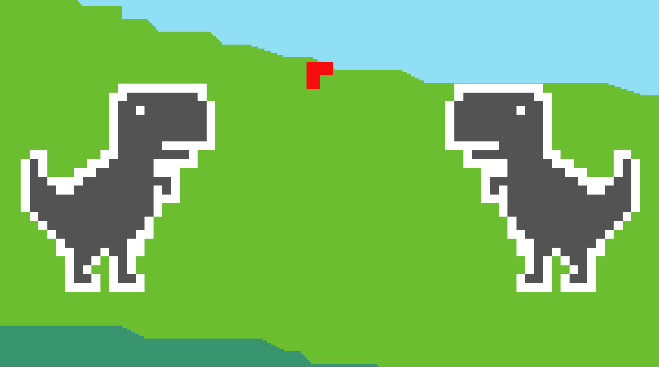


Рисунок 28 – Загрузочная концовка

На главном меню при нажатии на кнопку рекорда, откроется панель с рекордом (рисунок 29).

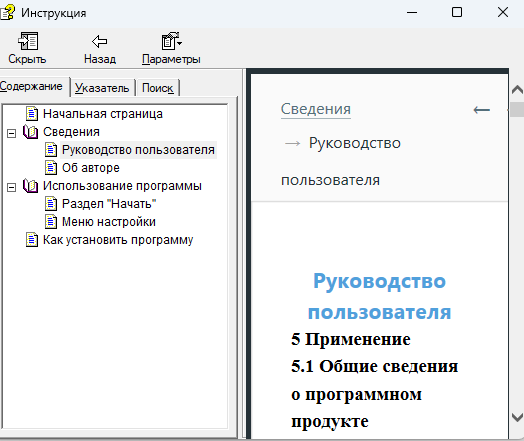


Рисунок 29 – Рекорд

Так же при нажатии кнопки настроек, откроется панель с настройками, где можно редактировать графику и звук (рисунок 30).

Рисунок 30 – Меню настроек

Если нажать на кнопку «Info» откроется справочная информация (рисунок 31).

Рисунок 31 – Справочная информация

Завершить работу с приложением можно несколькими способами:

1 Нужно нажать на крестик в углу игры, если она в оконном режиме.

2 Также можно закрыть приложение нажав на кнопку «Exit».

**Заключение**

Целью данного проекта на практику являлась разработка игрового приложения в стиле 2D-Runner «Пустынные бега».

Можно заметить, что в игровом приложении был реализован простой и удобный интерфейс, который позволяет использовать игровое приложение тому пользователю, который не обладает отличительными навыками работы с программным продуктом данного типа. При разработке были выполнены требования, поставленные в 1 и 2 разделе документации.

Данное игровое приложение предназначено для проверки своих навыков на внимательность и реакцию. Так же оно будет по душе людям всех возрастов.

После долгого и тщательного тестирования игрового приложения были выявлены небольшие ошибки и недоработки, которые в последствии были исправлены на стадии тестирования.

В целом при реализации программного продукта, были выполнены все условия, поставленные на начальном этапе разработки. Из чего мы можем сделать вывод, что игровое приложение можно передавать заказчику и вводить в эксплуатацию.

Так же в процессе создания программного продукта была подготовлена программная документация.

**Список использованных источников**

1. Современный учебник Unity [Электронный ресурс] – Режим доступа: https://unity.com/ru – Дата доступа: 03.11.2023
2. Создание справок [Электронный ресурс]/ Русско-язычное сообщество htmleditors. – Режим доступа: http://htmleditors.ru/Rasnoe/help/vvedenie.html. – Дата доступа: 21.11.2023
3. Создание инсталлятора [Электронный ресурс]/ InstallBuilders Company. – Режим доступа: http://ru.sminstall.com/. – Дата доступа: 21.1.2023
4. Правила оформления списка использованных источников [Электронный ресурс]. – Режим доступа: https://etalonline.by/document/?regnum=hk9900296. – Дата доступа: 21.11.2023