|  |  |
| --- | --- |
| Министерство образования и науки Российской Федерации  Федеральное государственное автономное образовательное учреждение высшего профессионального образования «Российский государственный профессионально-педагогический университет» | |
|  |  |
| Институт инженерно-педагогического образования  Кафедра информационных систем и технологий | |
|  | |
| КУРСОВАЯ РАБОТА  по дисциплине  **«Технологии программирования»** | |
| на тему  **«Разработка компьютерной игры**  **Grumpy Three Friends»** | |
| **Направление подготовки** 09.03.02 Информационные системы и технологии  **Профиль** Информационные технологии в медиаиндустрии | |
|  |  |
| Выполнили: | Студенты группы ИТм-403  Коньшин А.В.  Маракулина У.Е.  Шульгин П.В.  Юдин Д.Е. |
|  |  |
| Проверил: | Старший преподаватель Черноскутов М.Ю. |
| Екатеринбург 2018 | |

**СОДЕРЖАНИЕ**

[Введение 3](#_Toc533244683)

[1 Предпроектное обследование 4](#_Toc533244684)

[1.1 Выбор вида компьютерной игры 4](#_Toc533244685)

[1.2 Выбор аппаратно-программной платформы и инструментальных средств разработки компьютерной игры 6](#_Toc533244686)

[2. Техническое задание 9](#_Toc533244687)

[Введение 9](#_Toc533244688)

[Основания для разработки 9](#_Toc533244689)

[Назначения разработки 10](#_Toc533244690)

[Требования к программе или программному изделию 10](#_Toc533244691)

[Требования к функциональным характеристикам 10](#_Toc533244692)

[Требования к надежности 11](#_Toc533244693)

[Условия эксплуатации 11](#_Toc533244694)

[Требования к составу и параметрам технических средств 11](#_Toc533244695)

[Требования к информационной и программной совместимости 12](#_Toc533244696)

[Требования к маркировке и упаковке 12](#_Toc533244697)

[Требования к маркировке и упаковке не регламентируются ввиду планируемого распространения программы без физического носителя. 12](#_Toc533244698)

[Требования к транспортированию и хранению 12](#_Toc533244699)

[Требования к программной документации 12](#_Toc533244700)

[Технико-экономические показатели 12](#_Toc533244701)

[Стадии и этапы разработки 13](#_Toc533244702)

[Порядок контроля и приемки 13](#_Toc533244703)

[Приложение 13](#_Toc533244704)

[3. Руководство программиста 15](#_Toc533244705)

[3.1 Назначение и условия применения программы 15](#_Toc533244706)

[3.2 Характеристика программы 15](#_Toc533244707)

[3.3 Обращение к программе 15](#_Toc533244708)

[4. Руководство пользователя 17](#_Toc533244709)

[4.1 Общие сведения о пользовательском интерфейсе программы 17](#_Toc533244710)

[4.2 Запуск компьютерной игры «Grumpy Three Friends» 17](#_Toc533244711)

[4.3 Работа с главным меню игры 17](#_Toc533244712)

[4.4 Начать игру 18](#_Toc533244713)

[4.5 Настройки 19](#_Toc533244714)

[4.6 Достижения 20](#_Toc533244715)

[4.7 Разработчики 21](#_Toc533244716)

[4.8 Выход 22](#_Toc533244717)

[4.9 Решение технических проблем 22](#_Toc533244718)

[5. Отчеты участников команды 23](#_Toc533244719)

[5.1 Отчет программиста 23](#_Toc533244720)

[5.2 Отчет художника 25](#_Toc533244721)

[5.3 Отчет тестировщика 25](#_Toc533244722)

[5.4 Отчет тимлида 26](#_Toc533244723)

[Заключение 28](#_Toc533244724)

[Список литературы 29](#_Toc533244725)

# Введение

В настоящее время компьютерные игры являются одним из наиболее востребованных видов мультимедийных приложений. Это объясняется их положительным влиянием на развитие человека, предоставлением качественного развлекательного контента, а также современным переходом в глобальное киберспортивное направление.

Сейчас в мире существует множество производителей компьютерных игр. При этом в данной индустрии работают и большие компании, и индивидуальные разработчики. Но несмотря на огромное разнообразие игрового контента, сохраняет свою актуальность проблема востребованности в новых продуктах, что связано с наличием у каждого человека своих интересов и предпочтений в выборе игры, на которую он готов потратить свои деньги и время с учетом того, что рано или поздно этот продукт перестанет быть для него интересным, после чего начнется поиск уже новой игры.

Объектом исследования в курсовой работе является компьютерная игра «Grumpy Three Friends».

Предмет исследования составляют методы разработки компьютерных игр.

Цель курсовой работы – разработать компьютерную игру, работая в команде, деятельность которой будет организована через систему Git.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Определить идею для новой игры.
2. Выбрать метод разработки.
3. Разработать продукт.
4. Провести тестирование и отладку.

# Предпроектное исследование

## 1.1 Выбор вида компьютерной игры

Проект было решено создавать в виде компьютерной игры как наиболее популярное и востребованное направление в разработках интерактивных мультимедийных приложений. Выбран 2D формат в связи с определенной концепцией компьютерной игры.

Рассмотрим некоторые особенности основных жанров для возможной реализации продукта без использования 3D технологий.

Аркады – один из самых старых жанров. Главный их признак – простейшее управление. Например, геймеру не нужно ничего знать о том, как водить машину в реальной жизни. Достаточно нажать на кнопку со стрелкой, чтобы повернуть.

Однако это совсем не значит, что в аркаде очень легко выиграть. Многие разработчики следуют золотому правилу: легко освоить, трудно победить. Аркады можно условно разделить на несколько поджанров:

1. Скроллер – игра с линейными уровнями, которые прокручиваются влево или вправо. Сюда можно отнести классический Golden Axe.
2. Комната – сначала необходимо выполнить какую-либо задачу в ограниченном пространстве, после чего открывается дверь, которая позволит перейти на следующий подобный уровень. Типичный представитель – Digger.
3. Тир – целью является попадание по мишеням (Duck Hunt, некоторые уровни «Контры»).

Компьютерные игры жанра action предполагают управление человеком. Главное отличие от аркад – сложность. Причем выражается она не в количестве усилий, затраченных для победы, а в проработке геймплея и окружения. Почти всегда девелопер старается сделать так, чтобы виртуальная реальность была как можно более реалистичной (невозможность забраться по отвесной стене или прыгнуть выше нескольких десятков сантиметров, вид от первого лица, ограничение на скорость передвижения и т. п.). Можно сказать, что родоначальники все еще оставались аркадами, но большая свобода сразу выделила их в отдельную категорию.

Тактические стратегии – игры из жанра планирования, главное их отличие в том, что управление осуществляется не отрядами и армиями, а всего лишь несколькими юнитами. К тому же каждый боец будет иметь индивидуальные характеристики, свое личное снаряжение и оружие. Система развития персонажей походит на ту, которую применяют в RPG.

Жанр сюжетных развлечений забирает в себя тех представителей виртуальных развлечений, которые ставят во главу угла не столько геймплей, сколько повествование, атмосферу, качественный сюжет. О них чаще всего говорят: «Это та игра, в которой можно жить». Нередко в них присутствуют черты и экшена, и стратегий, однако не для этого в первую очередь устанавливают сюжетные приключения. Именно такое положение вещей не позволяет причислить к подобным проектам Diablo и его клонов, как бы сильно этого ни хотели фанаты сего продукта.

Компьютерные игры в жанре квест – самые чистокровные представители сюжетных приключений. В них геймеру заранее отводится определенная роль, и с этой точки зрения рассказывается какая-либо интерактивная история. Квесты почти всегда линейны, пройти от начала до конца можно только по одному пути. Возможностей решения каждой задачи – минимум. Главные действия – общение с NPC, поиск предметов, их комбинирование.

В настоящее время существует еще множество других жанров и поджанров компьютерных игр, но для реализации двухмерной игры наиболее подходящие были рассмотрены.

Было решено реализовать продукт в жанре сюжетной аркады смешанного стиля. Тип игры – одиночная, на разовое последовательное прохождение сюжета. Отсутствуют любые варианты сохранения данных в следствие небольшой длительности игрового процесса. Такое представление выбрано в соответствии с разработанной ранее концепцией проекта.

## 1.2 Выбор аппаратно-программной платформы и инструментальных средств разработки компьютерной игры

Проект разрабатывается в среде Unity для использования на персональных компьютерах. Именно в выбранном продукте наиболее качественно раскрыты возможности создания игр-платформеров и определения физических свойств объектов, что позволяет использовать различные методологии создания игровых элементов и событий.

Также проведен анализ программы по отношению к другим наиболее популярным аналогичным средствам создания компьютерных игр с точки зрения набора характеристик, необходимых для разработки качественного продукта. Результаты исследования представлены в Таблица 1.

Таблица 1 – Сравнение средств разработки компьютерной игры

|  |  |  |  |
| --- | --- | --- | --- |
| **Характеристика** | **Adobe Flash** | **Unity** | **Unreal Engine** |
| Работа с векторными форматами. | Поддерживает средства работы с векторными форматами. | Отсутствуют средства работы с векторными форматами. | Отсутствуют средства работы с векторными форматами. |
| Работа с растровыми форматами. | Присутствует основной (базовый) функционал. | Присутствует основной (базовый) функционал. | Присутствует основной (базовый) функционал. |
| Работа с аудио форматами. | Поддерживает только .wav формат. | Поддерживает все популярные аудио форматы. | Поддерживает все популярные аудио форматы. |
| Выходные данные. | Обеспечивает небольшой размер, но требует наличие Flash Player. | Средний размер выходных. Создает полностью независимый .exe файл приложения. | Наиболее высокий объем выходных данных. Создает полностью независимый .exe файл приложения. |
| Поддерживаемые языки программирования. | Только собственный Action Script. Специализированный, адаптирован под данный программный продукт. | Поддержка C# и JavaScript. | Основной язык – С++. |
| Сложность для изучения и разработки. | Простой и удобный интерфейс и функционал. | Относительно простой для работы и изучения. | Сложная, большая система, требующая высокие аппаратные характеристики компьютера. |
| Защита доступа к коду. | Полностью защищенный исходный код в выходном файле, к которому невозможно получить доступ. | Требует ручного создания защиты. | Требует ручного создания защиты. |

В результате были выделены основные положительные и отрицательные моменты Unity.

Достоинства:

1. Unity обеспечивает относительно небольшой размер выходных данных, в частности созданного .exe билда.
2. Популярный язык программирования C#, который поддерживает события, условия, циклы, массивы, функции и наследуемые классы.
3. Безопасность. Если программа переведена в готовый .exe формат, то обычный пользователь не сможет увидеть ее исходный код.

Недостатки:

1. Сложность освоения.

Для данного проекта достоинства оказались намного весомее недостатков, поэтому и был выбран Unity.

Дополнительные графические материалы создавались и редактировались при помощи графических редакторов Adobe Photoshop CC и Adobe Illustrator CC, выбранных в связи с их популярностью и обширной функциональностью.

# Техническое задание

## Введение

* Наименование программы.

Компьютерная однопользовательская игра в жанре аркады “Grumpy Three Friends”.

* Краткая Характеристика области применения программы или программного изделия.

Данная программа предназначена для приятного времяпровождения пользователя в свободное время.

* Краткая характеристика объекта, в котором используют программу или программное изделие

Программа предназначена для эксплуатации на персональных компьютерах пользователей.

## Основания для разработки

* Документ/документы на основании которых ведется разработка

Основанием для разработки программы является задание на курсовой проект по предмету «Технологии программирования» кафедры "Информационные системы и технологии" Российского Государственного Профессионально-Педагогического Университета в положении № 0-9823П.

* Организация, утвердившая этот документ и дата его утверждения

Данный документ утвержден организацией ФГАОУ ВО «Российский государственный профессионально-педагогический университет» 24 сентября 2018 года.

* Наименование и условное обозначение темы разработки

Создание компьютерной 2D-игры (далее по тексту – Игра).

## Назначения разработки

Функциональное и эксплуатационное назначение программы или программного изделия

* Функциональное назначение: предоставление пользователю возможности проведения свободного времени в игровой форме.
* Эксплуатационное назначение: использование на подходящих под минимальные системные требования персональных компьютерах. Конечные потребители – обычные люди, не нуждающиеся в дополнительном инструктаже и обучении.

## Требования к программе или программному изделию

## Требования к функциональным характеристикам

Игра должна обеспечивать возможность выполнения следующих функций:

* Запуск Игры
* Прохождение до конца
* Главное меню должно содержать следующие кнопки:
  + Начало Игры
  + Настройки Игры
  + Список достижений
  + Разработчики Игры
  + Выход из Игры
* Изменение конфигурации Игры
  + Осуществляется на экране настроек, перейти в которые можно из главного меню.
  + Конфигурация сохраняется для последующих запусков.
* Управление персонажем.
  + Перемещение влево-вправо
  + Прыжок
* Воспроизведение звука.
  + Воспроизведение фоновой музыки.
* Возможность вернуться в главное меню.
  + Осуществляется через меню Паузы в игровом уровне.
* Пропуск диалогов.
  + Переход к следующему шагу диалога.
* Использование системы достижений.
  + Возможность просмотреть полученные и неполученные награды за прохождение сюжета.

Организация входных данных (методы ввода)

* Метод ввода – манипуляторы клавиатура и компьютерная мышь.

Организация выходных данных (методы вывода)

* Монитор, аудиосистема.

## Требования к надежности

Программа должна быть устойчива к ошибкам, то есть не допускать их возникновения.

## Условия эксплуатации

* Температура окружающего воздуха: от 5 до 28 градусов Цельсия.
* Относительная влажность: от 40 до 70%.

## Требования к составу и параметрам технических средств

* + - Минимальная конфигурация ПК:
      * частота процессора не менее 1 ГГц;
      * операционная система Windows 7 или выше;
      * 1024 Мбайт оперативной памяти;
      * манипулятор клавиатура и компьютерная мышь;
      * не менее 920 Мбайт свободного пространства на диске;
      * рабочее разрешение экрана, 1280x720 точек;
      * звуковая карта, колонки и/или наушники;
      * сетевая карта.

## Требования к информационной и программной совместимости

Требования к информационной совместимости: исходные коды программы должны быть реализованы на языке C#.

Требования программной совместимости: программа должна обеспечивать совместимость с ОС Windows.

## Требования к маркировке и упаковке

Требования к маркировке и упаковке не регламентируются ввиду планируемого распространения программы без физического носителя.

## Требования к транспортированию и хранению

* Ссылка в общем доступе на Игру в облачном хранилище.

## Требования к программной документации

* Руководство пользователя.

## Технико-экономические показатели

* Ориентировочная экономическая эффективность не рассчитывается в виду некоммерческого статуса программы;
* Предполагаемая годовая потребность: 150 запусков по 10-15 мин в среднем;
* Экономические преимущества разработки по сравнения с лучшими отечественными и зарубежным образцами не рассчитываются в виду некоммерческого статуса программы;

## Стадии и этапы разработки

* Стадии разработки, этап и содержание работ
  + Разработка технического задания;
  + Разработка Игры;
  + Тестирование Игры;
  + Доработка документации;
* Перечень программных документов
  + Пользовательское лицензионное соглашение.
  + Руководство пользователя.
  + Техническое задание.
* Сроки разработки
  + До 22 декабря 2018 года.
* Исполнителями являются студенты группы ИТм-403:
  + Коньшин Александр Владимирович – ведущий разработчик, разработчик анимации;
  + Маракулина Ульяна Евгеньевна – цифровой художник, дизайнер, арт-директор, музыкальное сопровождение;
  + Шульгин Петр Владимирович – тестировщик, концепт-художник, идейное сопровождение, разработчик;
  + Юдин Дмитрий Евгеньевич – сценарист, гейм-дизайнер, режиссер, программный директор, менеджер по работе с персоналом;

## Порядок контроля и приемки

Предоставить на проверку Игру и сопровождающую документацию проверяющему.

## Приложение

* Сюжет Игры:

Звери попали под пристальное внимание ученого Альфонса Франегнейта и угодили на лабораторный стол. Его целью было скрещивание овощей с животными, чтобы были вкуснее (генетически модифицированные животные с генами овощей и фруктов для создания нового вкусного питательного продукта). Но пошло все не по плану, так получилось свирепые, агрессивные, жестокие, кровожадные мутанты, которых он решил собрать в войско. Его научная кухня, позже превратившаяся в зловещую организацию, называется “Salvos”. Главные герои - трио животных, не связанных между собой, решающие разобраться в происходящей ситуации.

# Руководство программиста

## 3.1 Назначение и условия применения программы

Программа является игрой для ПК, средством развлечения и предназначена для позитивного времяпровождения. Суть игры заключается в том, что игрок должен управлять предоставляемым ему персонажем так, чтобы тот смог добраться до правого конца уровня и не задеть ловушки более двух раз.

Условия, необходимые для корректного запуска игры:

* + - * ОС Windows 7 или выше 64 бит;
      * .NET Framework 4.5.2.

Минимальные системные требования:

* + - * частота процессора не менее 1 ГГц;
      * операционная система Windows 7 или выше;
      * 1024 Мбайт оперативной памяти;
      * мышь, клавиатура;
      * 240 Мбайт свободного места на жестком диске;
      * разрешение экрана не менее 1280x720.

Рекомендуемые системные требования:

* + - * частота процессора не менее 1.7 ГГц;
      * операционная система Windows 7 или выше;
      * 3 Гб оперативной памяти;
      * мышь, клавиатура;
      * 380 Мбайт свободного места на жестком диске;
      * разрешение экрана не менее 1366x768.

Требования к персоналу (программисту):

Программист должен обладать практическими навыками работы с пользовательским интерфейсом среды разработки Unity, знаниями языка C# по отношению к используемым скриптам для создания игр, а также знать и уметь пользоваться системой контроля версий GitHub.

## 3.2 Характеристика программы

Средняя длительность прохождения: 24 минуты.

Идеальная длительность прохождения: 7 минут.

Программа не сохраняет данные – при перезапуске все данные (достижения, настройки) возвращаются в исходное состояние.

Быстродействие программы не зависит от качества ПК (при соответствии системным требованиям), то есть при повышении мощности компьютера быстродействие не увеличивается – игра стабильно работает на любых сборках системы.

Программа является автономной и способна работать без подключения к сети Интернет.

Все средства контроля правильности выполнения программы заложены в используемых скриптах, написанных на C#, за счет проверок условий выполнения того или иного метода.

Система предоставляет возможность ручного контроля процесса – управление персонажем по нажатию определенных клавиш. Во время работы игры происходит регистрация полученных достижения в ходе работы текущего запуска. Также позволяет изменять предложенные настройки игры, просматривать список достижений и разработчиков.

## 3.3 Обращение к программе

Запуск готовой игры происходит через файл в формате .exe, который находится в корневой папке игры Grumpy Tree Friends (Рисунок 1).

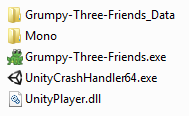


Рисунок 1 – Схема расположения файлов игры в проводнике

Весь исходный код программы расположен на удалённом репозитории GitHub по адресу: <https://github.com/SaPeUlDi/Grumpy-Three-Friends/>. Для внесения каких-либо изменений в программу, необходимо:

1. Войти в аккаунт GitHub (если его нет, необходимо пройти регистрацию и после войти).
2. Зайти на удалённый репозиторий проекта и создать fork. (как это сделать, можно узнать на этой странице: [https://git-scm.com/book/ru/v2/GitHub-Внесение-собственного-вклада-в-проекты](https://git-scm.com/book/ru/v2/GitHub-Внесение-собственного-вклада-в-проекты%20)).
3. Создать локальную копию репозитория.
4. Внести изменения.
5. Запушить на fork.
6. Создать запрос pull request'а в основной репозиторий ветик Unstable и ожидать одобрения запроса.
7. После того, как запрос будет одобрен, совершить pull request и merge в ветку Unstable.

Открыть сам проект в Unity можно, если зайти в программу Unity, нажать кнопку Open и выбрать корневую папку проекта. Структура папки разработки показана на Рисунке 2.

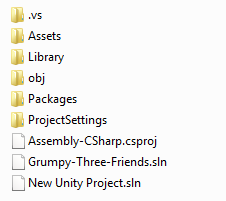


Рисунок 2 – Схема расположения файлов проекта в проводнике

При открытии проекта будет показан последний открытый разработчиком уровень при предыдущей работе с этим проектом в Unity.

## 3.4 Входные и выходные данные

Программа может принимать данные от пользователя только для настройки громкости звука и переключения оконного\полноэкранного режима работы. Остальные данные задаются фиксировано в проекте Unity.

Реализация выходной информации не предусмотрена.

## 3.5 Структура системы

Проект состоит из модулей-сцен, 6 из них объединены при помощи основного скрипта CharacterController.cs и могут независимо друг от друга исключаться из проекта для доработки безболезненно для всей системы при правильных изменениях названий сцен внутри скрипта в месте, указанном на Рисунке 3.



Рисунок 3 - Часть скрипта со сменой уровня

Остальные сцены имеют другую логику и содержат несколько скриптов:

* Unit – общий класс для всех юнитов в игре;
* Character : Unit – класс контроля персонажа. Содержит методы управления персонаже, отображение анимации и другое;
* Boss : Unit – класс, отвечающий за функционирование босса;
* Warm – класс, отвечающий за функционирование ловушек;
* CameraController – класс, отвечающий за контроль камеры;
* PauseS – реализация паузы;
* ButtonManager – менеджер кнопок панелей;
* LoadScene – класс, отвечающий за логику загрузки сцен;
* LivesBar – отображение жизней в интерфейсе игры;

Все сцены находятся в папке Assets/Scenes.

Скрипты расположены в Assets/Scripts.

Также в Assets представлены вспомогательные папки с используемыми медиаматериалами: Fonts шрифты, Images изображения и анимации, Music музыка и Shablon с шаблонами объектов.

## 3.6 Используемые функции

Некоторые функции требуют аргументов. Требования к аргументам описаны в таблице.

|  |  |  |
| --- | --- | --- |
| **Файл скрипта** | **Функция** | **Назначение** |
| CharacterController.cs  Основной скрипт работы всех уровней и персонажа. | Start() | Стандартная функция Unity. Выполняется 1 раз при первом запуске скрипта на сцене.  Задает основные объекты для других функций и некоторые первоначальные настройки, которые организованы сложнее обычной переменной или константы. |
| Update() | Стандартная функция Unity. Выполняется 1 раз на каждый кадр (фрейм) программы. Выполняет проверку всех необходимых функций как Flip(), Jump(), Trapped(), MoveRight() и т.д. |
| Flip() | Определяет направление движения персонажа для корректного отображения направления спрайта и анимации. |
| HitDelay() | Устанавливает задержку между соседними получениями урона от ловушки. |
| RemoveHitDelay() | Завершает задержку между соседними получениями урона от ловушки. |
| Trapped() | Проверка на столкновение с ловушкой. |
| Pause() | Приостановка игрового процесса и открытие игрового (на уровне) меню. |
| Jump() | Задает контроль прыжка – проверка на нахождение на земле и нажатие клавиши прыжка. |
| MoveRight() | Отвечает за перемещение персонажа вправо: до определенного значения расстояния от правого края окна перемещается сам персонаж, далее начинают перемещаться фон, ловушки и платформы, которые создают видимость движения персонажа.  Также реализует зацикливание фона и завершение уровня при касании финиша. |
| MoveLeft() | Отвечает за перемещение персонажа влево: до определенного значения расстояния от левого края окна перемещается сам персонаж, далее начинают перемещаться фон, ловушки и платформы, которые создают видимость движения персонажа. |
| GlobalNames.cs  Сохраняет данные о достижениях и громкости звуки на протяжении всех уровней. | Start() | Стандартная функция Unity. Выполняется 1 раз при первом запуске скрипта на сцене.  Запрещает уничтожение объекта музыки (с громкостью звука) при смене уровня. |
| Update() | Стандартная функция Unity. Выполняется 1 раз на каждый кадр (фрейм) программы.  При переходе в главное меню проверяет список полученных достижений и отмечает соответствующие в списке достижений в главном меню. |
| Mucis.cs  Задает громкость музыки и выбор песни для уровня. | Start() | Стандартная функция Unity. Выполняется 1 раз при первом запуске скрипта на сцене.  Задает громкость музыки, исходя из данных от GlobalNames.cs |
| MenuControls.cs  Отвечает за работу некоторых кнопок в главном меню. | PlayPressed() | Начинает новую игру при нажатии в главном меню кнопки «Новая игра». |
| ExitPressed() | Закрывает приложение. |

е

|  |  |  |
| --- | --- | --- |
| Unit  Содержит методы для дочерних классов | ReceiweDamage(Vector3 POZ) | Виртуальный метод, запускает метод Damage(). Требует вектор, содержащий позиции того объекта, который наносит урон. |
| isForce(Vector3 POZ) | Виртуальный метод(пустой) |
| Damage() | Метод уничтожающий объект. (переопределяемый) |
| Character  Управление персонажем | Start() | --//--//--//-- |
|  | Update() | --//--//--//-- |
|  | FixedUpdate() | Стандартная функция, вызывается с частотой фиксированных кадров (fixed framerate), если MonoBehaviour включен.  Используется FixedUpdate вместо Update когда необходимо работать с Rigidbody. |
|  | HitDelay() | Устанавливает задержку между соседними получениями урона от ловушки. |
|  | RemoveHitDelay() | Завершает задержку между соседними получениями урона от ловушки. |
|  | isForce(Vector3 poz) | Отталкивает персонажа в обратном направлении при касании ловушки/босса и получении урона. |
|  | ReceiweDamage(Vector3 poz) | Получение урона от ловушек/босса, отнимание жизней. |
|  | Dead() | Метод, отслеживающий смерть персонажа |
|  | Run() | Метод, отвечающий за движение персонажа по горизонтали. |
|  | Jump() | Реализация прыжка персонажа. |
|  | CheckGround() | Проверка – находится ли персонаж на платформе. |
| LoadScene  Отвечает за контроль загрузки уровней | LoadifDead() | При смерти персонажа, запускает уровень заново. |
|  | LoadifFinal() | Загружает следующий уровень, по завершению текущего. |
|  | Cat() | Загружает уровень босса с персонажем котом. |
|  | Dog() | Загружает уровень босса с персонажем псом. |
|  | Frog() | Загружает уровень босса с персонажем лягушонком. |
|  | LoadifMenu() | Загружает главное меню. |
| LivesBar  Отвечает за отображение жизней в интерфейсе игры. | Awaken() | Стандартный метод. Вызывается тогда, когда объект создаётся. |
|  | Start() | --//--//--//-- |
|  | Refresh() | Обновляет отображение жизней. |
| DieZone  Отвечает за смертельную зону, расположенную там, где нет платформы. Убивает персонажа. | OnTriggerEnter2D(Collider2D collider) | Одна из стандартных функций. Проверяет на соприкосновение коллайдеров объектов.  Проверяет на соприкосновение зоны с персонажем. |
| Warm  Отвечает за контроль ловушек | Awake() | --//--//--//--//--//-- |
|  | OnTriggerEnter2D(Collider2D collider) | Одна из стандартных функций. Проверяет на соприкосновение коллайдеров объектов.  Проверяет на соприкосновение ловушки с персонажем, если коснулся персонаж, нанести урон. |
| CameraController  Контроль камеры. | Awaken() | --//--//--//--//-- |
|  | Start() | --//--//--//--//-- |
|  | Update() | --//--//--//--//--  Проверяет позицию персонажа и перемещает камеру к этой позиции. |
| ButtonManager  Управляет кнопками различных панелей в игре. | PlayGame() | Продолжить игру(если стоит пауза) |
|  | MainMenu() | Загрузить главное меню во время паузы |
|  | OnTriggerEnter2D(Collider2D collider | --//--//--//--//--  Если касается персонаж финальной зоны, то активирует панель диалогов. |
| PauseS  Отвечает за паузу в игре | Update() | --//--//--//--//--  Отслеживает нажатие клавиши ESC, изменение переменной paused, активно ли DialPanel. В зависимости от ситуации, устанавливает паузу в игре или отключает её. |
|  | RevertPaused() | Изменяет значение переменной paused на противоположное. |
| Boss  Отвечает за контроль босса | Awake() | --//--//--//--  Запускает анимацию, получает необходимые компоненты. |
|  | Update () | --//--//--//--  Вызывает метод Move() |
|  | OnTriggerEnter2D(Collider2D collider | --//--//--//--  Проверяет: сверху или со стороны касается персонаж босса. Если сверху, получает урон босс, иначе персонаж. |
|  | Move() | Задаёт движение персонажа по определённой траектории. |
|  | ReceiweDamage(Vector3 poz) | Получение урона. |
|  | Dead() | Проверяет количество жизней босса, если равны 0, то загружает следующую сцену. |

## 3.7 Сообщения программисту

Во время игры никаких выводимых сообщений не предусмотрено. Программа работает только внутри своего интерфейса и полностью стабильна.

* + - * При тестировании, отладке или разработки в Unity, используются некоторые сообщения DebugLog(). Также могут появляться сообщения самого Unity.

# Руководство пользователя

Данное руководство пользователя содержит общую информацию о компьютерной игре “Grumpy Three Friends”, описание основных функций и возможностей программы, рекомендации по освоению, ответы на технические вопросы.

## 4.1 Общие сведения о пользовательском интерфейсе программы

Программа рассчитана на пользователей с базовым представлением работы с играми и приложениями.

## 4.2 Запуск компьютерной игры «Grumpy Three Friends»

Скачать игру можно по ссылке: <https://github.com/SaPeUlDi/Grumpy-Three-Friends/releases>

Для запуска игры необходимо:

* Чтобы все файлы находились в одной папке (Рисунок 1)

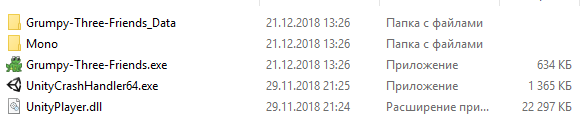


Рис 1 – Файлы, необходимые для запуска игры

Чтобы запустить игру, нужно дважды щелкнуть по ярлыку программы **Grumpy-Three-Friends.exe** с логотипом.

## 4.3 Работа с главным меню игры

В главном меню предложено пять кнопок: (Рисунок 2)



Рисунок 2 – Главное меню

Рассмотрим каждую из них:

## 4.4 Начать игру

При нажатии на эту кнопку начинается игра. Окно предыстории содержит в себе одну кнопку – левый клик мыши в любой области экрана для перехода к следующей части. (Рисунок 3)

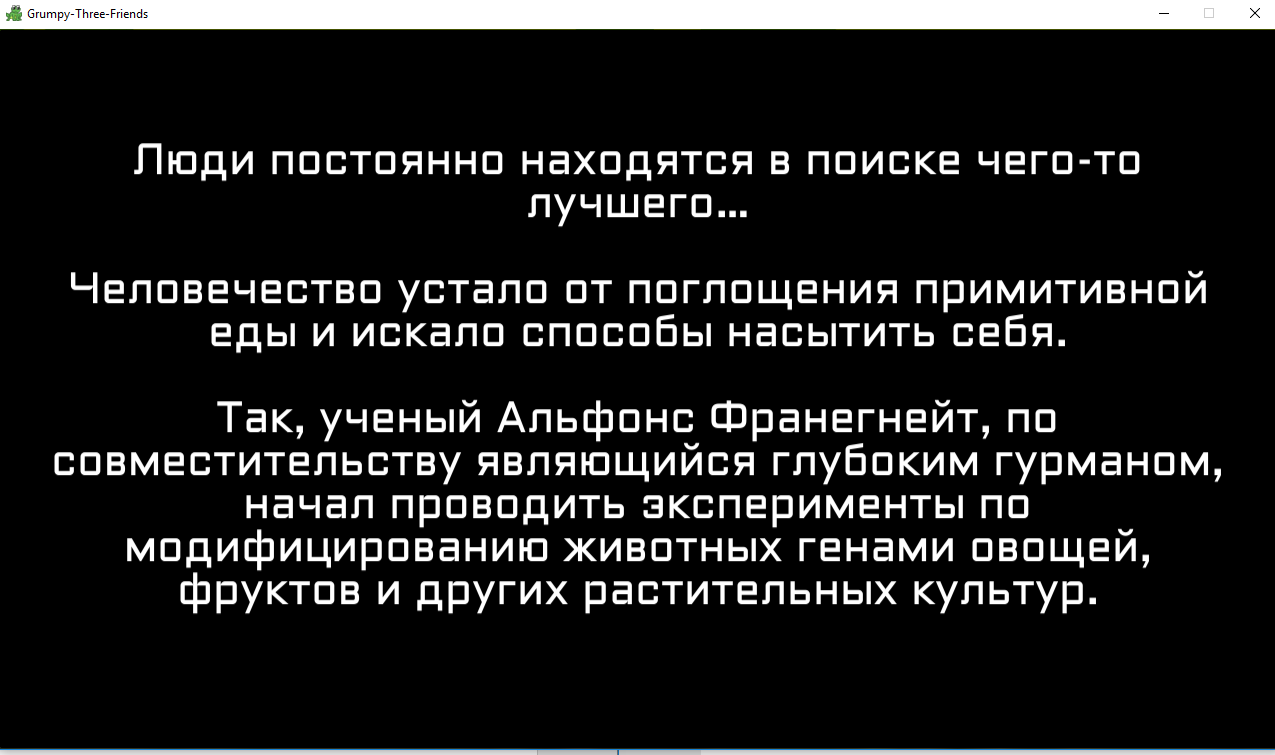


Рисунок 3 – Окно предыстории

Следующее за ним - окно катсцен, которое имеет такую же кнопку, однако область её активна лишь в ячейке диалога. (Рисунок 4)



Рисунок 4 – Окно диалога

После этого начинается непосредственно сам игровой процесс. Управление персонажем осуществляется с помощью кнопок:

* D – движение вправо,
* A – движение влево,
* Пробел/Space – прыжок,
* Esc – пауза.

Для прохождения уровня необходимо достичь его финала в правой части. На пути будут стоять ловушки как на земле, так и в воздухе. При соприкосновении с ловушкой отнимается одна жизнь. На третье соприкосновение уровень начинается сначала. Жизни восстанавливаются каждый раз в начале уровне вне зависимости от результата.

## 4.5 Настройки

Окно настроек игры позволяет изменять параметры игры. (Рисунок 5)

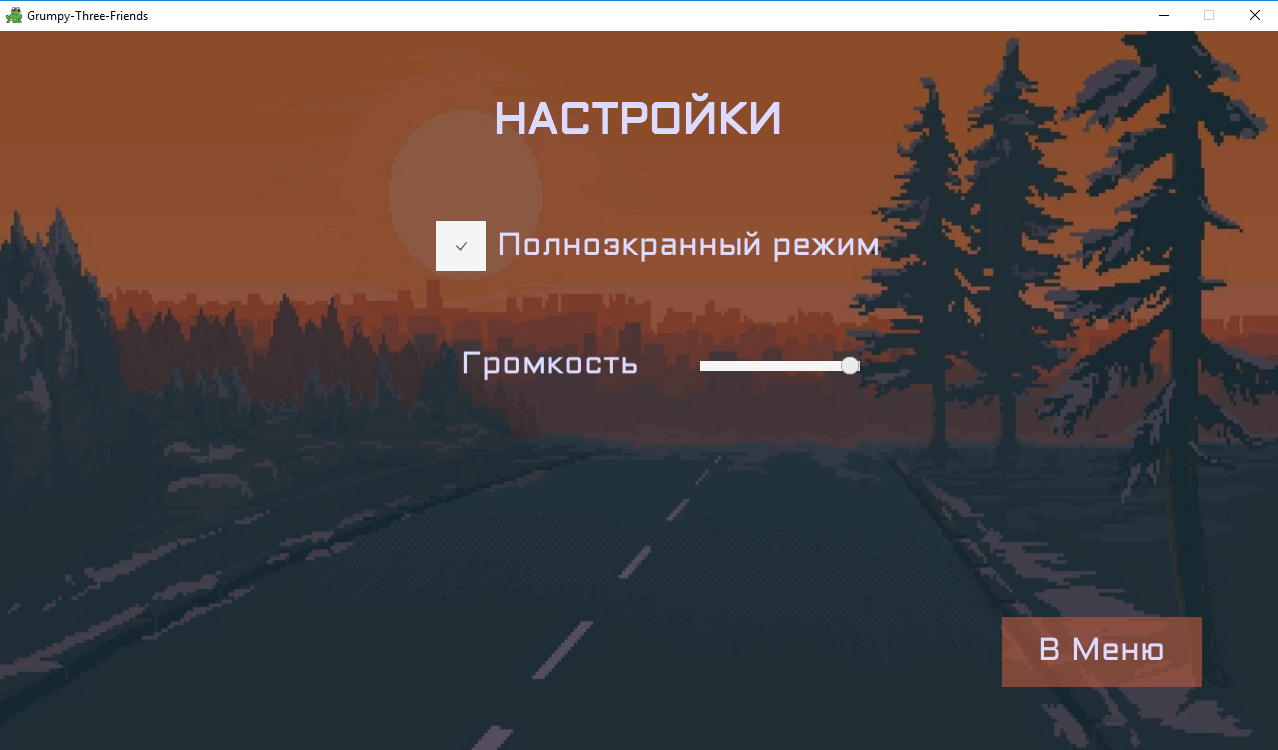


Рисунок 5 – Меню настроек

Можно включить или отключить полноэкранный режим, поставив или убрав соответствующую галочку.

Можно изменить громкость. Для этого нужно двигать соответствующий ползунок вправо или влево.

Ползунок, сдвинутый максимально влево будет означать, что громкость данного явления отключена совсем, а сдвинутый максимально вправо будет означать, что данное явление будет звучать на максимальной громкости.

Чтобы завершить сеанс работы с настройками и вернуться в главное меню, нужно нажать на соответствующую кнопку в правом нижнем углу окна.

## 4.6 Достижения

Достижения представляют собой прогресс прохождения игры. Стоит заметить, что при закрытии и повторном запуске приложения достижения сбрасываются. Это вызвано в связи с коротким прохождением сюжета. При закрытом состоянии иконка достижения окрашена в серые тона. После его получения она получает исконный цвет. (Рисунок 6)

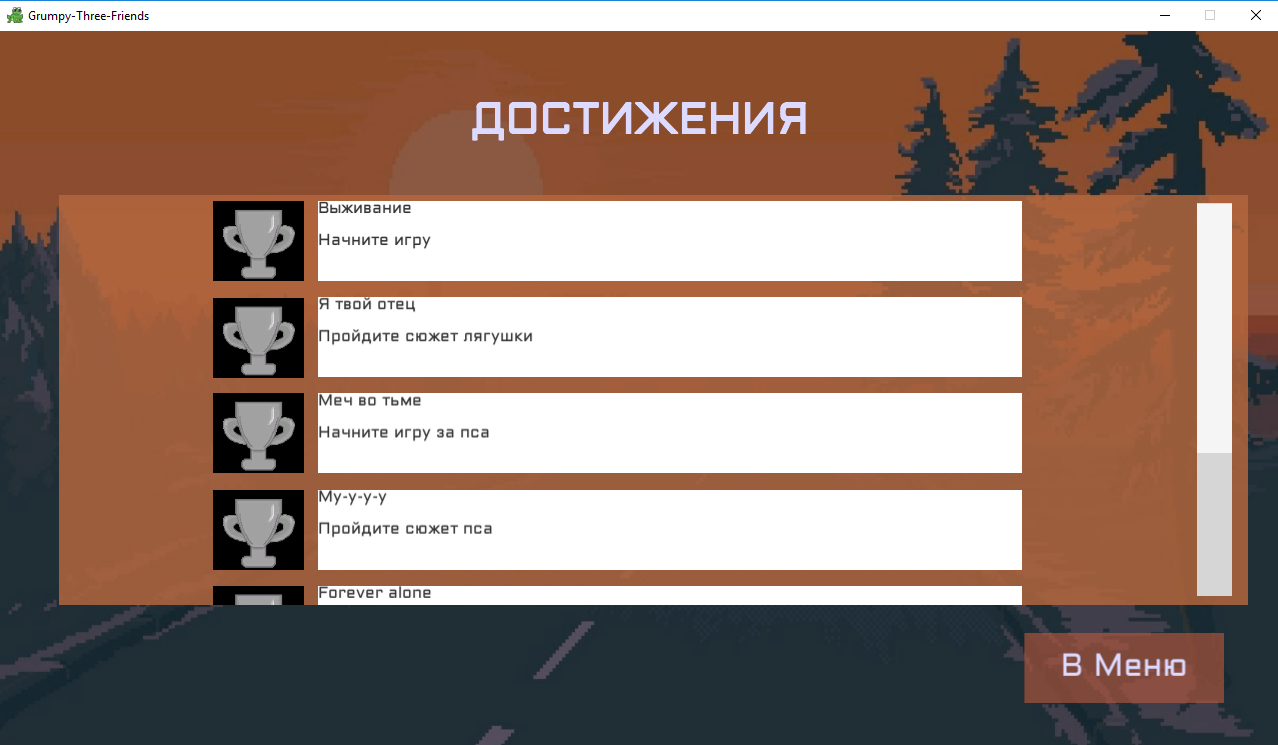


Рисунок 6 – Меню достижений

## 4.7 Разработчики

Меню разработчиков представлено для ознакомления с людьми, участвовавшими в процессе её создания.

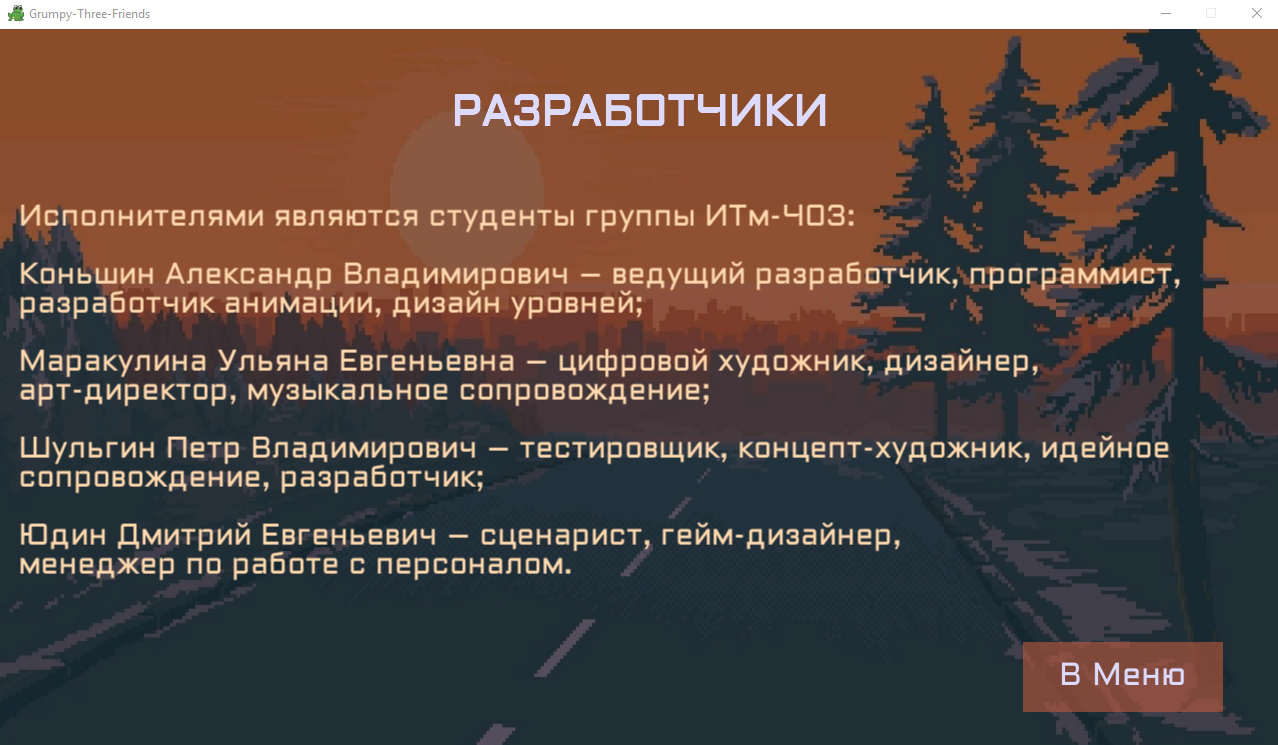


Рисунок 7 – Меню разработчиков

## 4.8 Выход

При нажатии на последнюю кнопку игра закроется.

## 4.9 Решение технических проблем

|  |  |
| --- | --- |
| Проблема | Решение |
| Игра не запускается | Игра разработана для операционных систем Windows 7,8,10. Проверьте соответствие операционной системы |
| Не удается продолжить выполнение кода, поскольку система не обнаружила UnityPlayer.dll | Проверьте папке игры наличие файла UnityPlayer.dll |
| Failed to load mono | Проверьте в папке игры наличие папки mono |
| Data Folder not found | Проверьте в папке игры наличие папки Grumpy-Three-Friends\_Data |
| В игре появляются черные зоны на экране | Обновите драйвер вашей видеокарты |

# Отчеты участников команды

## 5.1 Отчет программиста

**Программист** Коньшин Александр выполнил следующую работу:

1. Проработка настроек игры
   1. Задание оптимального начального разрешения.
   2. Последовательное объединение сцен (меню, уровни и т.д.) в проект для правильного создания билда (build).
   3. Запрет на произвольное изменение размеров рабочего окна.
2. Создание интерфейса игры
   1. Главное меню
      1. Кнопка «Новая игра» начинает игру с переходом к первому уровню.
      2. Кнопка «Настройки» отображает модуль настроек, где можно изменить уровень громкости звука, а также переключить режим из оконного в полноэкранный или обратно.
      3. Кнопка «Достижения» отображает модуль достижений, где показан список всех достижений с отметкой о получении каждого из них (желтый кубок).
      4. Кнопка «Разработчики» предоставляет информацию о разработчиках игры: ФИО, роль в команде.
      5. Кнопка «Выход» закрывает приложение и убирает его из списка действующих процессов.
   2. Игровые уровни
      1. Введен счетчик жизней персонажа (3 на каждый уровень).
      2. Добавлена возможность вызова паузы для приостановки игры и возможности выйти в главное меню.
3. Создание игровых уровней
   1. Несколько уровней с разными персонажами (лягушка, кошка, собака).
   2. Работает анимация движения персонажа.
   3. При помощи скриптов с дополнительной поддержкой компонентов движка Unity разработана физика управления персонажем, настроены коллизии для всех объектов, добавлен способ управления (перемещения) игрового персонажа.
   4. Построена логика цикличной смены фонового изображения при движении.
   5. Построена логика взаимодействия персонажа с поверхностями и ловушками.
4. Дополнительно
   1. Разработка
      1. Для каждой сцены (меню и уровней каждого персонажа) используется своя фоновая музыка
      2. Введена простая система достижений
   2. Тестирование
      1. Во время процесса разработки производились частые целевые тесты на наличие определенного бага, ошибки.
      2. Проводились периодические полные тестирования игрового процесса для каждой стабильной версии игры.

## 5.2 Отчет художника

**Художник** приложения Маракулина Ульяна выполнила следующую работу:

1. Проектирование схем и расположения интерфейса.
2. Разработка концепт-артов трех главных персонажей.
3. Создание спрайтов главных персонажей.
   1. Создание анимации для главных персонажей.
4. Работа с уровнями.
   1. Редактирование фонов
   2. Подбор музыкального сопровождения.
   3. Подбор шрифта.
5. Поиск анимированных врагов (ловушек).
6. Создание активных и неактивных иконок достижений.
7. Работа с главным меню.
   1. Подбор музыкального сопровождения.
   2. Подбор шрифта.

## 5.3 Отчет тестировщика

**Тестировщик** приложения Шульгин Петр выполнил следующую работу:

1. Тестирование приложения
   1. Тестирование игры через пользовательский интерфейс
      1. Тестирование прототипа игрового уровня.
      2. Выявление багов анимации.
      3. Обнаружение багов в столкновении.
      4. Выявление орфографических/лексических/граматических ошибок в текстах игры.
      5. Тестирование главного меню игры.
      6. Выявление ошибок в настройках игры.
      7. Тестирование первой версии игры.
      8. Тестирование функционала.
      9. Тестирование каждого игрового уровня на предмет багов, ошибок, неточностей, ошибок в тексте.
      10. Тестирование логики переходов с уровня на уровень.
      11. Тестирование второй версии игры.
      12. Тестирование уровня с боссом.
2. Тестирование пулл-реквестов (pull request)/исправленных ишью (issue)/выполненных карточек задач в СКВ Git Hub.
   1. Подготовка протестированных изменений к пулл-реквесту в ветку Release.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Описание теста | Ожидаемый результат | Полученный результат | Скриншот |
| 1 | Тестирование кнопок меню. | Все кнопки соответствуют содержанию открываемых им вкладок | Все кнопки соответствуют содержанию открываемых им вкладок | https://i.gyazo.com/62d1df344aa61e9f4ec4efc862e481ea.png |
| 2 | Тестирование переходов между уровнями | При завершении уровня, начинается новый. | При завершении уровня, начинается новый. |  |
| 3 | Тестирование коллизии | Стоя на платформе, персонаж не проваливается.  Пытаясь пройти сквозь стену, персонаж не проходит. | Стоя на платформе, персонаж не проваливается.  Пытаясь пройти сквозь стену, персонаж не проходит. |  |
| 4 | Тестирование получения урона от врагов | При касании врагов, отнимется одна жизнь. | При касании врагов, отнимется одна жизнь. |  |
| 5 | Тестирование смерти игрока | Когда у персонажа заканчиваются жизни или персонаж падает вниз, то начинается уровень сначала | Когда у персонажа заканчиваются жизни или персонаж падает вниз, то начинается уровень сначала |  |
| 6 | Тестирование выбора персонажа | Когда предстоит выбрать персонажа для битвы с боссом, то выбранный персонаж соответствует действительности | Когда предстоит выбрать персонажа для битвы с боссом, то выбранный персонаж соответствует действительности |  |
| 7 | Тестирование урона по боссу | Когда игрок касается босса сверху, то босс получает урон и теряет одну жизнь | Когда игрок касается босса сверху, то босс получает урон и теряет одну жизнь |  |
| 8 | Тестирование завершения игры | Когда у босса не остаётся жизней, загружается финальная сцена | Когда у босса не остаётся жизней, загружается финальная сцена |  |
| 9 | Тестирование паузы | Когда нажатии на клавишуESCAPE, игра приостанавливается, открывается меню паузы. При повторном нажатии на клавишу или кнопку «Продолжить» в меню паузы, то игра возобновляется | Когда нажатии на клавишуESCAPE, игра приостанавливается, открывается меню паузы. При повторном нажатии на клавишу или кнопку «Продолжить» в меню паузы, то игра возобновляется |  |
| 10 | Тестирование управления персонажа | При нажатии клавиши D, персонаж двигается вправо. При нажатии клавиши A, персонаж двигается влево.  При нажатии клавиши “Space”, персонаж подпрыгивает | При нажатии клавиши D, персонаж двигается вправо. При нажатии клавиши A, персонаж двигается влево.  При нажатии клавиши “Space”, персонаж подпрыгивает |  |

**Генератор-идей**:

1. Формирование идеи сюжета игры.
   1. Образное описание концепции игры.
   2. Описательное представление развития событий.
   3. Идейное описание персонажей и противников.
   4. Помощь сценаристу.

**Концепт-арт дизайнер**:

1. Разработка концепт-артов вражеских юнитов.

**Аниматор**:

1. Программирование анимации вражеским монстрам.

**Разработчик:**

1. Переработка уровней
   1. Оптимизация размеров объектов и их расположения в уровнях в соответствии с требованиями дизайнера.
   2. Оптимизация параметров персонажа (скорость/сила прыжка).
   3. Оптимизация текстовых сюжетных сцен в соответствии с требованиями сценариста, дизайнера.
   4. Оптимизация заднего фона трёх уровней.
2. Создание 2-х предпоследних уровней игры.
3. Реализация стилевого оформления.
   1. Смена стиля главного меню в соответствии с требованиями дизайнера.
   2. Смена стиля текста игры в --//--//--.

## 5.4 Отчет тимлида

Лидер команды Юдин Дмитрий выполнил следующую работу:

1. **Работа с командой**
   1. Распределены роли в команде в соответствии с их предпочтениями и навыками.
   2. Создание организации (organization) “SaPeUlDi”, а также её администрирование
      1. Создание команды (team) “UlPeDiSa” и назначение прав её участникам: write.
   3. Создание проекта (project) и его автоматизация (новые ишью (issue) в колонку “to do”, закрытые ишью в колонку “testing”.
   4. Одобрение (approve) всех ишью и пул-реквестов (pull request) в ветку “unstable”.
   5. Транспортировка из ветки “release” в ветку “master”.
2. **Ведение документации**
   1. Заполнение и проверка всех документов.
   2. Создание и ведении вики-страниц.
3. **Сценарист**
   1. Совместно с тестировщиком был создан на основе идейного сопровождения сюжет, мир и предыстория игры.
   2. Написаны диалоги (мысли) персонажей.
4. **Гейм-дизайнер**
   1. Создание достижений. Расписаны названия, а так же описания того, как их получить.

# Заключение

Компьютерные игры затягивают в себя современных людей, становятся частью их жизни, поэтому в связи с необходимости разработки новых компьютерных игр для поддержания их развлекательной функции была разработана игра, поставленные цели и задачи для которой были выполнены.

Проведено предпроектное обследование, в ходе которого разрабатывалась концепция проекта, определялась целевая аудитория, выбирался вид издания, средства разработки и составлялось техническое задание.

В ходе проектирования компьютерной игры был написан сценарий, спроектирована структурная схема проекта, составлена спецификация игровых объектов.

Для окончательной реализации проходили работы по подбору специализированного мультимедийного контента, описанию алгоритмов структурной схемы.

В итоге, данное приложение было оформлено в выбранном жанре за счет его потенциальной востребованности, а сценарии разработаны в соответствии с нормами целевой аудитории. Все требуемые приемы и навыки, полученные в течение семестра по данной дисциплине, продемонстрированы.

В ходе создания студенты ознакомились с некоторыми методами работы в среде Git, а также изучили соответствующие задачам каждого программные средства. Кроме того, был получен опыт по организации работы по создании видеоигр, включая разработку сюжета, концепции игры, определение дизайнерских решений.

# Список литературы

1. Обновление форка репозитория через интерфейс GitHub [Электронный ресурс]. – Режим доступа: http://bpteam.net/posts/2016-01-12-обновление-форка-репозитория-через-интерфейс-github.
2. Работаем с Git, основные команды для начинающих GitHub [Электронный ресурс]. – Режим доступа: http://dnzl.ru/view\_post.php%3Fid%3D317.
3. Введение - Первоначальная настройка Git [Электронный ресурс]. – Режим доступа: https://git-scm.com/book/ru/v1/Введение-Первоначальная-настройка-Git.
4. Основы создания 2D персонажа в Unity 3D 4.3. Часть 1: заготовка персонажа и анимация покоя [Электронный ресурс]. – Режим доступа: https://habr.com/post/211472.
5. Как создать внутриигровое меню в Unity [Электронный ресурс]. – Режим доступа: https://habr.com/post/346370.