МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ
им. ГУМАРБЕКА ДАУКЕЕВА»
Институт информационных технологии
Кафедра IT–инжиниринг

ДОПУЩЕН К ЗАЩИТЕ
Заведующий кафедрой
PhD, доцент
_____ А.У. Утегенова
«____» _____ 2021 г.

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Разработка игровых приложений для мобильных устройств под ОС Android

Специальность: 5В070400 – «Вычислительная техника и программное обеспечение»

Выполнил: Слаев Е.Н. Группа: ВТ-17-2
Научный руководитель: PhD, доцент Мамырбаев О.Ж.

Консультанты:
по экономической части: доцент Нұрпейіс Е.М.
_____ «____» _____2021 г.
по безопасности жизнедеятельности: доцент Имангалиева А.К.
_____ «____» _____2021 г.
по программному обеспечению: ст. преп. Жумагулова Ш.П
_____ «____» _____2021 г.
Нормоконтролер: ст. преп. Абсатарова Б.Р.
_____ «____» _____2021 г.
Рецензент: доцент Тойбаева Ш.Д
_____ «____» _____2021 г.

Алматы 2021

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ
им. ГУМАРБЕКА ДАУКЕЕВА»

Институт информационных технологии

Кафедра IT-инжиниринг

Специальность 5В070400 – «Вычислительная техника и
программное обеспечение»

ЗАДАНИЕ
на выполнение дипломного проекта

Студенту Слаеву Ердену Нурболатовичу

Тема проекта: Разработка игровых приложений для мобильных устройств под ОС Android

Утверждена приказом по университету № 25 от «18» февраля 2021 г.

Срок сдачи законченного проекта «___» _____ 2021 г.

Исходные данные к проекту (требуемые параметры результатов исследования (проектирования) и исходные данные объекта): Android Studio - среда разработки мобильной игры, C# – язык программирования, создание функций и объектов.

Перечень вопросов, подлежащих разработке в дипломном проекте, или краткое содержание дипломного проекта:

– аналитическая часть;
– проектная часть;
– экспериментальная часть;
– экономическая часть;
– безопасность жизнедеятельности;
– приложение А. Листинг программы.

Перечень графического материала (с точным указанием обязательных чертежей): графических иллюстрации 16, таблиц 8.

Основная рекомендуемая литература:

1 Г. Боканова Методические указания по выполнению экономической части дипломных работ Алматы, АУЭС, 2013 – 37с.

2 Прохорова О.Г. Безопасность жизнедеятельности в дипломных проектах. - Новомосковск, 2013.

3 Васильев А.Н. Программирование на C# для начинающих. Особенности языка, изд Бомбора 2018. - 528 с.

Консультация по проекту с указанием относящихся к ним разделов проекта

Раздел	Консультант	Сроки	Подпись
Технико-экономическое обоснование проекта	доц. Нұрпейіс Е.М.		
Безопасность жизнедеятельности	Доц. Имангалиева А.К.		
Программное обеспечение	Жумагулова Ш.П.		
Нормоконтролер	ст.преп. Абсатарова Б.Р.		

ГРАФИК
подготовки дипломного проекта

Наименования разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю	Примечания
Анализ предметной области		
Выбор программного обеспечения		
Сравнительный анализ аналогов		
Написание технического задания		
Создание UML диаграмм		
Реализация программного продукта		

Дата выдачи задания «___» _____ 20__ г.

Заведующий кафедрой _____ А.У. Утегенова

Научный руководитель проекта _____Мамырбаев О.Ж.

Задание принял к исполнению студент _____Е.Н. Слаев

Андатпа

Runner жанрындағы ойын болып табылатын Android операциялық жүйесіне арналған жергілікті мобильді қосымшаны құру үшін зерттеу және шешім табу, сонымен қатар негізгі әдістер мен функциялар менің дипломдық жобамда көрсетіледі. Бұл қосымшаны PlayStore мобильді қосымшалар дүкенінде сәтті жариялауға болады, өйткені ол барлық параметрлерге сәйкес келеді. Бұл бағдарлама үшін ең жақсы ойын қимылдары мен мобильді ойын жасау технологиялары талданды. Сондай-ақ жобаның экономикалық негіздемесі, оның орындылығын растайтын және қарастырылып отырған қосымшаны әзірлеу үшін еңбек талдауы жасалды.

Аннотация

Исследование и нахождение решения для создания нативногомобильного приложения для операционной системы Android, котроый является игрой в жанре Runner, а также основные методы и функии, будут изложены в моем дипломном проекте. Это приложение может успешно публиковаться в магазине мобильных приложений PlayStore, так как полностью соотвествует всем параметрам. Для этого приложения был проведен анализ лучших игровых движений и технологий создания мобильных игр. А так же составлено экономическое обоснование проекта, подтверждающее его целесообразность, и анализ труда для разработки рассматриваемого приложения.

Annotation

Research and finding a solution for creating a native mobile application for the Android operating system, which is a game in the Runner genre, as well as the main methods and functions, will be outlined in my graduation project. This application can be successfully published in the PlayStore mobile app store, as it fully meets all the parameters. For this application, an analysis of the best game movements and technologies for creating mobile games was conducted. As well as the economic justification of the project, confirming its feasibility, and the analysis of labor for the development of the application under consideration.

Введение

В современном мире, в мире технологий кажется не возможным даже представить человека без смартфона, планшета, компьютера. Мы привыкли иметь под рукой все эти гаджеты, без них уже жизнь покажется труднее, ведь наша повседневная жизнь тесно связана с нашими друзьями с исскустенным интелектом. Это средство общения, это тот же самый калькулятор, также это средство передачи информации ввиде видео или документов. В целом это мультемедийное устройство совршающее множество функций. Смартфоны с новой мобильной игровой платформой могут конкурировать с классическими портативными игровыми системами, такими как Nintendo DS или Playstation Portable.

Технология устройства смартфон довльно просто. Несколько отдельных блоков памяти являются состовляющими основу, также процессор и радиомодулькоторый, в своюочередь, состоит из передатчика и приемника и отвечает за связь. Интересное здесь- операционная система, которая находиться и установлена на внутренней памяти. Это как сердце системы, так как от нее зависит все возможности устпройства. ПК, а так же смартфоны имеюют различные операционные системы. Далее будет изложена и обсуждена все выше указанное.

Так как востребованность в мире растет каждый раз, то это требует постоянного обновления, и растет спрос на различные приложения. Каждая компания стремиться иметь свое приложение, для того чтобы быть клиентом и иметь всегда под рукой. На сегодняшний день трудно представить некотрые компании без специализированных программ. Это позволяет управлять на рынке в любой момент времени.

К большому сожалению, сегодня нет конкретных инструментов для разработки мобильных приложения. Каждый производитель хочет запомниться своей оригинальностью и уникальностью что иногда это может привести к сбоям в приложениях.

Сейчас компьютеры преобретают личный характер, дающий доступ в любое время и в любом месте, а мобильные телефоны уже как много лет имеют множетсво других функций кроме разговоров. Они используются для передачи данных, видео, различных документов. Могут также выполнять вычесления. Моюильные устройства могут стать новым покалением персональных компьютеров (ПК). Помимо этого, ожидалось от производителей, такие как ASUS, HP, DELL сделают консруктивные параметры, базирующиеся на OS Android.

В будущем IT - индустрия обещается вырости в размерах. Объем программных обеспечений будет широкий для мобильных приложений.

По моему мнению, тенденция роста открывает возможность к мобильным устройствам для традиционных языков програмирования. Следоватльно это дает огромный шанс в области мобильных приложений.

Таким образом, актуальность моего дипломного проекта ясна. Развтие и

поддержка приложения, которые основаны на ОС Android, будут востребованы на рынке програмного обеспечения.

Компьютерные игры- один из видов приложений. Игры, которые развивают, обучают, а также парой могут развлечь нас во время досуга. Под играми котрые развивают, я имею ввиду те приложения, которые готовят собственное мвышление, изобретательность, вооброжение, а конечно развитие творческих способностей. Спросите какие это игры? На этот вопрос ответить моя дипломная работа. Новизна состоит в том, что впервые выведена модель обучающего процесса для улучшения реакции и внимания игрока.

1 Аналитическая часть

1.1 Разработка игр

На данный момент в мире технологий видеоигры занимают особое место. Этот сегмент является одним из самых популярных в индустрии развлечений. И на данный момент индустрия игр сравнимо, например, с киноиндустрией. Кроме того, за последние пять лет индустрия видеоигр значительно обогнала киноиндустрию. Да и в целом структура игровой индустрии в целом уже давно выделилась среди видов развлечений.

Разработку игр нельзя рассматривать отдельно от индустрии видеоигр игр. Game dev или же разработка игр само по себе является лишь частью сложной "экосистемы", которая обеспечивает весь жизненный цикл производства, распространения и потребления продуктов, например, таких как компьютерные или же мобильные игры.

В современной структуре видеоигровой индустрии в данный момент можно выделить следующие уровни: платформы, игровые движки, разработка игр, публикация и эксплуатация, продвижение и потребление.

1.2 Описание игровых жанров

Компьютерные игры впервые увидели свет только в 50-60х годах прошлого века, но с тех пор индустрия видеоигр добилась огромного прогресса в своем развитии. Очень быстро очень примитивные и на вид незамысловатые программы превратились в новый вид искусства.

Стоит отметить, что единой системы классификации на сегодняшний день не существует, даже можно сказать, что современные игры сочетают в себе множество различных функций, которые уже едва вписываются в какие-либо конкретные рамки.

Журналисты, критики и игроки делят сами игры по платформам, визуальным представлением (место расположение камеры внутри игры, технология графического изображения), критериями публикации (бесплатные игры или же Free-To-Play, игры класса AAA, инди-игры и т.д.), а также количеством игроков, но наиболее популярной классификацией считается классификация по жанрам.

Важно понимать, что классификация игр по жанрам довольно условна. Во-первых, многие жанры не имеют четкого, общепринятого определения. Во-вторых, гейм-разработчики не прекращают свои эксперименты, разбавляя новые элементы для создания игровой механики. Другими словами, современные игры могут принадлежать к одному или нескольким жанрам одновременно или представлять собой что-то новое.

Для того, чтобы развеять любые сомнения по поводу решения классифицировать ту или иную игру в том или ином жанре, предлагаю систему

классификации, которую многие используют, где классификация игры в том или ином жанре осуществляется путем выявления доминирующих частей.

В итоге, основываясь на характеристиках рынка игр на мобильных платформах, я разделил лучшие игры для iOS и Android на следующие категории:

Action-adventure — это жанр игры, представляющий собой сочетание приключений и экшена. Наличие экшена это динамичность и интерактивность игры. В то время как приключенческая игра, сюжет, а также решение различных задач.

Приключенческий экшен очень похож на классическое приключение / миссию, но, как уже упоминалось выше, его отличает более высокая динамика геймплея, часто похожая на динамику шутер игры.

Шутер (англ. Shooter — стрелок) — игровой жанр, где больше внимания уделено перестрелкам, но у этого жанра есть свои отличия. Шутеры бывают от первого лица (FPS) или от третьего лица (TPS). Разница заключается в положении игровой камеры, которая фиксирует все события в игре. В играх от первого лица визуализация игры происходит глазами персонажа, что придает эффект присутствия. Для игры от третьего лица камера снимается сверху от героя, позволяя игроку видеть, помимо прочего, то, что происходит за спиной персонажа.

Слэшер (Slasher) в переводе с английского означает "Вырезать" или же "Рубить", что точно описывает основные действия героя слэшера во время игры. Главная задача персонажей игр в этом жанре-уничтожить огромное количество врагов с помощью холодного оружия.

Существуют и другие наименования игр с именами в этом жанре: hack and slash (англ. Hack — разрубать или же рубить, англ. Slash — резать, "резать и рубить"), hack and slay (англ. hack — резать и англ. slash — убивать, "резать и убивать").

Стелс — экшн (Stealth) с англ. скрытое действие — игровой жанр, персонажи которого должны постоянно прятаться, передвигаться тайком при выполнении различных заданий, в то время как другие этого даже не замечают, всячески избегая противников и не быть обнаруженными.

Персонаж стелс-экшена более уязвим и может легко погибнуть, отличие от других видов экшена, таких как шутер или слэшер. Это означает, что игроку недостаточно развить рефлексы и в первую очередь особое внимание следует уделить деталям, быстрому анализу ситуации и спасению жизни героя.

Иногда может потребоваться незаметно устранить врагов и спрятать тела, чтобы не сеять панику и не привлекать слишком много внимания к герою, ярким примером такого жанра служат такие игры как Assasin или Hitman. В чистом виде такие Stealth-action игры встречаются редко.

Жанр Survival / Horror в переводе с англ. выживание и главной отличительной чертой игр на выживание / ужасы является атмосфера, а не механика игры. В Survival / Horror геймплей может сочетать элементы многих других жанров, в то время как атмосфера всегда одна и та же - смесь

напряжения, тревоги и страха.

Антагонисты в Survival / Horror появляются неожиданно, вызывая постоянное напряжение у игрока. Движение часто происходит в условиях плохой видимости: в плохо освещенных помещениях, на улице, глубокой ночью и т. д.

Основная цель игроков Survival / Horror — выживание. Но поддерживать жизнь своего персонажа не так просто, потому что, в отличие от Shooter или Slasher, персонажи ужасов, как правило, имеют низкое здоровье или нехватку ресурсов.

Есть также игры, которые более точно предлагают игрокам механику выживания (Survival). И вся механика игры ориентирована на выживание и добычу ресурсов.

Shoot'em up – можно назвать прародительем классического жанра Shooter от первого и третьего лица. Игровая механика в этом жанре определяется тем, что игрок сражается против превосходных сил врагов, конечной целью которых является уничтожение всех противников.

Shoot ' em up имеет множество подгрупп, и самым распространенным шутер игрой являетя TDS (Top Down Shooter). Такие игры больше похожи на современные сторонние шутеры, чем в целом, есть сюжет разных, больших уровней. Основное отличие от TPS часто только одно - "вид сверху" и "верхний угол".

К жанру action RPG можно применить термин термин "ролевой боевик", что в целом будет верно. RPG (Ролевая игра) — жанр, в котором игроки управляют одним или несколькими персонажами, которые имеют определенные характеристики, а иногда и расы, которые отличаются друг от друга. Список характеристик может включать в себя здоровье, интеллект, силу, ловкость, а также различные другие способности персонажа. Во время игры характеристики героя обычно растут в зависимости от выполненного действия, увеличивается количество навыков, количество пассивных навыков так называемых бафов. Благодаря этому навыки растут и сложность игры тоже.

Результаты исхода игры в этом жанре зависит не только от характеристик и навыков персонажа, но нужны навыки самого игрока, так как реакция или анализ стратегии.

Пошаговые ролевые игры (или тактические ролевые игры) — это игры, в которых персонажи атакуют один за другим поочередно как в шахматах, отсюда и название жанра “пошаговое”. Часто в таких играх четко выражена стратегическая составляющая, которая почти полностью стирает грань между пошаговыми тактическими стратегиями (TBT) и пошаговыми РПГ (TRPG).

В пошаговом РПГ игрок управляет одним или несколькими героями, развивает их навыки, выбирает для них снаряжение или создает их, если вы игре предусмотрены такие функции. В ходе игры характеристики героя (или героев) обычно увеличиваются, количество и сила навыков и пассивных навыков увеличиваются.

Самое важное различие между пошаговыми РПГ и экшен РПГ

заключается в том, что успех зависит не от скорости реакции игрока, а от тактических и стратегических навыков пользователя в игре.

Roguelike (на сленге "рогалик") — это один из жанров компьютерных видеоигр, которые подразумевают жесткий игровой процесс. Такое название жанру было дано в честь прародителя — игры Rogue.

Большая часть игрового процесса "рогаликов" происходит по очереди, уровни и события, происходящие среди игроков, генерируются случайным образом. Часто смерть игрока приводит к концу игры и он должен перезапустить всю игру. Roguelike также означает абсолютную нелинейность, а также множество вариантов прохождения.

Классическая игровая механика стратегия в реальном времени (RTS) сводится к сбору ресурсов, строительству базы, его укреплению и атаке вражеских армий. Но не все стратегические игры следуют этим канонам игрового жанра.

Самой важной отличительной особенностью RTS является то, что для выполнения задания вам необходимо разработать тактику, стратегию боя, при этом все события игры происходят в режиме реального времени, а это требует от вас критического мышления и принятия определенного скоростного решения для игрока.

Этот игровой жанр часто имеет сочетание жанра экономических и пошаговых стратегий.

Пошаговые и тактические стратегии (TBS/TBT) отличаются от стратегий реального времени тем, что игроки движутся по очереди (тоже как в шахматах). Классическая игровая механика TBS включает в себя сбор ресурсов, строительство баз, атаку противников. Если в игре начинает доминировать взаимодействие с объектами, с учетом ландшафта и использования объектов или предметов, тогда такие стратегии непосредственно относят к тактическому жанру (ТБТ). TBT часто имеет ряд элементов в механике пошаговой стратегической игры, что размывает границы между двумя этими игровыми жанрами.

Одной из наиболее важных особенностей всех TBS/TBT является то, что успех зависит не от скорости реакции игрока, а от тактических и стратегических навыков пользователя в данной игре.

MOBA (Multiplayer Online Battle Arena - "многопользовательская онлайн боевая арена", ранее такие игры считались Dotalike) — жанр игр, основанный на стратегии в реальном времени (RTS) и элементах игровой механики на кастомной карте, созданный игроком Dota Warcraft III.

Две команды игроков сражаются на карте (обычно это сторона света и тьмы), каждая из которых дает вам преимущество. Главная цель в играх жанра MOBA strategies — полное уничтожение вражеской базы. От игр жанров RTS такие стратегии отличаются наличием одного особенного героя для всех участников игры, которые развивают и совершенствуют свои навыки за всю игру. Процесс игры прост тем, что игрок управляет только своим героем, а его юниты (союзные вспомогательные персонажи) в первую очередь управляются

компьютером.

Экономическая стратегия - жанр, в котором суть фокусируется на экономических процессах. В общем, игрок получает бизнес, такой как гостиница, кондитерская или магазин. Это может быть страна, деревня или определенный участок.

Игроку предстоит развивать объект, который принадлежит ему: заработок ресурсов, создание производства и торговля. Это требует грамотного расходования имеющихся ресурсов, приоритетов и планирования.

Основной целью экономических стратегических игр является увеличение ресурсов и увеличение прибыли.

Жанр Tower Defense подразумевает защиту или оборону своего объекта, например, замка или военной базы. Прежде всего игроку нужно знать, как держать оборону, грамотно расходовать материалы для укрепления и выбирать подходящий вариант для обороны, поэтому название жанра переводится как "оборонительные башни". Вы можете обороняться, используя различные защитные башни, которые строит игрок. Игрок сам выбирает позицию оборонительного сооружения.

Tower Offense это как Tower Defense, только игрок не обороняется, а атакует объект противника.

Игры в жанре Simulator имитируют реальность, другими словами, пытаются помочь игроку воспроизвести какой-то реальный опыт из реальной жизни.

Часто слово «симулятор» добавляют к названиям игр другого жанра, например, "симулятор водителя" или "симулятор снайпера", что неверно и делает бессмысленным отличать этот жанр от других жанров.

Поэтому в реальности симуляторами можно назвать только игры, воспроизводящие профессию или опыт обращения с различным оборудованием. Игры, которые действительно отвечают основным критериям жанра-симуляция реальности.

На самом деле не все игры классифицируемые, как "спортивные симуляторы" таковы, потому что спорт в реальной жизни отличается от спорта в игре, поиграв в симулятор боксера игрок не научится боксировать и не узнает о боксе ничего нового, а симулятор вод теля может научить как минимум некоторым функциям. Однако этот термин активно используется для всей категории игр. Однако в центре внимания спортивных симуляторов по-прежнему находится не воспроизведение реальности, а соревнование. Игроки обычно управляют одним персонажем или всей командой. Все действия выполняются в соответствии со строгими правилами на ограниченных территориях: ринг, беговая площадка, баскетбольная площадка или поле для бейсбола.

Гоночные игры или же Racing иногда называют автомобильными симуляторами. На самом деле, не совсем верно, что гоночные игры классифицируются как симуляторы, так как в таких играх не так много симуляторов реальности. Как и симуляторы, игроки управляют автомобилем, и

симуляция езды уходит, а соревновательная часть становится особенностью этой игры.

Auto, moto и bike trial-freestyle — это дочерний жанр гоночных игр, который включает в себя преодоление препятствий на разных транспортах.

Обычно в таких жанрах предусмотрены бонусы за различные трюки и опасные моменты, поэтому к названиям добавляют фристайл.

Особенностью Платформеров является наличие таких платформ, которые позволяют перемещать персонажей в разные точки игрового мира. Платформы представляют собой различные опоры, такие как островки земли, подвешенные в воздухе или дома. В общем, герою часто приходится совершать прыжки, чтобы преодолеть препятствия, скалы, прыгать с платформы на платформу.

В дополнение к преодолению препятствий, игрокам часто требуется уничтожать врагов, а затем собирать предметы или монеты, золото. Некоторые элементы могут дать персонажу различные способности на определенное время, например: герой на некоторое время прыгает дальше, быстрее бегает, становится неуязвимым и т.д.

Adventure/Quest (приключения/квест) — это жанр приключенческой игры, основными элементами которой являются сюжет, исследование мира, а также выполнение различных квестов. В таких жанрах обычно квесты состоят из головоломок.

Жанр может быть разделен по категориям. Миссии могут быть графическими, текстовыми и визуальными романами. Хотя это последнее, это больше похоже на словесное приключение.

В визуальных романах взаимодействие с игровым миром осуществляется через текст и с иллюстрациями. Однако интерактивность в этом жанре довольно низкая.

Графическая приключенческая игра имеет механику наведения и щелчка, что означает, что игрок может взаимодействовать с миром, а не просто читать текст, иначе он не считался бы игрой.

Beat'em up Fighting (с англ. борьба и избиение) — это два разных игровых жанра, но у них есть общий суть и цель. Цель состоит в том, чтобы победить одного или нескольких противников в рукопашном бою. Иногда сила может зависеть от выбранного оружия.

В жанре файтинга дуэль происходит на арене, пространство которого ограничено. Бой обычно идет один на один, но это не исключает возможности участия в матчах команды или вызовах бойцов-помощников.

В играх Beat'em up нет так называемых арен и у игрока стоит единственная задача — перебить всех своих врагов.

Puzzle —жанр, в которой основное внимание уделяется необходимости решения различных логических задач. Игровая механика основана на изучении определенных правил, создании шаблонов, поиске комбинации для достижения определенного результата. Это требует концентрации и использования логики.

Спорно, что Arcada — это жанр игр. Термин аркада появился еще давно, когда были популярны аркадные автоматы и в них можно было поиграть в игры

с простейшей механикой. В дальнейшем аркадными играми стали называть все игры, процесс прохождения которых был очень прост и логичен.

В настоящее время слово "аркада" используется как название жанра игры, как приставка, характеризующая сложность игры.

Аркадная игра — это игра, где главная цель состоит в том, чтобы пройти уровни и собрать бонусы за максимально ограниченное время. Легкое освоение — важный критерий для игры жанра аркада.

Раннер и бесконечные жанры беговых игр, которые фокусируются на бегущем герое. Почти всегда персонаж бежит автоматически, игроку нужно только менять направление бега или прыгать и уворачиваться от препятствий. Это требует, в частности, концентрации внимания и хорошей реакции.

Бесконечный бегун или же Endless Runner — это классический Runner, которая не имеет финиша и главной особенностью игры является рекорд. Игрок с каждым разом стремится побить новый рекорд. В таких играх почти всегда присутствуют игровые бонусы за определенные действия.

Кликеры или же на английском Idle Games — это жанр игр с простейшей игровой механикой. Название происходит от уникального звука мыши, когда вы нажимаете на нее. В мобильной интерпретации игрок должен касаться экрана мобильного устройства без остановки, то есть кликать по экрану. Каждое прикосновение к объекту на экране приносит игроку инструмент. Например, у игрока есть валюта. Он может использовать его для приобретения дополнительных инструментов или символов, которые позволяют автоматизировать сбор ресурсов.

Collectible Card Game в переводе с английского «Коллекционная карточная игра» — игровой жанр, история которого началась задолго до появления видеоигр.

В начале игры у игрока есть базовый набор карт, который может быть расширен с течением времени, они могут быть получены в награду или покупая. Другими словами, со временем у каждого игрока появляется уникальный набор карт, каждая из которых обладает уникальной силой. Все это делает простую карточную игру коллекционной.

Ритм-игры — это жанр аркадных игр, где музыкальная составляющая является как бы фундаментом или же основой игры и от игрока требуется иметь чувство ритма. На фоне музыкального сопровождения в таких играх игрок успевает нажимать на определенные участки игрового поля в соответствии с заданным ритмом.

Невзирая на все особенности геймплея таких игр, rhythm games являются одной из самых популярных жанров, благодаря огромному распространению в азиатских странах.

С развитием видеоигр настольные игры были заменены виртуальными компаньонами, которые со временем выделились в отдельную, новую категорию, имеющую свои отличительные особенности.

В настольных играх игроки ограничены игровым полем, все операции обычно выполняются шаг за шагом и некоторые настольные игры стали

настолько популярными и востребованными, что позже они стали отдельным жанром, как например карточные игры.

Песочница или на английском Sandbox – разновидность игр с нелинейным повествованием, недавно выделенной в отдельный жанр. "Песочницы" отличаются тем, что в них нет сюжетов и игрок может выбрать любой маршрут и любое доступное действие, исход игры будет зависеть от его выбора. "Смекалка" и "Добыча ресурсов" часто являются компонентами разных "Сандеров".

Sandbox игры могут быть экшенами от первого лица, платформами, стратегиями косвенного контроля. Все они имеют одну общую черту — у них нет определенной цели, вы можете изменить открытый мир, вы можете создать что-то свое или изменить имеющиеся объекты и процесс игры не ограничен временем.

Кроме того, элементы песочницы можно найти в GTA и симуляторе The Sims или же в Minecraft.

Детские игры рассчитаны в первую очередь на определенную возрастную группу игроков, что приводит к определенному количеству функций.

Детские игры имеют оптимальный уровень сложности для своей целевой аудитории и чаще всего преследуют достаточно конкретные цели — развитие и обучение, например игры с изучением алфавита, цифр, опазнования предметов и животных, знакомство с цветами и фигурами.

Жанры игр Augmented Reality (дополненная реальность) в последнее время набирают обороты и применяются в разработке компьютерных и мобильных игр. В App Store и Google Play уже есть много AR-игр и приложении для быта, как Ikea. Конечно, не все из них высокого качества, но с ростом технологии и вырастит качество.

Augmented Reality используют преимущества окружающей среды. Таким образом, мобильный телефон станет окном, через которое мы будем наблюдать существующий мир, дополненный другими элементами или же говоря иначе, окружающие нас реальные объекты дополняются наложенной них графикой.

Massively Multiplayer Online Game сокращенно MMO, что в переводе с английского значит "массовая многопользовательская онлайн игра".

MMO-Action — это жанр игр, который представляет из себя смесь экшен игр и MMO. На самом деле это могут быть симуляторы боевых кораблей, танковых сражений или морских сражений. Самое главное — это общий виртуальный мир, где события происходят без участия игрока самого игрока.

MMORPG (massively multiplayer online role-playing game) — которая переводится как образованная массово многопользовательской онлайн игры. massively multiplayer online role-playing game — это жанр игр, который представляет собой смесь массовой онлайн игры и ролевой игры. ММОРПГ отличается тем, что процесс игры проходит в общем виртуальном мире, где игрок может взаимодействовать с другим игроком в режиме реального времени.

Без взаимодействия с другими игроками изучение огромного

внутриигрового мира становится неинтересным и поэтому внимание уделяется общению, которое необходимо не только для решения различных задач, но и для повышения социального статуса игрока и прохождения заданий.

Обычно мобильные стратегии жанра MMO включают в себя базовую структуру атак на базы других игроков или же атак на других игроков в режиме PVP (player vs player) и поэтому большое внимание уделяется развитию своего персонажа.

1.3 Обзор всех мобильных платформ

Операционная система мобильных телефонов (ОС) — это важнейшая особенность современных девайсов, которая отличает смартфон от старых мобильных телефонов. Выбор мобильных телефонов и его операционная система в наше время считается главным фактором, которая дает оценку современным девайсам.

До сегодняшнего дня с момента появления мобильных телефонов и смартфонов было создано много операционных систем, такие как:

1 BlackBerry OS (RIM) — которая отличается от многих ОС и благодаря алгоритму шифрования AES данный продукт получил широкую популярность в Соединенных Штатах. Все исходящие и входящие данные защищены шифрованием и поэтому спецслужбы других стран не заинтересованы в этой операционной системе.

2 Symbian OS — операционная система, занимавшая большую часть рынка смартфонов к концу 2010 года. В начале первого десятилетия в операционной системе оставалась только 1 платформа: 60 серия, которая в основном используется в устройствах Nokia и в некоторых моделях Samsung, которые сейчас уже устарели.

3 Windows Mobile — ОС компании Microsoft, занимавшая самый большой сегмент рынка операционных систем для смартфонов с 1996 года и в наше время находится в стадии поддержки и разработки.

4 Разработка Windows Phone 7 от Microsoft в целях охвата рынка и былой популярности, которая очень сильно отличается от Windows Mobile.

5 Bada — на сегодняшний день, по сравнению с другими ОС считается относительно новой мобильной платформой. Bada разработана компанией Samsung и самым первым телефоном на этой платформе стал S8500 wave.

6 Linux — эта операционная система не получила широкого распространения на мобильных устройствах, но разработка считается перспективным направлением. ОС Linux в основном популярны в Азии и непосредственно распространяются в этой местности.

7 Palm OS является одной из самых популярных платформ, но мобильные телефоны Palm OS в сегодняшний день широко не используются. В 2007 году был выпущен последний смартфон Palm Centro.

8 Android — считается самой популярной операционной системой, которая обслуживает планшеты, смартфоны, электронные книги, часы,

смартбоксы, нетбуки и т.д. ОС основана на ядре Linux и был приобретен Google. Затем Google начал создание альянса Open Handset Alliance (OHA), который сейчас обслуживает его и занимается поддержкой и развитием. Android приложения созданы на основе языка программирования Java и Kotlin. Собственный набор для разработки Android позволяет системе использовать библиотеки и компоненты приложений на языке программирования C, Kotlin и Java.

9 iOS — мобильная ОС, созданная и разработанная американской компанией Apple, который увидел свет в 2007 году и создан был для iPhone и iPod Touch, а затем для таких устройств как iPad и Apple TV. Для Windows Phone и Google Android он не доступен, только на устройствах Apple.

10 Windows Phone 8 — является вторым поколением ОС Microsoft Windows Phone и впервые был выпущен 29 октября. Эта операционная система содержит интерфейс Proto-pumaja под названием Metro (современный пользовательский интерфейс). В Windows Phone 8 задействована новая архитектура Windows nt, которая используется в операционных системах Microsoft. Windows Phone 7. X версия не может откатиться до версии Windows Phone 8, а новые приложения, созданные для Windows Phone 8, не могут запускаться на Windows Phone X.

1.4 Описание ОС Android

ОС Android основана на платформе мобильных устройств Linux и разработана компанией Google Open Handset Alliance (OHA), которая позволяет создавать приложения на основе языка программирования Java, управляющие вашим устройством с помощью различных библиотек, разработанные Google. Вы также можете писать приложения на языке C или других языках в программе AND (Android Native Developer kit.1.5, опубликованного 30 апреля 2009 года. К основным изменениям разработки являются поддержка функции Bluetooth A2DP, запись и просмотр видео в режиме камеры, возможность автоматического подключения к Bluetooth-гарнитуре.

Первым устройством под управлением Android стал смартфон T-Mobile G1, разработанный компанией HTC, который был запущен в 2008 году 23 сентября. Вскоре несколько уведомлений последуют за другими производителями смартфонов, которые планируют выпустить устройства Android.

У Google есть много основных преимуществ, которые отличают устройства на платформе Android от аналогичных продуктов:

1 Полная открытость и доступность Android позволяет пользователю с помощью обычных стандартных API запросов получить доступ ко множеству основных функции девайса

2 Нарушение границ — вы можете связать информацию из Интернета с телефонными данными, такими как контактные данные или данные о географическом местоположении, чтобы получить новые функции.

3 Равенство программ — для Android нет особой разницы между главными приложениями телефона и сторонним программным обеспечением, можно установить программу для вызова по умолчанию скачав любую программу из маркета или же сменить заставку экрана.

4 Быстрая и простая разработка приложений — для этого потребуется всего лишь SDK чтоб создать и запустить приложения для Android, включая настоящий симулятор инструментов, а также расширенные инструменты отладки.

Вышеописанная гибкость ОС Android имеет свою цену, когда компании, которые предпочитают разрабатывать собственные пользовательские интерфейсы, постоянно ищут новые версии операционной системы и устройства, выпущенные несколько месяцев назад, устаревают и теряется актуальность, так как их уже не обслуживают и не создают новые версии приложении, позволяющие пользователям использовать новые функции ОС. Многие эксперты отмечают, что платформа основана на Java и поэтому преимущества и возможности операционной системы Linux не в полной мере используются на Android. Кроме того, он не использует популярные графические инструменты (Toolkit) и библиотеки (Ot или GTK) на платформе, которая вряд ли перенесет большое количество приложений для Linux с полной версии домашнего компьютера на мобильную платформу из-за отсутствия общего сервера и библиотеки изображений. Кроме того, сообщалось, что Google начнет удалять приложения на пользовательских девайсах по своему собственному усмотрению, если будут нарушены его условия использования.

Большая доля кода лицензирована Apache 2, а открытая и неограниченная лицензия позволяет свободно использовать исходный код для создания собственных систем. Однако система, чтобы быть совместимой с Android, должна начать идти в ногу с программами Android - процесс сертификации базовой совместимости со сторонними приложениями, созданными сторонними разработчиками.

Аналитики и эксперты прогнозируют хорошие коммерческие перспективы для ИТ-рынка Google Android, в принципе, для продуктов, основанных на программном обеспечении с открытым исходным кодом, что уже не является сенсацией и постепенно они занимают ИТ-пространство, вытесняют признанных лидеров, создавая конкуренцию, что само по себе может только положительно сказаться на восстановлении рынка.

1.5 Основные понятия и определения разрабатываемой игры

Runner — жанр компьютерной или мобильной видеоигры, в которой персонаж игры движется в каком-либо направлении, преодолевая полосы препятствия, перепрыгивая пропасти и устраняя врагов (если имеются).

Геймплей — процесс игры с точки зрения играющего. Геймплей включает в себя различные аспекты игры, технические элементы, такие как внутриигровая механика, определенные методы взаимодействия между игрой

и пользователем и т.д.

Игровой движок является центральным программным компонентом компьютерных видеоигр и других интерактивных программ, которые обрабатывают графику в режиме реального времени. Это обеспечивает базовые технологии, упрощает и ускоряет разработку и позволяет приложению работать на разных платформах.

Графический движок — это промежуточное ПО и его основной функцией является отображение или же визуализация (рендеринг) двумерной или трехмерной компьютерной графики.

Физический движок — промежуточное программное обеспечение, которое выполняет компьютерное моделирование физических законов реального мира в виртуальном мире с различной степенью приближения.

Язык сценариев (scripting language) — язык сценариев высокого уровня, который описывает действия, проходящие внутри системы. Разница между программами и сценариями довольно расплывчата. Сценарий — это программа, которая использует готовые программные компоненты.

По словам Джона Устерхаута, автора Tcl, языки высокого уровня можно разделить на языки системного программирования и языки сценариев. Последние также называются языками для склеивания или языками системной интеграции. Скрипты обычно интерпретируются и не компилируются, хотя языки программирования предназначены для JIT-компиляции (Just-in-time compilation в переводе с английского «компиляция на лету»). JIT — технология повышения производительности программных систем, в которых используется байт-код и этот же байт-код компилируется в машинный код во время работы программы, иначе говоря язык сценариев можно интерпретировать как специальный язык для расширения возможностей командной оболочки или текстового редактора и административных инструментов операционной системы.

Спрайт (Sprite) — является графическим элементом в компьютерной графике.

Спрайты — это рисунки, которые появляются на экране с аппаратным ускорением. До 90-х годов концепция спрайта распространялась на все двумерные символы, и после разработки мультимедиа и улучшения насыщенности цвета и его глубины было выпущено программное обеспечение и это ПО ввело общий интерфейс программирования для двумерной и трехмерной графики. Итак, теперь 3D изображения выводятся на экран так же, как и 2D изображения.

Полигональная сетка — это набор вершин, граней и рёбер, которые задают форму полигонального объекта в компьютерной графике и трехмерном моделировании.

Каркас — определяет влияние кости на пик полигональной сетки.

1.6 Описание разрабатываемой мобильной игры

По статистике и анализу популярных видеоигр игр мы можем наблюдать, что на сегодняшний день популярность на рынке имеют приложения, реализованные с простым визуальным стилем. Производительность современных смартфонов, конечно, позволяет прорисовывать красивую картинку с большим количеством маленьких деталей, но небольшой размер экрана приводит к наложению этих же объектов друг на друга. Это приводит к первому решению по разрабатываемому проекту и визуальная составляющая игры будет сведена к минимуму.

Статистика популярных игр показал, что сейчас конкурируют между собой 3 жанра: головоломки, аркадные игры и экшены. Для своего дипломного проекта я выбрал жанр Arcada с геймплеем Runner.

В конечном итоге мой проект должен быть предназначен для пользователей, использующих мобильные устройства на базе операционной системы Android. Игра предназначена для развлечения пользователей, но в первую очередь для развития таких навыков как внимательность, быстрая реакция и принятие быстрого решения в критических ситуации. Игра будет в визуальном плане трехмерной. Визуальная составляющая приложения реализована следующим образом: фон будет темно-оранжевым, текст белым и желтым. Этот стиль поможет оперативной системе не нагружать память устройства, что значительно поможет увеличить количество пользовательских устройств, на которых запущено приложение и позволит исполнителю данного проекта тратить больше времени на другие аспекты разработки, такие как рефакторинг кода из-за легкого выполнения приложения. Для пользователя такой подход к оформлению визуальной части игра будет иметь наименьшее негативное влияние на глаза, так как большое количество ярких изображений хорошо смотрится на экранах высокой четкости, но на экранах с низким разрешением быстро начинают утомлять глаза пользователя.

Игровой процесс дипломного проекта немного отличается от канонического Раннера, суть которой заключается в преодолении максимального расстояние обойдя все препятствия одним или двойным прыжком, которые будут появляться в процессе игры. В игре существует так называемые точки на поворотах и нужно реагировать на эти точки касанием на экран и за каждое касание игрок получает 1 очко, если игрок не преодолеет препятствие или не успеет отреагировать, игра заканчивается. Говоря иначе, главная цель игрока - это максимальное расстояние, набор монеток и как можно больше очков. С каждым разом уровень игры становится все сложнее, что требует большего внимания и быстрой реакции.

В моем приложении генерация игрового мира происходит автоматически благодаря префабам, т.е. игровые объекты (платформы, разрывы, препятствия и монеты) автоматически генерируются на определенном расстоянии впереди персонажа. Скорость передвижения персонажа продолжает увеличиваться, что затрудняет достижение высоких результатов. Также увеличение скорости не

позволяет игроку войти в ритм игры и выбрать идеальный момент для прыжка.

1.7 Язык программирования C#

C# является одним из многих языков программирования .NET. Он объектно-ориентирован и позволяет создавать повторно используемые компоненты для широкого спектра типов приложений Microsoft представила C# 26 июня 2000 года, а 13 февраля 2002 года он стал продуктом версии v1.0.

C# — это эволюция семейства языков C и C++. Однако он заимствует функции из других языков программирования, таких как Delphi и Java. Если вы посмотрите на самый базовый синтаксис как C#, так и Java, код выглядит очень похожим, но опять же, код тоже очень похож на C++, что намеренно. Разработчики часто задают вопросы о том, почему C# поддерживает определенные функции или работает определенным образом. Ответ часто коренится в его наследии C++. Последние языковые функции, такие как интегрированный запрос языка (LINQ) и асинхронное программирование (Async), не обязательно уникальны для C#, но добавляют ему уникальности.

Важным моментом является то, что C# является “управляемым” языком, что означает, что для его выполнения требуется среда выполнения .NET Common Language Runtime (CLR). По сути, когда выполняется приложение, написанное на C#, CLR управляет памятью, выполняет сборку мусора, обрабатывает исключения и предоставляет множество других служб, для которых вам, как разработчику, не нужно писать код. Компилятор C# создает промежуточный язык (IL), а не машинный язык, и CLR понимает IL. Когда CLR видит IL, он точно в срок (JIT) компилирует его, метод за методом, в скомпилированный машинный код в памяти и выполняет его. Как упоминалось ранее, CLR управляет кодом по мере его выполнения.

Поскольку для C# требуется CLR, в вашей системе должна быть установлена CLR. Все новые операционные системы Windows поставляются с версией CLR, и она доступна через Центр обновления Windows для старых систем. CLR является частью .NET, поэтому, если вы видите обновления для среды выполнения .NET Framework, она содержит библиотеку классов CLR и .NET Framework (FCL). Из этого следует, что если вы копируете приложение C# на другую машину, то на этой машине также должна быть установлена среда CLR.

Вместо библиотеки времени выполнения (например, API для ввода-вывода файлов, обработки строк и т. Д.), предназначенной для одного языка, .NET поставляется с библиотекой классов .NET Framework (FCL), которая включает в себя буквально десятки тысяч повторно используемых объектов. С тех пор все. Языки .NET нацелены на CLR с одним и тем же IL, все языки могут использовать FCL. Это сокращает кривую обучения для любого разработчика, переходящего от одного из них. Чистый язык к другому, но также означает, что Microsoft может добавить гораздо больше функций, потому что существует только один FCL, а не отдельная реализация общих функций в каждом языке

программирования. Аналогично, сторонние поставщики программного обеспечения могут писать управляемый код, который может использовать любой разработчик .NET, независимо от языка. В дополнение ко всем службам, которые вы ожидаете от библиотеки времени выполнения, таким как коллекции, ввод-вывод файлов, сеть и т. Д., FCL включает API для всех других технологий .NET, таких как настольные и веб-разработки.

C# — это всего лишь язык программирования. Однако, поскольку C# нацелен на CLR и имеет доступ ко всему FCL, вы можете многое сделать. Чтобы получить представление о возможностях, откройте FCL и посмотрите на доступные технологии. Вы можете писать настольные приложения с помощью Windows Presentation Foundation (WPF) и консольных приложений. Для Интернета вы можете написать ASP.NET приложения. Когда вам нужно получить доступ к данным, есть ADO.NET Entity Framework и LINQ. Некоторые из новейших технологий Microsoft включают Windows Store и Windows Phone. Вы также можете создавать масштабируемые облачные приложения с помощью Windows Azure. Конечно, это лишь некоторые из доступных технологий, и как язык программирования общего назначения, вы можете сделать гораздо больше, чем это с помощью C#.

1.8 Среда разработки

В процессе создания мобильной игры нужно было выбрать язык программирования, средство взаимодействия с анимацией и самое главное – среда разработки.

Из игровых движков на данный момент самым популярным и легким в освоении является Unity, на котором я и остановил свой выбор. Unity движок включает в себя: движок отображения («визуализатор»), физический движок, звук, систему сценариев, сетевой код, управление памятью устройства и многопоточность. При создании игры можно сэкономить время и деньги благодаря функции повторного использования одного игрового движка как фундамент, для клонирования на его основе других игр.

Основываясь на решении использовать готовый игровой движок, необходимо выбрать язык программирования сценариев.

Для создания анимации инструмент может использовать автоматические и ручные типы анимации. Автоматическая анимация включает в себя создание ключевых кадров на определенных формах, объектах, и добавляется движения и автоматическое заполнение промежуточных кадров. Автоматическая анимация используется для создания анимации каркаса игры (т.н. каркаса). Для ручной анимации можно использовать различные графические редакторы, такие как Adobe Animate, Photoshop и другие. Ручная анимация создается покадрово вручную, для удобства можно использовать графический планшет.

1.9 3D модели для Unity

Для моего проекта 3D модели были взяты из бесплатного сервиса, где пользователи размещают свои модели.

Формат файлов Autodesk FBX — это популярный формат обмена трехмерными данными, который используется между трехмерными редакторами и игровыми движками. Изначально формат был создан как файл для инструмента захвата движения Filmbox от Kaydara. Имя и расширение формата FBX происходит от имени приложения FilmBox. В 2005 году Кайдара выпустил общедоступный SDK для закрытого и проприетарного формата файлов FBX и провел согласованную PR-кампанию, чтобы стимулировать принятие файлового формата FBX для высококачественного обмена 3D-данными между различными инструментами.

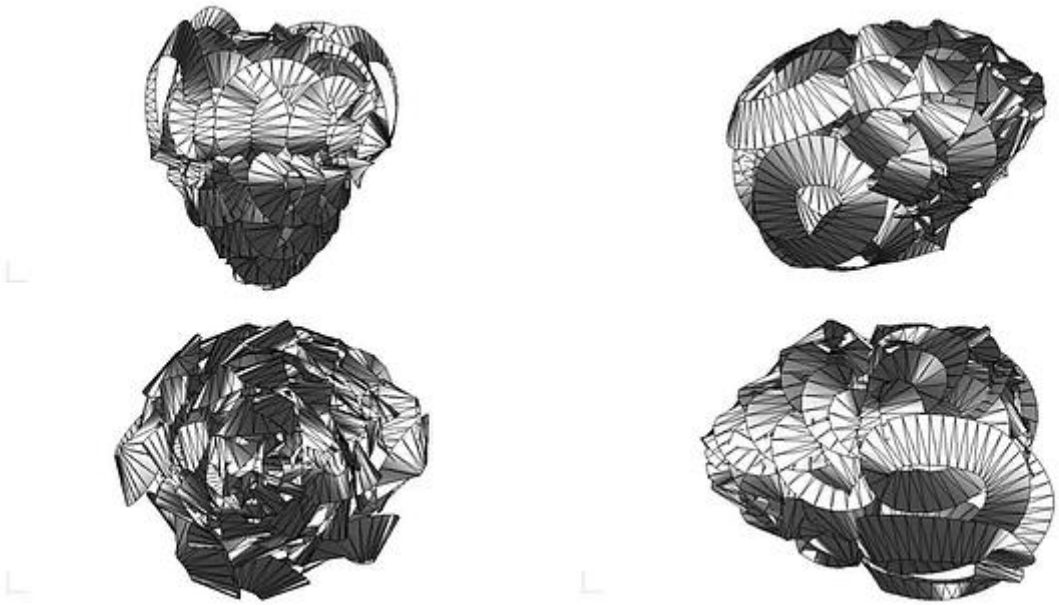


Рисунок 1.1 3D модель формата FBX

Filmbox в конечном итоге был переименован в MotionBuilder и позже был приобретен Autodesk.

Альтернативными популярными форматами файлов 3D-моделей и сцен являются форматы Wavefront OBJ 3D и Khronos Group glTF 3D, но в моем дипломном проекте был использован формат FBZ.

Преимущества формата FBX:

Формат fbx является золотым стандартом, который поддерживает разнообразие данных в этом формате. FBX поддерживает 3D-модели, иерархию сцен, освещение материалов, анимацию, кости, сдирание кожи, смешанные формы. Fbx, как более старый формат, также поддерживает данные, которые сегодня широко не используются, такие как поверхности шишек и кривые. Если вы выберете fbx в качестве формата обмена, вы знаете, что он, скорее всего, будет представлять необходимые данные для свойств, связанных с геометрией.

Преимущество формата файлов FBX, который также разделяет формат 3D-модели OBJ, заключается в том, что он позволяет хранить как данные о местоположении, так и данные УФ-излучения и обычные данные с различными топологиями. Он идеально подходит для высококачественных инструментов моделирования, а также позволяет использовать сложные функции, такие как точное разделение поверхности. Это, однако, немного затрудняет загрузку этих моделей в игрушечные движки, так как эти данные должны быть обработаны заранее, чтобы унифицировать всю топологию.

Формат файла FBX является быстрым и эффективным, поскольку он использует двоичный формат. Это связано с тем, что, когда дело доходит до хранения данных в двоичном формате, быстрее записывать, а затем читать их, потому что текстовый формат должен быть преобразован в двоичные данные с помощью удобочитаемых чисел и наоборот. Формат файла FBX также эффективен для определения местоположения, поскольку двоичное представление чисел занимает меньше места, чем читаемые человеком числа. Двоичные файлы часто усложняются, когда вам нужно читать, а затем записывать в них напрямую, но FBX SDK полностью скрывает этого пользователя fbx SDK.

Если вы являетесь разработчиком, использующим язык, поддерживаемый SDK, интеграция формата файла FBX в приложение очень эффективна и удобна. Процесс импорта и экспорта дополнения FBX-это относительно простой и простой процесс, включающий в себя соединение SDK и использование его API для ввода и вывода данных. Вся сложность формата файла скрыта от разработчика программного обеспечения, интегрирующего SDK. Вам не нужно беспокоиться, поддерживает ли SDK все функции FBX, потому что это официальный, только FBX SDK.

Отчасти потому, что SDK гарантирует, что вы можете читать все предыдущие версии формата файлов fbx, он гарантирует, что последние версии устройства правильно читают файлы FBX, созданные другими устройствами. Даже если формат файла FBX изменится, SDK гарантирует, что как старый, так и новый формат могут быть прочитаны с помощью различных путей кода, прозрачных для пользователей SDK. Таким образом, FBX не полон проблем совместимости, которые повлияли на аналогичные сложные форматы, такие как COLLADA (.dae). Если FBX правильно поддерживает эту функцию, вы можете передавать эту информацию своим приложениям без каких-либо проблем.

Соответственно, FBX можно использовать для легкой передачи сложных данных 3D-сцены между Clara.io , 3ds Max, Maya, Unity 3D и Unreal Engine.

1.1.1 Обзор игровых движков

Существует три самых популярных и конкурентоспособных игровых движка Unity, UDK и Cry ENGINE. В создании моей игры была задействована программа Unity, поэтому рассмотрим его сначала и сравним с другими.

Unity — это игровой движок с широким спектром функций, удобным, дружественным интерфейсом. Его главным преимуществом является мультиплатформенность, что означает, что проекты могут быть легко и быстро перенесены на такие платформы, как Android, iOS, Windows Phone 8 или BlackBerry. Кроме того, вы также можете использовать Unity для разработки игр PS 3, Xbox360, Wii U, браузеров.

Графический редактор Unity может выполнять очень ограниченное количество действий. Вы не можете моделировать в нем. Кроме основной работы с примитивами. Однако Unity легко интегрируется со сторонними 3D-редакторами (3D Maya, 3ds Max, Softimage, CINEMA 4D, Blender и т. Д.).), Что означает, что нет проблем с чтением различных форматов. После выпуска Unity 4.3 появилась возможность поддерживать 2D-графику, спрайт и 2D-физику, поэтому вы можете использовать движок для создания графики для 2D-игр.

Существует две версии Unity: бесплатная и профессиональная. Версия engine pro стоит 1500 долларов или 75 долларов в месяц. С глобальным освещением, текстурным дисплеем, буровыми установками Mecanim IK и т. Д. В бесплатной версии Unity есть водяные знаки, от которых очень проблематично избавиться.

Unreal Development Kit или UDK — это бесплатная версия движка Unreal Engine 3, написанного Epic Games, который используется для разработки многих игр. Этот движок обладает высокими графическими возможностями и может быть использован для разработки мобильных игр. UDK, в отличие от Unity, имеет свой собственный мощный инструмент для проектирования игровых уровней непосредственно в движке.

В частности, для создания FPS был разработан движок Unreal Engine. Unreal имеет свой собственный объектно-ориентированный язык программирования для написания сценариев, аналогичный Java или C++.

UDK, как и Unity, работает на различных платформах, включая iOS, Android, Windows Phone 8, Xbox360, PS 3, Playstation Vita, Wii U.

Как и в случае с Unity, вам нужно заплатить UDK только тогда, когда проект будет выпущен. Когда проект будет опубликован, с вас будет взиматься лицензионный сбор в размере 99 долларов США. Кроме того, если прибыль игры превысит 50 000 долларов, вам придется заплатить 25%.

CryEngine — это движок, разработанный компанией Crytek, который был впервые представлен в первой части Far Cry. С помощью этого движка мы можем создавать игры для ПК и консолей, включая PS4 и Xbox One. Не вдаваясь в подробности, я бы сказал, что графические возможности CryEngine значительно превосходят возможности первых двух двигателей. CryEngine был использован для разработки Ryse, Сына Рима. Этот движок, как и UDK, имеет интуитивно понятные, мощные функции для проектирования уровней.
Несмотря на то, что CryEngine является одним из трех самых мощных движков в мире, для его понимания требуется определенное количество времени.

1.1.2 Подробнее о платформе Unity

Unity – межплатформенная среда разработки компьютерных игр. Unity позволяет создавать приложения, работающие под более чем 20 различными операционными системами, включающими персональные компьютеры, игровые консоли, мобильные устройства, Интернет-приложения и другие.

Выпуск Unity состоялся в 2005 году и с того времени идёт постоянное развитие. Изначально Unity предназначался исключительно для компьютеров Mac, но потом постепенно выходили обновления, позволяющее работать под Windows и другие ОС.

Основными преимуществами Unity являются наличие визуальной среды разработки, межплатформенной поддержки и модульной системы компонентов. К недостаткам относят появление сложностей при работе с многокомпонентными схемами и затруднения при подключении внешних библиотек.

На Unity написаны сотни игр, приложений и симуляций, которые охватывают множество платформ и жанров. Вместе с тем Unity используется как крупными разработчиками, так независимыми студиями.

Редактор Unity имеет простой Drag&Drop интерфейс, который легко настраивать, состоящий из различных окон, благодаря чему можно производить отладку игры прямо в редакторе. Движок поддерживает два скриптовых языка: C#, JavaScript (модификация).

Проект в Unity делится на сцены (уровни) – отдельные файлы, содержащие свои игровые миры со своим набором объектов, сценариев, и настроек. Сцены могут содержать в себе как, собственно, объекты (модели), так и пустые игровые объекты – объекты, которые не имеют модели («пустышки»). Объекты, в свою очередь содержат наборы компонентов, с которыми и взаимодействуют скрипты. К объектам можно применять коллизии (в Unity т. н. коллайдеры – collider). В редакторе имеется система наследования объектов – дочерние объекты будут повторять все изменения позиции, поворота и масштаба родительского объекта. Скрипты в редакторе прикрепляются к объектам в виде отдельных компонентов.

При компиляции проекта создается исполняемый(.exe) файл игры (для Windows), а в отдельной папке – данные игры (включая все игровые уровни и динамически подключаемые библиотеки).

1.1.3 Описание игрового приложения

В последнее время находиться онлайн для многих людей стало одним из обязательств или же потребностью. Для этого необходимо носить с собой повсюду свои гаджеты, такие как смартфоны, планшеты и другие мультимедийные устройства. Так как, на сколько мне известно, интернет повсюду и это не составляет никакой проблемы подключаться и быть в сети в любое время суток. Следовательно, есть связь с интернетом, есть гаджеты для подключения, что ведет к спросу различных приложений, ну и игр для досуга.

Что касательно игр, в наше время разрабатываются игры не только для развлечений, но и для обучения. Время технологий дает обширный выбор разных игр, и возможность для разработчиков придумывать что-то новое так как есть спрос. Если обратить внимание, на рынке есть не только игры от известных производителей, а также много незнакомых разработчиков, которые имея достаточно знаний могут привлечь к себе своих потребителей. А также мне бы хотелось дополнить, что игры, которые выглядят просто или меньше, они от этого хуже не становятся, наоборот могут быть интересными и востребованными.

Главная цель моего проекта – это разработка игры для мобильных устройств под операционной системой Android на Unity. В этой разработке используется язык программирования C# и инструментальное программное обеспечение, также среда разработки Visual Studio и редактор для скелетной анимации.

1.1.4 Назначение программного продукта

Моей основной целью создания этой игры является отлично провести время, при этом провести его с пользой. Главная цель игры помочь отлично скоротать время. В каждый момент скуки время летит очень быстро. Кроме того, игра значительно повышает атмосферу, поэтому она надолго наполняет положительными эмоциями. Используйте игру, чтобы улучшить свое мышление. Игра учит вас анализировать свою деятельность. Это повышает усидчивость и настойчивость. Ребенок может улучшить навыки распознавания объектов, уровень владения иностранным языком, так как интерфейс на английском языке. Реакция будет лучше, потому что вы должны принимать быстрые решения в игре. Игра "Runner Man", в которой вам нужно преодолевать препятствия и вовремя реагировать на повороты.

1.1.5 Комплекс технических средств

Минимальные системные требования для запуска игры Runner Man:
- Android 4.5 или выше;
- Двухядерный процессор (1.2 GHz) или лучше;
- оперативная память 1 GBили выше;
- свободное место на телефоне 50 Мб или выше.

2 Проектная часть

2.1 Создание проекта

Ядро геймплея выглядит довольно простым и не привязанным к движку, потому начали с проектирования в виде C# кода. Похоже что можно выделить отдельное ядро базовой логики. Вынесем его в отдельный проект.

Юнити работает с C# решением и проектами внутри немного непривычно для обычного .Net разработчика, файлы .sln и .csproj генерируются самим Unity и изменения внутри этих файлах не принимаются к рассмотрению на стороне Unity. Он их просто перезапишет и удалит все изменения.

Приступаем к создание нашего проекта и запускаем Unity

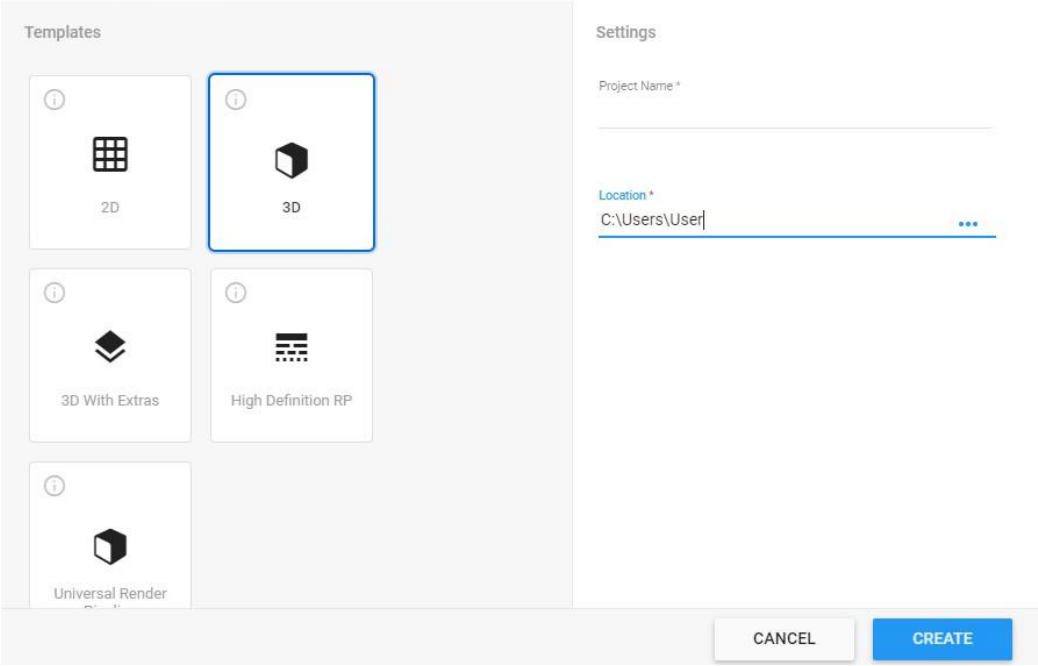


Рисунок 2.1 – Создание проекта в Unity 2019

Проект содержит игровой объект AppInfo, в котором вы можете заполнить важные метаданные, связанные с приложением, такие как идентификатор AppStore и идентификатор пакета. Эти значения будут использоваться для таких функций, как кнопка "Оценить нас" и открытие страницы Facebook или Twitter.

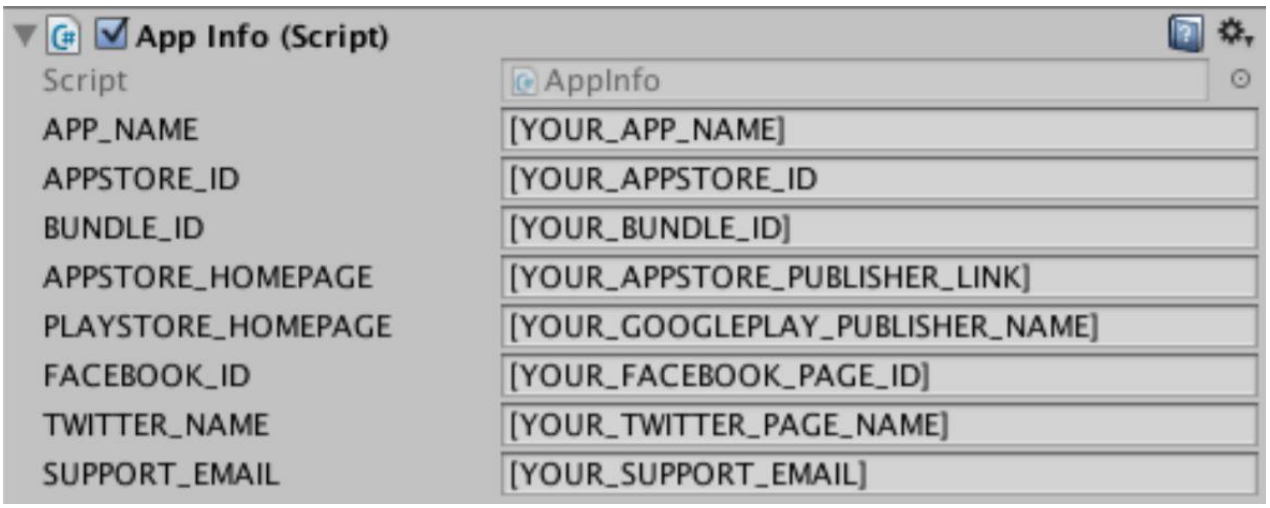


Рисунок 2.2 – Окно информации

2.2 Настройка геймплея

Большинство важных параметров игрового процесса можно настроить в компоненте GameManager, который прикреплен к игровому объекту, также названному GameManager в иерархии.

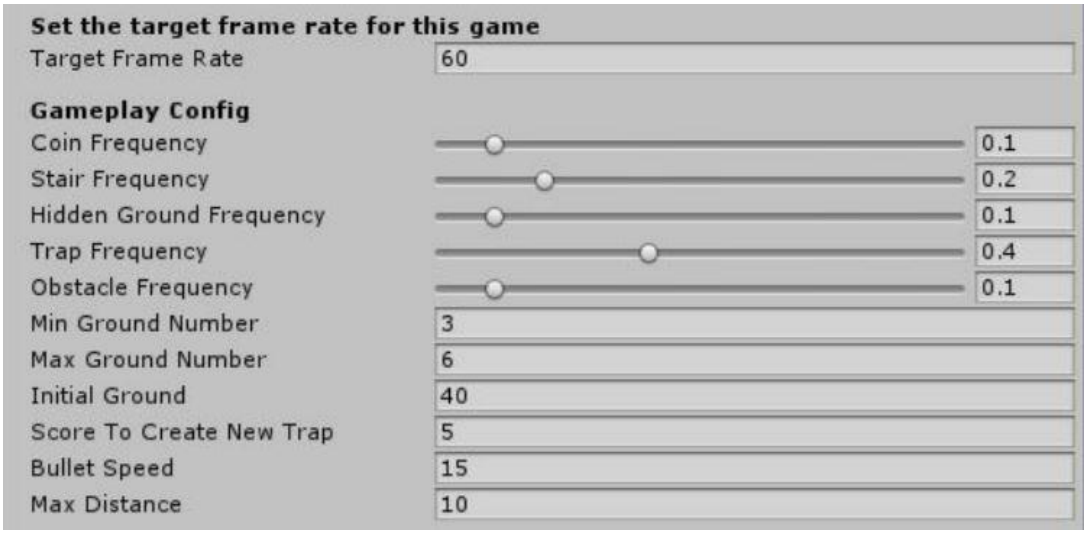


Рисунок 2.3 – Компонент GameManager

Можно настроить игровой процесс, изменив следующие переменные:

targetFrameRate: целевая частота кадров для игры, которая должна составлять не менее 60 кадров в секунду для игр, требующих плавного и быстрого движения.

CoinFrequency: вероятность появления “золота” (или монеты).

StairFrequency: вероятность появления “Лестницы”.

HiddenGroundFrequency: вероятность появления скрытых оснований (пробелов).

TrapFrequency: вероятность появления ловушек.

ObstacleFrequency: вероятность появления препятствий.

MinGroundNumber & MaxGroundNumber: минимальное и максимальное заземление номер на дорожке. Фактическое число земли на пути рандомизируется между этими двумя значениями.

Initial Ground: начальное количество площадок (включая ловушки, скрытые площадки).

ScoreToCreateNewTrap: количество типов ловушек будет увеличиваться на 1 при каждом достижении значения, делимого на это значение.

BulletSpeed: скорость пули ловушки тора.

maxDistance: расстояние между игроком и огненным шаром

2.3 PlayerController

Объект Player в иерархии содержит компонент PlayerController, в котором можно настроить поведение игрока (главного персонажа).

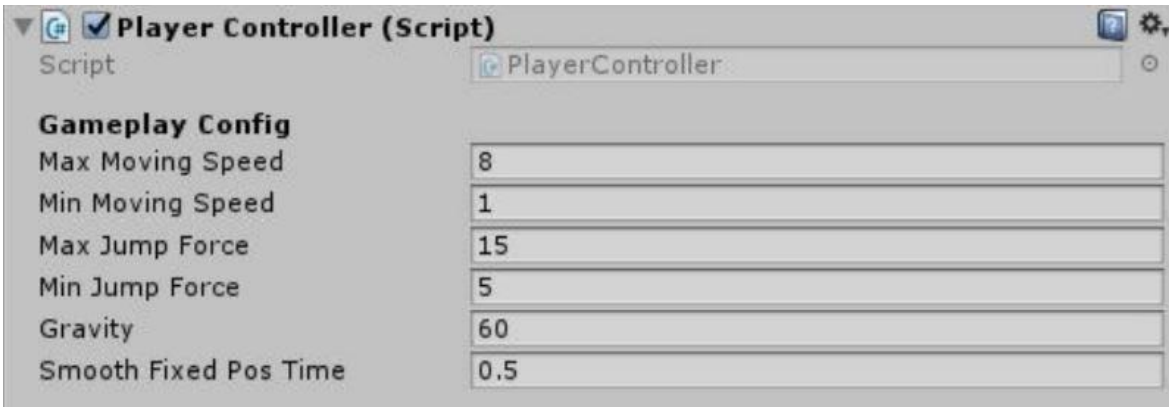


Рисунок 2.4 – Компонент PlayerController

MaxMovingSpeed: максимальная скорость перемещения игрока, это также нормальная скорость перемещения.

MinMovingSpeed: минимальная скорость перемещения игрока, это скорость, с которой игрок попадает в SpeedDownTrap.

MaxJumpForce: максимальная сила прыжка игрока, это также нормальная сила прыжка.

MinJumpForce: минимальная сила прыжка игрока, это сила прыжка, когда игрок попадает в SpeedDownTrap.

Гравитация: гравитация, которая удерживает игрока.

SmoothFixedPosTime: время для фиксации позиции игрока при каждом повороте игрока. Чем больше значение, тем более плавно фиксируется позиция игрока.

2.4 Добавление дополнительных характеристик

Добавим больше параметров, для этого выполним следующие простые шаги:

1 Перейдем в раздел Assets/Prefabs/Game/Characters и продублируем один из доступных префабов персонажей.

2 Изменим имя сборного модуля на предпочтительное.

3 Заменим сетку в компоненте MeshFilter дочерних элементов на сетку

4 новой модели.

5 Заменим материал в компоненте MeshRenderer дочерних элементов на новый материал персонажа.

6 Введем имя персонажа и цену компонента персонажа. Установим флажок isFree, чтоб выдать этого персонажа бесплатно (он будет автоматически разблокирован).

7 Изменим размер массива символов в игровом объекте CharacterManager, затем перетащим в него новый символ и нажмем Apply, чтобы сохранить изменения в его сборке.

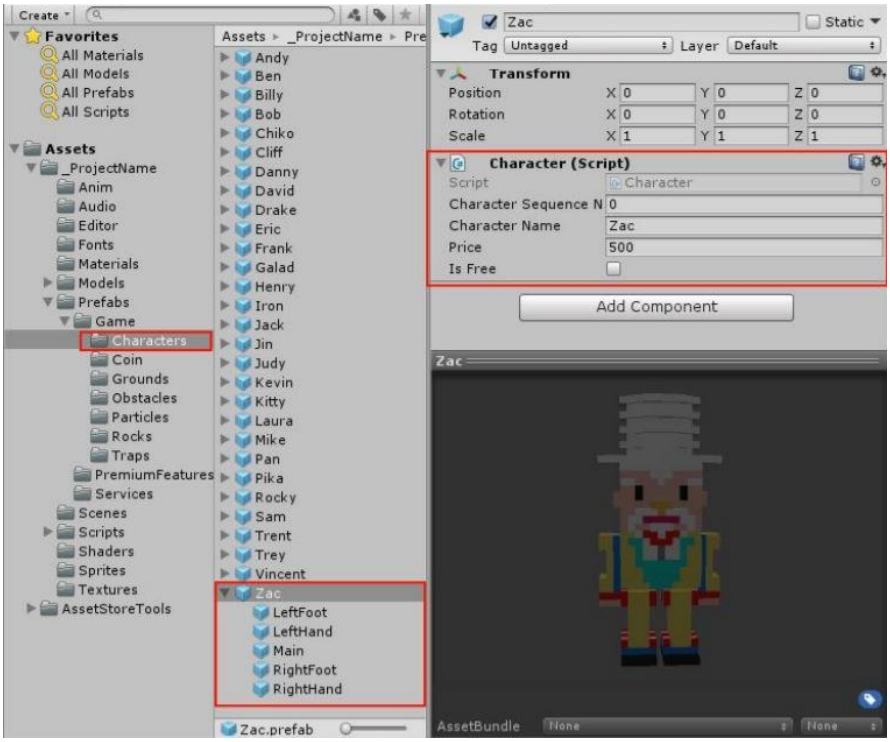


Рисунок 2.5 – Создание моделя персонажа

Теперь новый персонаж добавлен и готов к использованию в игре! Вы увидите его в списке в сцене выбора персонажей.

Фон градиентного неба в этой игре реализован вызовом градиентного шейдера VerticalGradientSkybox находится в папке шейдеров. Это простой в использовании шейдер вертекста/фрагмента. Чтобы изменить цвета градиента фона:

1 Перейдти к игровому объекту CameraBackground в разделе MainCamera.

2 Изменить цвета градиента: ColorTop, ColorBottom и отрегулировать GradientBias, пока вы не будете удовлетворены результатом.

3 Нажать Применить, чтобы сохранить изменения.

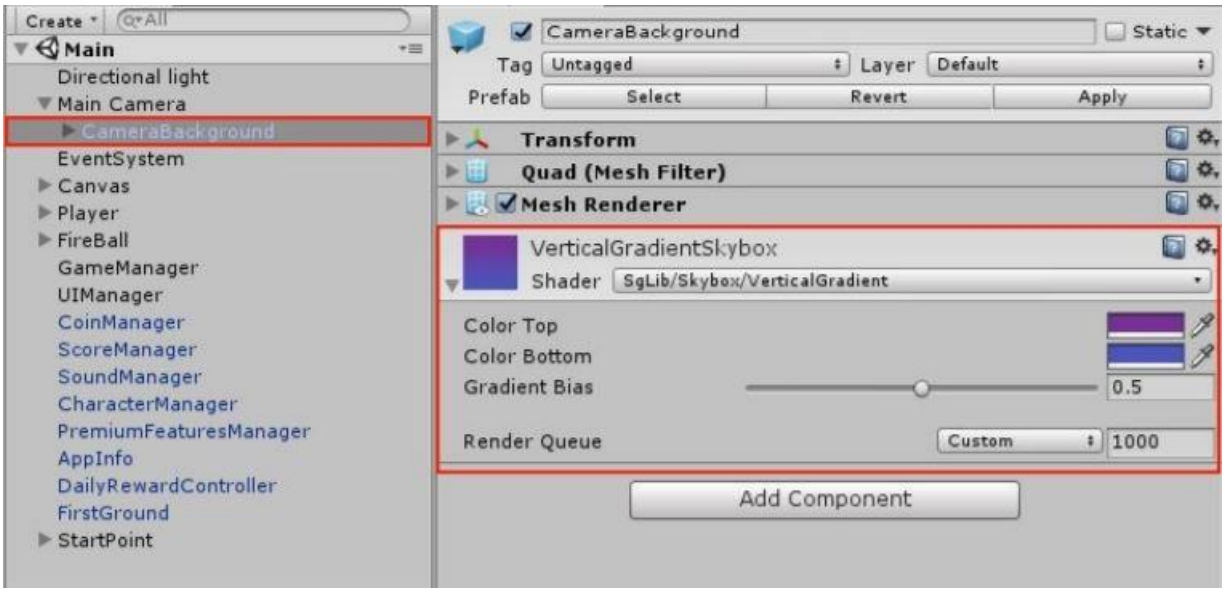


Рисунок 2.6 – Настройка фона игры

Все спрайты, используемые в этой игре (для кнопок и других компонентов пользовательского интерфейса), находятся в папке "Спрайты".

Рисунок 2.7 – Директория спрайтов

Все звуки, включенные в эту игру, бесплатны для использования в коммерческих проектах и находятся в папке Audio.

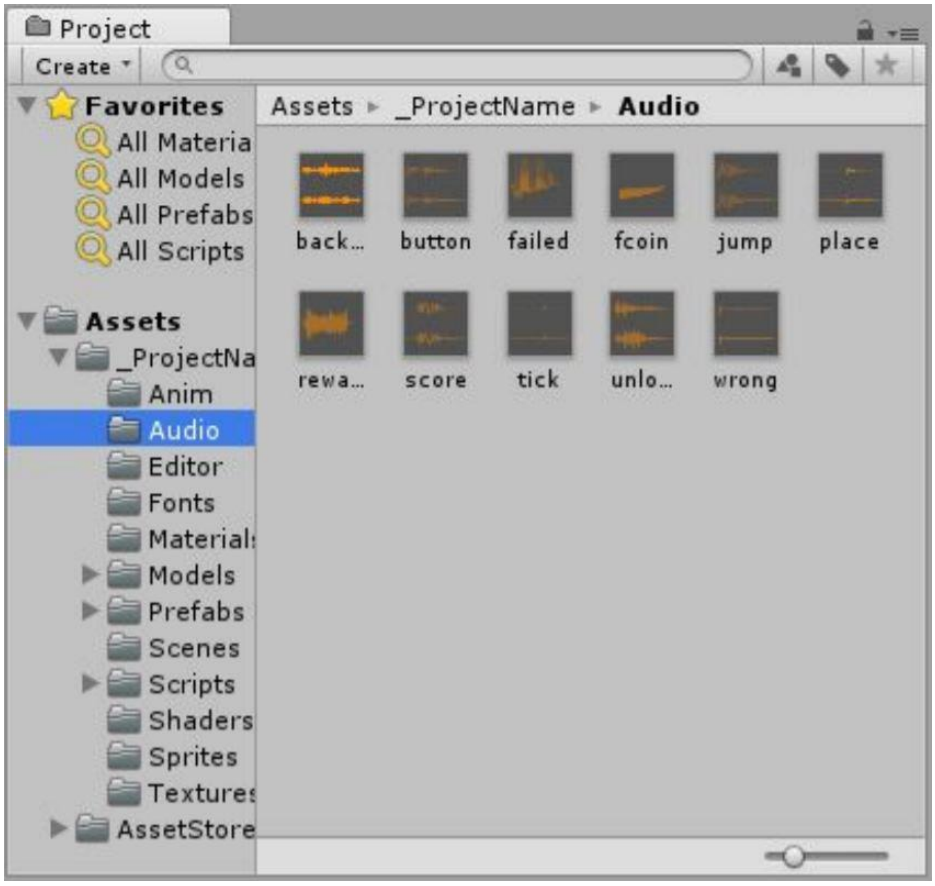


Рисунок 2.8 – Директория аудиоматериалов

В этой игре есть класс SoundManager для управления действиями в игре, такими как воспроизведение музыки или отключение/включение звуков.

2.5 Реализация кода в Visual Studio

Как уже говорилось выше язык программирования Си Шарп относится к семейству языков С и так как его синтаксис очень близок к С++ и Java для проекта был выбран именно этот язык.

Преимущества:

– Всем известно, что язык программирования С # является одним из популярных объектно-ориентированнах языков (все языковые сущности утверждают, что они объектно-ориентированы).

– компонентно-ориентированный подход к программированию, который способствует меньшей зависимости получаемого кода от архитектуры машины, гибкости, переносимости и легкому повторному использованию (фрагментов) программ);

– акцент на безопасности кода (по сравнению с С и С++);

– единая система набора текста;

– расширенная поддержка событийно-ориентированного программирования.

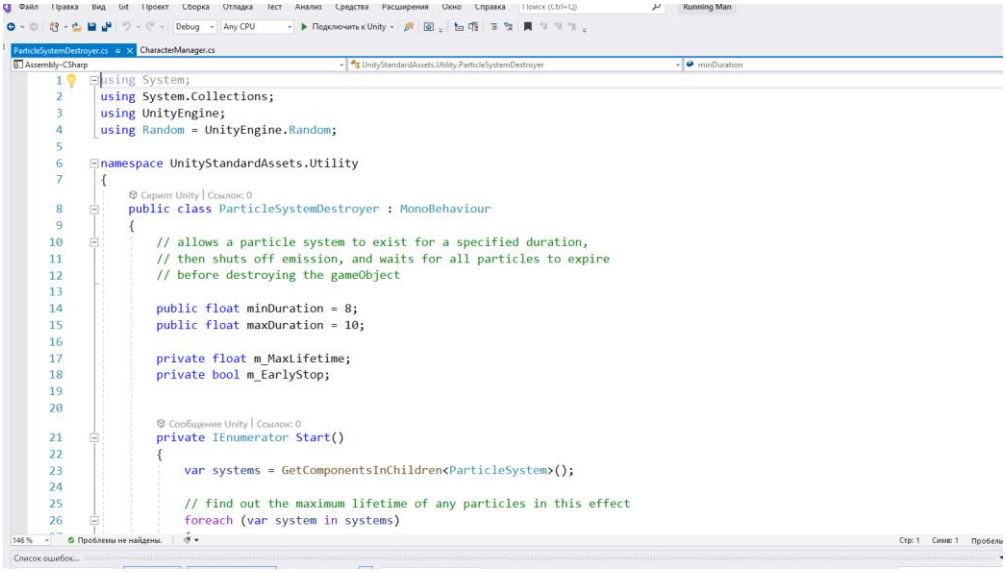


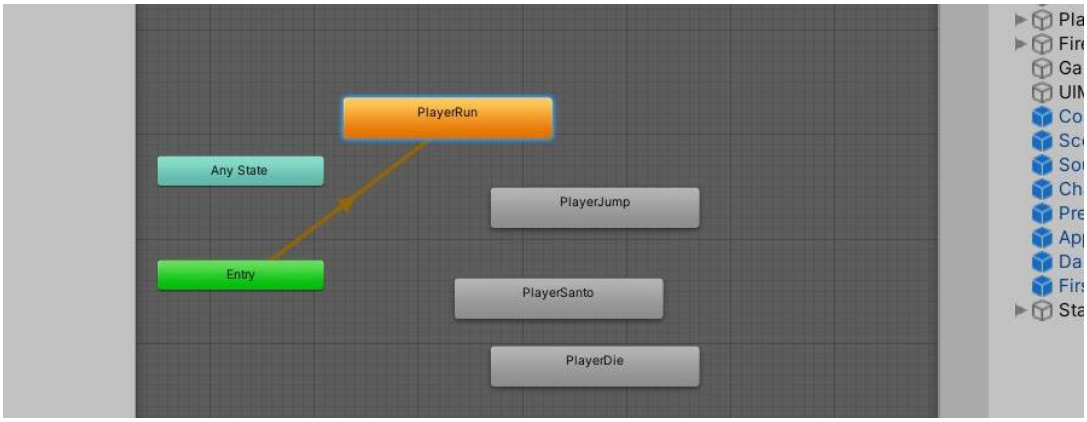
Рисунок 2.9 – Интерфейс Visual Studio

2.6 Модели в Unity

Файлы моделей, размещенные в папке Assets внутри проекта Unity, автоматически импортируются и сохраняются как ассеты Unity.

Файл модели может содержать 3D модель персонажа, здания или части мебели. Модель импортируется в виде набора ассетов. В окне Project(проект) главный импортированный объект представляется в виде Model Prefab(префаб модели). Обычно также существует несколько Mesh объектов, на которые ссылается Model Prefab.

Файл модели может также содержать данные анимации, которые можно использовать для анимации данной модели или других моделей. Данные анимации импортируется как один или несколько анимационных клипов.



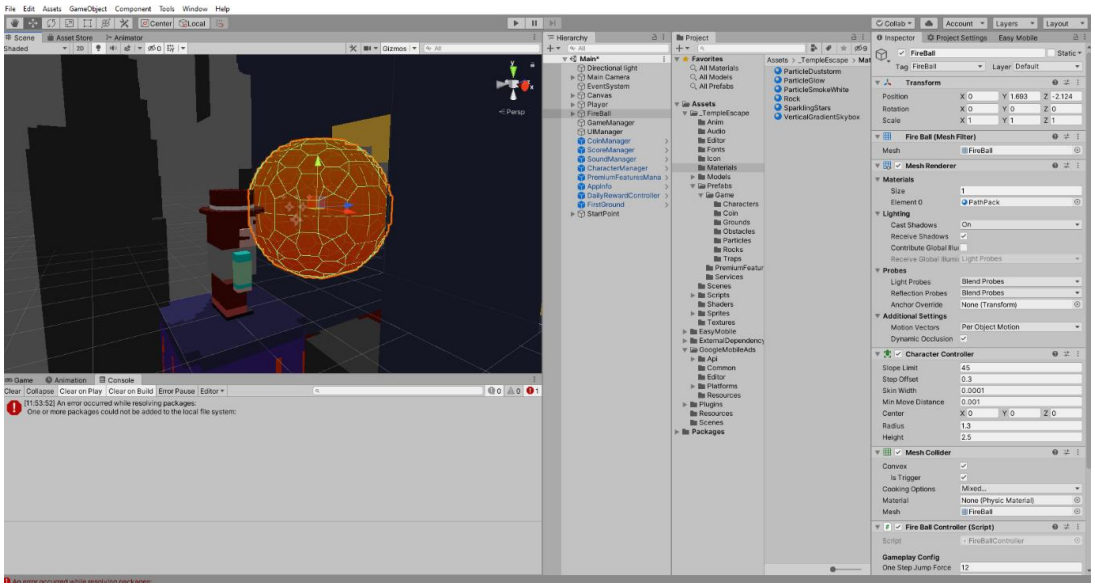


Рисунок 2.9.1 – Логическая диаграмма игры

Рисунок 2.9.2 – Добавление 3D модели

Риггинг — это создание арматуры, скелета модели. У арматуры есть несколько соединённых костей, к которым можно прикрепить вершины, чтобы они двигались при перемещении кости.

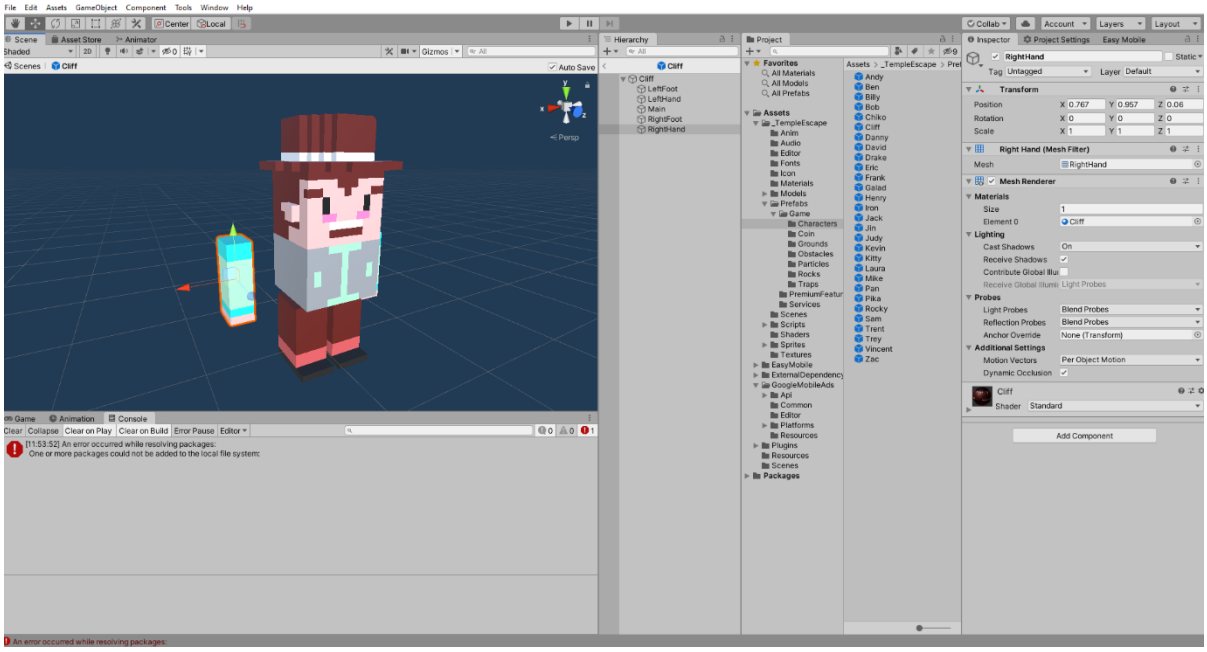


Рисунок 2.9.3 – Процесс обработки каркаса персонажа

3 Экспериментальная часть

1.1 Проверка работы игры

После запуска игры выходит главный экран с названием игры Running man (Бегущий человек), ниже с правой стороны бонусные очки, которые можно получать в течении определенного времени. Ниже слева кнопка для выбора игрока, их так же можно приобрести за монеты, которые даются в течении игры.

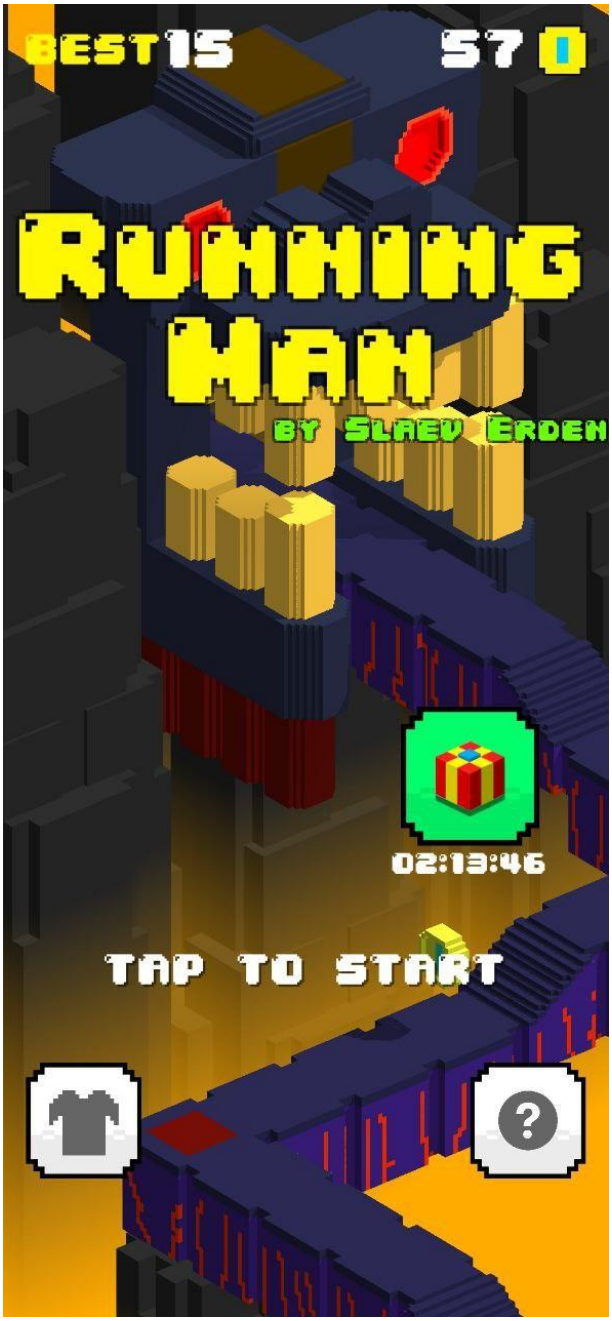


Рисунок 3.1 – Главный экран игры

В правом нижнем углу кнопка посказки, после нажатия появляется всплывающее окно с простой подсказкой

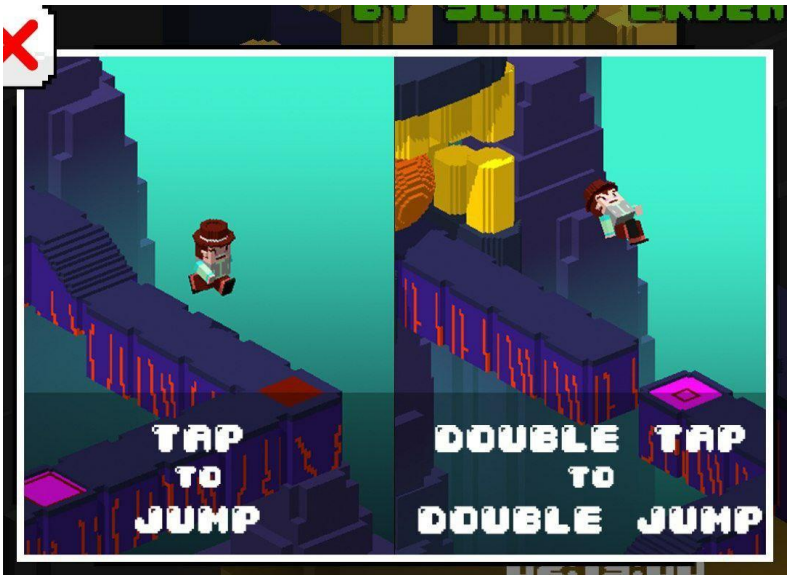


Рисунок 3.2 – Окно подсказки

Суть игры заключается в том, чтоб набрать как можно больше очков опыта, каждый рекорд записывается и отображается в левом верхнем углу.

Уровень не имеет конца, он генерируется автоматически засчет префаба, который выступает в роли шаблона для создания экземпляров хранимого объекта в сцене.

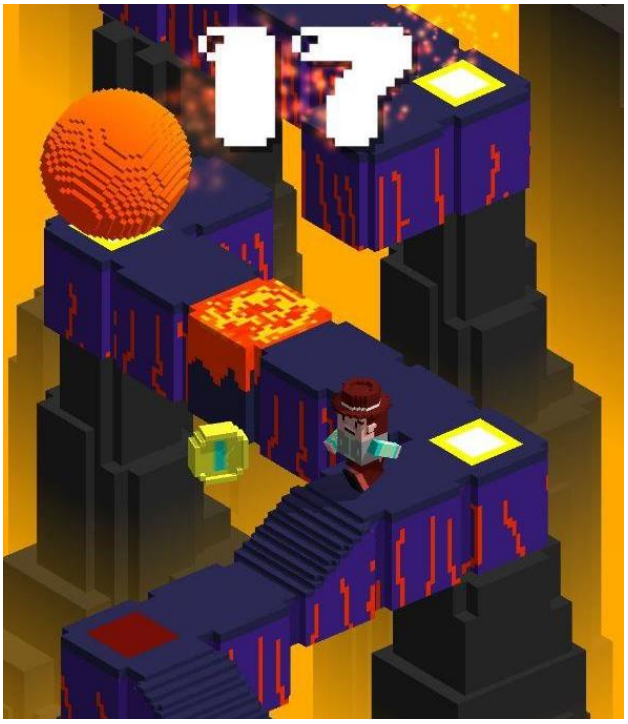


Рисунок 3.3 – Процесс игры

Как уже упоминалось выше, изменения в префабе автоматически применятся ко всем её экземплярам, однако можно изменить и отдельные экземпляры. Это полезно, например в случае, когда нкжно создать несколько похожих NPC, но с внешними различиями, чтобы добавить реалистичности. Чтобы было чётко видно, что свойство в экземпляре префаба изменено, оно показывается в инспекторе жирным шрифтом (если к экземпляру префаба добавлен совершенно новый компонент, то все его свойства будут написаны жирным шрифтом).

Рисунок 3.4 – Простой интерфейс окна настроек

Окно настроек состоит из двух функции:
1 Отключить звук.
2 Отключить фоновую музыку.

4 Экономическая часть дипломной работы

Темой дипломного проекта является «Разработка игровых приложений для мобильных устройств под ОС Android».

Постоянный прогресс высоких технологий и увеличение их возможностей привели к росту спроса на программы и приложения для этих самых технологий. Среди прочих направлений в этой области отдельное, особое место занимают программы на мобильные устройства. Среди них есть как необходимые для повседневной жизни, упрощающие быт и разные сферы жизнедеятельности, так и приложения – игры. Чем объясняется популярность игр. Это обусловлено тем, что кому-то хочется скоротать время, кто-то проникается азартом достижения цели и преодоления препятствии, кому-то приятно иметь возможность заниматься тем, что интересно для него в жизни, где и когда угодно (симулятор футбола), кто-то хочет развивать логику или реакцию и так далее. Все эти причины способствуют популярности мобильных игр и стимулированию их к разработке.

Экономическая часть дипломной работы служит для определения целесообразности создания продукта, в данном случае программного. Данный раздел состоит из анализа затрат на создание программного продукта, расчета цены самого программного продукта, эксплуатационных затрат, а также анализа и расчет результатов от создания и использования программного продукта.

4.1 Определение объема и трудоемкости разработки программного обеспечения

Расчет плановой сметы затрат на разработку программного обеспечения основывается на объеме программного продукта. Общий объем (Vo) программного продукта определяется количеством и объемом функций, выполняемых программой

Оценка объема программного продукта приводит к выбору наиболее подходящей единицы измерения для размера продукта. На практике широко распространены следующие единицы измерения:

- количество строк исходного кода (строки кода, LOC);
- функциональные точки (Fuxtionpaint, FR);
- точки свойств (Rrrrrupaint, PP);
- количество "пузырьков" в потоковом графике (диаграмма потока данных, DFD);
- количество сущностей в диаграмме сущностей (диаграмма отношений Eptu, ERD);
- процесс / управление-совместимые " квадраты "(CPP / CPP);
- количество различных элементов спецификации управления (элемент);
- количество документации (количество строк, количество строк);

– количество объектов, атрибутов и служб в графе объектов (объекты, атрибуты, службы).

Расчет объема программного продукта (количество строк исходного кода) включает определение типа программного обеспечения (Приложение А), всестороннюю техническую демонстрацию функций программного обеспечения и определение объема каждой функции. На этапе технико-экономического обоснования проекта имеются только приблизительные (прогнозные) оценки на основе фактических данных, имеющихся для аналогичных проектов, ранее завершенных или с использованием существующих стандартов (Приложение В). На основе информации о функциях разрабатываемого программного обеспечения количество функций и общий объем программного обеспечения определяются из каталога функций, который определяется (корректируется) с учетом условий разработки программного обеспечения организации.

где V_{yi} - уточненный объем с помощью отдельной функции (LOC).

Норматив и общая трудоемкость разработки программного обеспечения определяются в соответствии с количеством программного обеспечения на единицу объема и нормативами затрат труда. Проект состоит из 40 компонентов, и всего строк этих компонентов составляет 4257.

Стандартная сложность (t_n) разработки программного обеспечения определяется на основе количества, принятого для расчета (V_y), и категории сложности (Приложение В), определяемой с учетом сложности и новизны проекта и уровня разработки стандартных модулей.

Стандартная трудоемкость (T_N) используется в качестве основы для определения общей трудоемкости (t_o), которая рассчитывается по-разному в зависимости от размера проекта.

Разрабатываемое программное обеспечение имеет сложность 1 категории. Стандартная трудоемкость его разработки составит $T_n = 25$.

Общая трудоемкость малых проектов рассчитывается по формуле (4.1):

$$T_o = T_н \times K_c \times K_т \times K_н \qquad (4.1)$$

где, K_c – коэффициент, учитывающий сложность ПО;

$K_т$ – поправочный коэффициент, учитывающий степень использования при разработке стандартных модулей;

$K_н$ – коэффициент, учитывающий степень новизны ПО.

Коэффициент сложности рассчитывается по формуле (4.2):

$$K_c = 1 + \sum_{i=1}^{n} K_i , \qquad (4.2)$$

где, K_i – коэффициент, соответствующий степени повышения сложности ПО за счет конкретной характеристики;

n − количество учитываемых характеристик.

$$К_С = 1 + 0{,}06 + 0{,}07 + 0{,}12 = 1{,}25$$

Поправочный коэффициент, учитывающий степень использования при разработке ПО стандартных модулей ($К_т$), определяется удельным весом этих модулей в общем объеме проектируемого продукта $К_т = 0{,}7$.

Разрабатываемое ИС относится к Б-категории новизны ПО и является развитием определенного параметрического ряда ПО, не используется на основе новых типов ПК и в средах новых ОС. Значение коэффициента новизны соответствующее этим факторам равно $К_Н$=0,9.

$$Т_о = Т_н \times К_с \times К_т \times К_н = 25 * 1{,}25 * 0{,}7 * 0{,}9 = 19{,}68 \text{ норма/час}$$

$$Т_о = 25 * 1{,}25 * 0{,}7 * 0{,}9 = 19{,}68 \text{ норма/час}$$

4.2 Общая трудоемкость для крупных проектов.

Если стадия эскизного проекта в задании не предусмотрена, то удельный вес стадии технического проекта dтп равен сумме удельных весов стадий эскизного и технического проектов ($d_{тп} = d_{эп} + d_{тп}$). В том случае, когда объединяются стадии «Технический проект» и «Рабочий проект» в одну стадию «Технорабочий проект», трудоемкость «Технорабочего проекта» определяется по формуле (4.3):

$$Т_{трп} = 0{,}85 \times Т_{тп} + 1 \times Т_{рп}, \qquad (4.3)$$

где, $Т_{трп}$ − трудоемкость стадии «Технорабочий проект»;
$К_{тп}$ − трудоемкость стадии «Технический проект»;
$К_{рп}$ − трудоемкость стадии «Рабочий проект».

$$Т_{трп} = 0{,}85 * 2{,}53 + 1 + 16{,}3 = 19{,}45$$

Все стадии разработки ПО различаются трудоемкостью. Трудоемкость разработки стадий ПО ($Т_{уз}$, $Т_{уэ}$, $Т_{ут}$, $Т_{ур}$, $Т_{ув}$) определяется с учетом удельного веса трудоемкости стадии в общей трудоемкости ПО (d), сложности ($К_с$), новизны ПО ($К_н$) и степени использования стандартных модулей ($К_т$). При этом коэффициент $К_т$ используется только на стадии «Рабочий проект» при написании исходного кода (разработки программы).

Трудоемкость стадий ПО рассчитывается по следующим формулам:

Трудоемкость стадии ТЗ:

$$Т_{уз} = T_Н \times K_С \times d_з \times K_Н = 25 \cdot 1{,}25 \cdot 0{,}10 \cdot 0{,}9 = 2{,}81 \text{ чел/дн}$$

Трудоемкость стадии ЭП:

$$T_{уэ}=T_H \times K_С \times d_э \times K_Н = 25 \cdot 1{,}25 \cdot 0{,}08 \cdot 0{,}9 = 2{,}25 \text{ чел/дн}$$

Трудоемкость стадии ТП:

$$T_{ут}=T_H \times K_С \times d_р \times K_Н = 25 \cdot 1{,}25 \cdot 0{,}09 \cdot 0{,}9 = 2{,}53 \text{ чел/дн}$$

Трудоемкость стадии РП:

$$T_{ур}=T_H \times K_С \times d_з \times K_Н \times K_m = 25 \cdot 1{,}25 \cdot 0{,}58 \cdot 0{,}9 \cdot 0{,}7 = 16{,}3 \text{ чел/дн}$$

Трудоемкость стадии ВН:

$$T_{ув}=T_H \times K_С \times d_в \times K_Н = 25 \cdot 1{,}25 \cdot 0{,}15 \cdot 0{,}9 = 4{,}21 \text{ чел/дн}$$

Общая трудоемкость определяется как сумма трудоемкостей по стадиям (4.4):

$$T_у = T_{уз} + T_{уэ} + T_{ут} + T_{ур} + T_{ув}$$

$$T_у = 2{,}81 + 2{,}25 + 2{,}53 + 16{,}3 + 4{,}21 = 28{,}1 \text{ норма/час} \qquad (4.4)$$

Отсюда делаем вывод, что общая сумма трудоемкости составляет 28,1 норма-часов.

Таблица 1 – Трудоёмкость разработки программного продукта

Стадия жизненного цикла	Численность персонала
1. Техническое задание	2,81
2. Эскизный проект	2,25
3. Технический проект	2,53
4. Рабочий проект	16,3
5. Внедрение	4,21
Итого	28,1

4.3 Расчет полных затрат на разработку

Расчет полных затрат на разработку проектного решения в виде информационных технологий ($С_{пі}$) осуществляется по формуле (4.5):

$$С_{пі} = З_{фот} + З_{сзі} + М_і + Р_{сі} + Р_{мі} + Р_{нкі} + П_{зі} + Р_{ні}, \qquad (4.5)$$

где $З_{фот}$ – общий фонд оплаты труда разработчиков, тенге;

$З_{сзі}$ – отчисления по социальному налогу, тенге;

$М_і$ – затраты на материалы, тенге;

$Р_{сі}$ – затраты на специальные программные средства, необходимые для разработки проектного решения, тенге;

$Р_{мі}$ – затраты, связанные с эксплуатацией техники, тенге;

$Р_{нкі}$ – затраты на научные командировки, тенге;

$П_{зі}$ – прочие затраты, тенге;

$Р_{ні}$ – накладные расходы, тенге.

Размер фонда оплаты труда разработчиков ($З_{ФОТ}$) рассчитывается по формуле (4.6):

$$З_{ФОТ} = З_{оі} + З_{ді}, \quad (4.6)$$

где $З_{оі}$ – основная заработная плата, тенге;

$З_{ді.}$ – дополнительная заработная плата, тенге.

Базовая заработная плата рассчитывается исходя из общей трудоемкости, планируемой численности работников и условий, предусмотренных для разработки программного обеспечения. Работа оплачивается на основе единой тарифной ставки (ЕТС) Республики Казахстан, в которой указаны тарифные категории и тарифные коэффициенты. Имеется инструкция по распределению по тарифным категориям персонала во внебюджетном секторе экономики Республики Казахстан с учетом категории, должности, образования, сложности и практического опыта выполняемой работы.

Таблица 2 – Сведения по работникам, задействованным в проекте

Специалист - Исполнитель	Количество, человек	Заработная плата в месяц, тенге
Разработчик	1	150000
Итого		150000

Месячная тарифная ставка каждого исполнителя ($Т_м$) определяется путем умножения действующей месячной тарифной ставки 1-го разряда ($Т_{м1}$) на тарифный коэффициент ($Т_к$), соответствующий установленн ому тарифному разряду (47):

$$Т_м = Т_{м1} \times Т_к, \quad (47)$$

Почасовая ставка рассчитывается путем деления месячной ставки на расчетное среднемесячное рабочее время в 40 часов в неделю ($Ф_р$) (4.8):

$$Т_ч = Т_м / Ф_р, \quad (4.8)$$

где $Т_ч$ – часовая тарифная ставка (тыс.тенге);
$Т_м$ – месячная тарифная ставка (тыс.тенге).

$$Т_ч = 150000 / 8 * 21 = 892 \text{ тг/час}$$

Таблица 3 – Заработная плата

Наименование содержания работ	Исполнитель	Трудоёмкость норма-час	Заработная плата за час работы	Сумма заработной платы
Разработка ПО	Программист	171	892	152532
Итого				152532

$$З_{оі} = 152532 \text{ тенге}$$

Дополнительная заработная плата составляет в среднем 10% от основной заработной платы и рассчитывается по формуле (4.9):

$$Здi=Зоi \times Нд/100 \quad (4.9)$$

$$Здi = 152532 \cdot 0{,}1 \approx 15253 \text{ тенге}$$

где $Нд$ - коэффициент дополнительной заработной платы разработчиков (4.10).

$$З_{ФОТ} = З_{оі} + З_{ді} \quad (4.10)$$

$$З_{ФОТ} = 152532 + 15253 = 167785 \text{ тенге}$$

Социальный налог составляет 11% (ст. 358 п. 1 НК РК) от дохода работника, и рассчитывается по формуле (4.11):

$$З_{сзі} = (ФОТ\text{-}ПО) \times 11\%, \quad (4.11)$$

где ПО - пенсионные отчисления, которые составляют 10% от ФОТ и социальным налогом не облагаются:

$$ПО = ФОТ \times 10\% = 16778{,}5$$

Следовательно $З_{сзі}$ = (167765-16778,5)*11% ≈ 16610

Величина затрат на материалы на основании исходных данных определяется по формуле:

$$M_i = (З_{осн.} \times Н_{мз})/ 100\%,$$

где $Н_{мз}$ − норма расхода материалов от основной заработной платы (3-5%).

Величина затрат на материалы при норме расхода – 3%:

$$M_i = \frac{152532 \cdot 3}{100} = 4575 \text{ тенге}$$

Статья затрат "Специальное оборудование" (Пси) включает в себя затраты ресурсов на приобретение специальных дополнительных технических, программных средств, необходимых для разработки специального программного обеспечения, включая затраты на проектирование, изготовление, отладку, установку, эксплуатацию (4.12):

$$P_{ci} = \sum_{i=1}^{n} Ц_{ci}, \quad (4.12)$$

где $Ц_{ci}$ − стоимость конкретного специального оборудования (тыс.тенге);

n − количество применяемого специального оборудования.

Таблица 4 – Затраты на спецоборудование

Наименование материального ресурса	Количество израсходованного материала	Цена за единицу, тг	Сумма, тг
Ноутбук Lenovo ideapad 330	1	170000	170000
Мышь Компьютерная A-4TECH X7	1	11500	11500
ОС Windows 10 Pro (ключ активации)	1	22000	22000
Android Studio	1	0	0
Итого			203500

Расходы на программное обеспечение, указанные в разделе "прочие расходы" (P_{zi}), включают расходы на получение и подготовку конкретной научно-технической информации и литературы. Они определяются в соответствии со стандартом, разработанным во всей организации, в процентах

от базовой заработной платы (4.13):

$$П_{зі} = З_{оі} \cdot \frac{Н_{пз}}{100} \quad (4.13)$$

где $Н_{пз}$ - норматив прочих затрат в целом по организации в (%), в дипломной работе нужно брать 20%.

$$П_{з} = 152532 \cdot 0{,}2 = 30506 \text{ тенге}$$

Затраты, связанные с техническим обслуживанием оборудования управления, вспомогательного оборудования и экспериментального производства, включенные в статью "накладные расходы" ($Р_{нi}$), относятся на программное обеспечение, определенное в соответствии со стандартом, в процентах от основной оплаты труда ораторов. Стандарт устанавливается во всей организации (4.14):

$$Р_{нi} = З_{оі} \cdot \frac{Н_{нр}}{100}, \quad (4.14)$$

где $Р_{нi}$ - накладные расходы на конкретную ПО (тыс. тенге);

$Н_{нр}$ - норматив накладных расходов в целом по организации в (%), в дипломной работе нужно брать 70%.

$$Р_{нi} = 152532 \cdot 0{,}7 = 106772 \text{ тенге}$$

Таблица 5 – Затраты на разработку

Затраты на разработку	Условное обозначение	Значение, тенге	В процентах от общей суммы
Фонд оплаты труда	Зфот	167785	31.67%
Социальный налог	Зсзі	16610	3.14%
Материалы	Мі	4575	0.86%
Спецоборудование	Рсі	203500	38.41%
Прочие затраты	Пзі	30506	5.76%
Накладные расходы	Рні	106772	20.16%
Итого	$С_{пі}$	529748	100%

Рентабельность и прибыль созданного программного обеспечения ($П_{сі}$) определяются на основе результатов анализа рыночной конъюнктуры, переговоров с заказчиком (потребителем) и соглашения о цене реализации, включая налог на добавленную стоимость. Для разработки программного обеспечения, предназначенного для использования в организации, оценка

программного продукта должна проводиться в соответствии с действующими правилами и показателями внутреннего самофинансирования (по ценам, определенным для расчета услуг между классами). Прибыль рассчитывается по формуле (4.15):

$$П_{оі} = С_{пі} \times У_{рпі}/100, \quad (4.15)$$

где $П_{оі}$ – прибыль от реализации ПО заказчику (тыс.тенге);

$У_{рпі}$ – уровень рентабельности ПО (%), в дипломной работе брать 40-60%;

$С_{пі}$ – себестоимость ПО (тыс.тенге).

$$П_{оі} = 529748 * 0,4 = 211899 \text{ тенге}$$

Прогнозируемая цена ПО без налогов ($Ц_{пі}$) (4.16):

$$Ц_{пі} = С_{пі} + П_{сі} \quad (4.16)$$

$$Ц_{пі} = 529748 + 211899 = 741647$$

Прогнозируемая отпускная цена ($Ц_{оі}$) (4.17):

$$Ц_{оі} = Ц_{пі} + НДС\ . \quad (4.17)$$

Ставка налога на добавленную стоимость НДС в РК на 2013 год составляет 12% от отпускной цены ПО.

$$Ц_{01} = 741647 + 88997 = 830644$$

Организация-разработчик участвует в освоении ПО и несет соответствующие затраты, на которые составляется смета, оплачиваемая заказчиком по договору. Затраты на освоение определяются по нормативу ($Н_о$ = 10 %) от себестоимости ПО в расчете на 3 месяца и рассчитываются по формуле (4.18):

$$Р_{оі} = С_{рі} \times Н_о/100\% \quad (4.18)$$

$$Р_{оі} = 529748 * 0,1 = 52974 \text{ тенге}$$

Затраты на сопровождение ПО ($Р_{сі}$). Организация-разработчик осуществляет сопровождение ПО и несет соответствующие расходы, которые оплачиваются заказчиком в соответствии с договором и сметой на сопровождение. Затраты на сопровождение определяются по установленному нормативу ($Н_с$ = 20 %) от себестоимости ПО (в расчете на год) и

рассчитываются по формуле (4.19):

$$P_{ci} = C_{pi} \times H_c / 100\% \quad (4.19)$$

$$P_{ci} = 529748 * 0,2 = 105949 \text{ тенге}$$

4.4 Расчет цены программного продукта

Годовые эксплуатационные текущие затраты в условиях функционирования информационных технологий (С) рассчитываются по формуле (4.20):

$$С = ЗП + ОТ + А + М + НР, \quad (4.20)$$

где ЗП – годовые затраты на оплату труда специалистов при выполнении ими своих функций в рамках автоматизируемого процесса после внедрения ИТ, тенге;

ОТ – отчисления по социальному налогу, тенге;

А – затраты на амортизацию, тенге;

М – годовые материальные затраты на сопровождение программного продукта, тенге;

НР – накладные расходы, тенге.

$$С = 150000 + 16610 + 20000 + 48000 + 106772 = 341382$$

Таблица 6 – Текущие эксплуатационные затраты

Затраты на разработку	Условное обозначение	Значение, тенге	В процентах от общей суммы
Заработная плата	ЗП	150000	43.94%
Социальный налог	ОТ	16610	4.87%
Материалы	М	48000	14.06%
Накладные расходы	НР	106772	31.28%
Итого:		341382	100%

Текущие эксплуатационные затраты до внедрения информационных технологий рассчитываются аналогичным образом.

4.5 Расчет результатов от создания и использования ЭИС (или ожидаемой условно-годовой экономии)

Ожидаемая условно-годовая экономия определяется по формуле (4.21):

$$Э_{уг} = C_1 - C_2 + \sum Э_i, \quad (4.21)$$

где $Э_{уг}$ – величина экономии, тенге;

C_1 и C_2 – показатели текущих затрат по базовому и внедряемому вариантам, тенге.;

$\sum Э_i$ – ожидаемый дополнительный эффект от различных факторов, тенге.

$$Э_{уг} = 529748–341382 + (800000 * 0,5) = 588366$$

Дополнительный эффект рассчитывается в соответствии с факторами экономической эффективности.

При необходимости определения результатов от создания и использования ИС в динамике плановые показатели результативности приводятся к началу расчета путем умножения их на коэффициент дисконтирования.

4.6 Расчет основных показателей экономической эффективности

Показатели экономической эффективности рассчитываются в соответствии с принятой методикой расчета, однако следует учитывать следующие моменты:

– при расчете используется система общих показателей и частных показателей, отражающих отраслевые и функциональные характеристики проекта;

– для проектов с альтернативными решениями (бенчмарками) рассчитывается сравнительная эффективность. Вам нужно больше вариантов, включая один или несколько вариантов дизайна. Одним из них может быть существующая версия;

– в случае проектов, не имеющих аналогии, рассчитывается абсолютная эффективность, выраженная в экономии общей стоимости жизни и материализованного труда как в производстве, так и в эксплуатации. Если абсолютная эффективность будет отрицательной, проект будет исключен из дальнейшего рассмотрения.

Ожидаемый годовой экономический эффект от внедрения ИС определяется по следующей формуле (4.22):

$$Э_г = Э_{уг} - K \times E_н, \quad (4.22)$$

где $Э_г$ – ожидаемый годовой экономический эффект, тенге;

$Э_{уг}$ – ожидаемая условно-годовая экономия, тенге;

К – капитальные вложения, тенге;

$E_н$ – нормативный коэффициент экономической эффективности капитальных вложений.

Нормативный коэффициент экономической эффективности капитальных вложений определяется по формуле (4.23):

$$E_н = \frac{1}{T_н}, \quad (4.23)$$

где $T_н$ – нормативный срок окупаемости капитальных вложений, лет.

Нормативный срок окупаемости капитальных вложений, принимается исходя из срока морального старения технических средств и проектных решений ИС ($T_н$=1,2,3…n), для программных продуктов срок окупаемости принимаем равным 4 года.

$$E_н = 1 / 4 = 0,25$$

$$Э_г = 588366 – 529748 * 0,25 = 588366 – 132437 = 455437$$

Расчетный коэффициент экономической эффективности капитальных вложений составляет (4.24):

$$E_р = \frac{Э_{уг}}{K}, \quad (4.24)$$

где $E_р$ – расчетный коэффициент экономической эффективности капитальных вложений;

$Э_{уг}$ – ожидаемая условно-годовая экономия, тенге;

К – капитальные вложения на создание системы, тенге.

$$E_р = 588366 / 529748 = 1,1$$

Расчетный срок окупаемости капитальных вложений составляет (4.25):

$$T_р = \frac{1}{E_р}, \quad (4.25)$$

где $E_р$ – коэффициент экономической эффективности капитальных вложений.

$$T_р = 0,9$$

Таблица 7 – Показатели сравнительной экономической эффективности от внедрения программного продукта

Наименование показателей	Значение
Условная годовая экономия затрат, тенге	588366 тенге
Коэффициент экономической эффективности капитальных вложений ($E_р$)	1,1
Срок окупаемости капитальных вложений ($T_р$), лет	0,9 лет

Вывод

В этой части дипломной работы с помощью расчётов мы пришли к выводу, что условная годовая экономия затрат составляет 588366 тенге.

Реализация удобного и логически понятного интерфейса игры на платформе Android соответствует потребности пользователя в вопросах функций и дизайна. Срок окупаемости не так велик и менее чем за год можно выйти в плюс.

5 Безопасность жизнедеятельности

5.1 Анализ потенциально опасных и вредных факторов

Многие пользователи ПК страдают от психофизических факторов, таких как психическое переполнение, переполнение визуальных и слуховых анализаторов, монотонность работы и эмоциональная перегрузка. Воздействие этих неблагоприятных факторов приводит к снижению производительности, вызванному усталостью. Усталость связана с изменениями в центральной нервной системе и с тормозными процессами в коре головного мозга. Например, громкий шум вызывает трудности в распознавании цветовых сигналов, снижает цветочувствительность, остроту зрения, зрительную адаптацию, нарушает визуальную информацию, снижает производительность труда на 5-12%. Воздействие шума 90 дБ снижает производительность на 30-60%.

Медицинское обследование работников показало, что по мере снижения производительности труда высокий уровень шума приводит к потере слуха. Постоянное проживание человека в зоне комбинированного воздействия различных неблагоприятных факторов может привести к профессиональному заболеванию.

5.2 Воздействие вредных факторов на организм человека, их нормирование, способы и средства защиты.

Наиболее вредными и опасными производственными факторами на работе являются:

- химические;
- физические;
- биологические.
- психофизиологие;

Следует помнить, что разного рода физические факторы могут негативно воздействовать на здоровье программиста:

- высокая и низкая температура воздуха;
- избыточное загрязнение воздуха пылью и газом;
- высокая и низкая влажность;
- недостаточная освещенность РМ;
- шум, превышающий допустимые пределы;
- повышенный уровень ионизирующего излучения;
- повышенный уровень электромагнитных полей;
- повышенный уровень статического электричества;
- риск поражения электрическим током;
- монитор низкого качества.

Химические опасности включают:

– появление активных частиц из-за ионизации воздуха во время работы компьютера.

К психологически вредным факторам, влияющим на работу оператора, относятся:

– нервные и эмоциональные перегрузки;
– психическое напряжение;
– перенапряжение визуального анализатора.

Оператор ПК более чувствителен к следующим видам заболеваний: снижение остроты зрения, боль и боль в глазу, усталость в глазах, физические расстройства (сонливость или бессонница, головная боль, головокружение, онемение конечностей и т.д.).); более частые психические расстройства, нарушения сна, различные неврологические заболевания, грипп, острые респираторные инфекции, острые инфекции дыхательных путей, бронхит, бронхиальная астма, невроз, остеохондроз, то есть неправильный орган зрения, боли в сердце, одышка, сухость кожи, слизистых оболочек, особенно носа, горло.

Чтобы уменьшить неблагоприятные последствия и опасные факторы, которым подвергается человек, работающий на компьютере, он должен соблюдать стандарт.

Основными нормализованными параметрами являются:

– визуальные параметры мониторов;
– освещение рабочего места;
– значения параметров электромагнитного излучения;
– оптимальные значения параметров микроклимата;
– уровень ионизации воздуха;
– уровень звука и звуковое давление в октавном диапазоне частот;
– стандарты вибрации.

5.3 Освещенность и воздухообмен рабочего места программиста.

В промышленных системах освещения применяются следующие основные требования: соответствие уровня освещенности на рабочем месте сотрудника, выполняемый визуальную работу, относительно равномерное распределение яркости рабочих поверхностей, а также окружающего пространства, отсутствие резких теней, прямой и отраженной яркости, постоянное освещение рабочей зоны; оптимально организованный световой поток, излучаемый осветительным оборудованием, эффективность, электробезопасность, противопожарная защита.

Таблица 5.1 – Коэффицента запаса

Помещения	Примеры помещений	Газоразрядные лампы	Лампы накаливания
Засыпленность свыше	Цементные заводы, литейные цеха и т.п.	2	1,7
Дым, копоть 1-5	Кузнечные, сварочные цеха и т.п.	1,8	1,5
Менее 1мг/м^2	Инструментальные, сборочные цеха	1,5	1,3
Значительная концентрация паров кислот и щелочей	Цеха химических заводов, гальванические цеха	1,8	1,5
Запыленность значительно менее 1 мг/м^3, отсутствие паров кислот и щелочей	Жилые, административные и офисные и т.п. помещения	1,2	1,1

В соответствии с требованиями к освещению стандарты перечислены ниже.

– освещение на столе должно быть 300-500;

– для освещения документов могут быть установлены лампы. Местное освещение не должно создавать отражений на поверхности экрана и может увеличить освещенность экрана более чем на 300 люкс;

– сила прямого света источников света должна быть ограничена, а освещенность на поверхности объектов в поле зрения не должна превышать 200 кд / кв. м.;

– ограничить отраженную яркость на рабочих поверхностях (экран, стол, клавиатура и др.) правильно выбрав тип ламп и расположение рабочих мест по отношению к естественным и искусственным источникам света, при этом экраны ВДТ и ПК не должны превышать 40 кд/кв. м.;

– ограничить неравномерное распределение яркости в поле зрения пользователя ВДТ и ПК, при этом соотношение яркости между рабочими поверхностями не должно превышать от 3:1 до 5:1 и 10:1 между рабочими поверхностями и поверхностями стен и оборудования;

– в качестве источника света для искусственного освещения должны использоваться люминесцентные лампы типа ЛБ. Металлогалогенные лампы мощностью до 250 Вт могут использоваться для установки отраженного освещения в промышленных, административных и общественных помещениях.

Лампы накаливания могут использоваться в местных органах освещения.

Еще наиболее важным фактором здоровой атмосферы и хорошего самочувствия программистов является вентиляция.

Вентиляция – это система мер и устройств, предназначенных для обеспечения соответствия метеорологических условий и чистоты воздуха гигиеническим требованиям в рабочей и эксплуатационной зонах помещения.

Вентиляция получается путем перемещения воздуха: он загрязнен из помещения, свежий в помещение.

В зависимости от способа движения воздуха вентиляция делится на: естественную или искусственную (механическую). В случае смешанной вентиляции естественная и механическая вентиляция комбинируются в разных вариантах.

Искусственная (механическая) вентиляция-вентиляция организуется с помощью электрических вентиляторов. Механическая вентиляция устроена в присутствии газов и паров: если выхлопные шахты не могут быть установлены, она присутствует внутри большого количества переборок, которые препятствуют потоку воздуха (если невозможно организовать аэрацию-из-за разницы внешнего и внутреннего гравитационного давления) при вентилировании токсичных взрывчатых веществ. Естественная вентиляция-характеризуется движением воздуха за счет естественных процессов, не требующих какого-либо механизма. Я различаю организованную и неорганизованную естественную вентиляцию.

Можно разделить вентиляцию по назначениям:

– вытяжную;

– приточную;

– приточно-вытяжную.

По организации воздухообмена вентиляция подразделяется на:

– вентиляция, обеспечивающая полную замену воздуха в помещении;

– вентиляция, которая обеспечивает частичную замену воздуха(локальную).

Система вентиляции в помещении, где работают программисты, должна быть оборудована воздушной заслонкой. Это требование часто не воспринимается всерьез, хотя оно вполне объяснимо. Зимой воздух очень сухой, влажность в офисах достигает 15-20%. Но отношение к увлажнителям воздуха всегда было отрицательным — это дорогое, довольно капризное средство. Современные системы вентиляции не имеют этой проблемы. Однако, возможно, из-за инерции, очень немногие здания имеют центральную систему увлажнения. Здание должно быть оборудовано кондиционером, который должен поддерживать среднюю температуру от 22 до 23 °C.

Летом все просто – даже традиционный бытовой кондиционер обеспечивает температуру 22 °C. Дело в том, что зимой, даже в сильные морозы, офисы жарко освещаются, большое количество оргтехники, персонала, посетителей выделяют достаточно тепла, чтобы отапливать помещения до 25-28 °C.

Есть два выхода:

1 Использовать сильную вентиляцию, чтобы обеспечить в помещении достаточным количеством воздуха при температуре 17-19 °C, чтобы "разбавить" тепло (в густонаселенном офисе разнообразие зимнего воздухообмена может достигать 9-10, и очевидно, что мало кто пойдет на это, потому что огромные воздуховоды, множество воздуховодов, огромные системы вентиляции-дорогие и неудобные;

2 используйте системы кондиционирования воздуха, которые можно использовать для охлаждения зимой. К ним относятся охладители с гликолевым контуром, системы тепловых насосов (внутренние минихиллеры с водяным охлаждением) и градирни с гликолевым контуром, специальные системы VRF. Чтобы выполнить еще одно требование классификации, арендаторы должны иметь возможность контролировать температуру воздуха в каждом офисном блоке с помощью локальных замков (с помощью устройств, которые доводят температуру до заданного значения). Это могут быть внутренние блоки VRF или фанкойлы. Однако все офисные комплексы сейчас сдаются в аренду на этапе shell & core, и цена серьезно падает.

5.4 Расчет общего освещения РМ программиста

Для офисного помещения размерами А = 4 м, В = 6 м произвести расчет общего освещения методом удельной мощности. Для освещения использовать люминесцентные лампы, установленные в двухламповые светильники ПВЛМ (пылеводозащитные). Высота помещения 4 м. Коэффициенты отражения потолка $\rho_{пт} = 50$ %, стен $\rho_{с} = 50$ %, рабочей поверхности $\rho_{р} = 8$ %.

1) Для помещения офисного типа нормативной освещенностью для газоразрядных ламп является $E_{п} = 200$лк, высота рабочей поверхности $h_{п} = 0.8$ м. Примем свес $h_{с} = 0.5$ и определим высоту подвеса светильника (5.1):

$$h_{р} = 4 – 0.7 – 1 = 2.3 \text{ м} \quad (5.1)$$

По таблице 5.1 коэффицент запаса $К_{Зн} = 1.2$.

Примем для использования лампы ЛД-80.

2) Определим число рядов светильников:

Светильники ПВЛМ имеют кривую силы света типа Д и для них $\lambda=1.4$

Оптимальное расстояние между рядами светильников (5.2):

$$L_{оптг} = \lambda \times h = 1.4 \times 2.3 = 3.22 \text{ м}. \quad (5.2)$$

Поскольку ширина помещения 4 м, то исходя из $L_{оптг} = 3.22$ м

Примем ориентированную схему размещения светильников в 2 ряда параллельно длинной стороне В.

3) Определим удельную мощность.

При $h_р$ = 2.3 м и площади помещения S = A × B = 4 × 6 = 24 м^2, для светильника ПВЛМ с лампами ЛД-80 находим $\omega_т$ = 9.1 Вт/м^2, для $Е_т$ = 100 лк и $К_{зт}$ = 1.2.

Поскольку нам необходимо обеспечить освещенность $Е_и$ = 300 лк при $К_з$ = 1.2, проведем перерасчет (5.3):

$$\omega = \frac{Е \times К}{Е \times К} \times \omega = \frac{300 \times 1.2}{100 \times 1.2} \times 9.1 = 27.3 \text{ Вт/м}^2 \quad (5.3)$$

4) Определим мощность осветительной установки (5.4):

$$P = \omega \times S = 27.3 \times 24 = 655.2 \text{ Вт} \quad (5.4)$$

5) Определим необходимое число светильников (5.5):

$$n = \frac{P}{n \times P} = \frac{655.2}{2 \times 80} = 4.095 \approx 4 \quad (5.5)$$

6) Исходя из размеров помещения 4 × 6 м, окончательно примем схему размещения светильников в 2 ряда по 2 светильника в ряд параллельно длинной стороне В. Расстояние между центрами светильников:

$$L_л = 6 / 2 = 3 \text{ м}$$

$$L_п = 4 / 2 = 2 \text{ м}$$

5.6 Расчет воздухообмена в помещении

Для подбора кондиционера, соответствующего условиям труда, необходимо рассчитать требуемый для комфортных условий воздухообмен. Проведём расчёт воздухообмена по формуле (5.6):

$$V = 3.6 \times Q_{изб} / C_в \times \rho_в \times (t_{уд} - t_{пр}) \text{ м}^3/\text{час} \quad (5.6)$$

$Q_{изб}$ - избыточное тепло, которое необходимо удалить из помещения.

$C_в$ - удельная массовая теплоёмкость воздуха ($C_в$ = 1,042 кДж).

$\rho_в$ - плотность воздуха в помещении (В = 1,2 кг/м^3).

$t_{уд}$ - температуры удаляемого и приточного воздуха (22°С).

$t_{пр}$ - Температура приточного воздуха $t_{пр}$ при наличии избытков тепла должна быть на 5 °С ниже температуры воздуха в рабочей зоне и поэтому

$$t_{пр} = 21 - 5 = 16°C.$$

Избыточное тепло, которое необходимо уменьшить на рабочем месте,

определяется по следующей формуле (5.7):

$$Q_{изб} = Q_л + Q_{об} + Q_{ср} \quad (5.7)$$

$Q_л$ – тепло от людей и определяется формулой:

$Q_л = q \times n$ (где q тепло, выделяемое одним человеком при работе в определённой температуре, q = 45 Вт при лёгкой работе и n количество людей, находящихся в помещении)

$$Q_л = 8 \times 45 = 360 \text{ Вт}$$

$Q_{об}$ – тепло от оборудования $Q_{ср}$ – тепло от солнечной радиации через световые проёмы и определяется по формуле:

$Q_{об} = 0.2 \times N$ (где N - суммарная мощность электрических приборов, мощность одного системного блока и монитора 53 Вт, в помещении 8 компьютеров).

$$Q_{об} = 0.2 \times (8 \times 53) = 84.8 \text{ Вт}$$

$Q_{ср}$ - поступление тепла от солнечной радиации через световые проёмы определяем по формуле (5.8):

$$Q_{ср} = F_{ост} \times Q_{ост} \times A_{ост} \times k \quad (5.8)$$

$F_{ост}$ - площадь световых проёмов, определяемая по формуле (5.9):

$$F_{ост} = h_o \times l_o \times N_o \text{ м}^2 \quad (5.9)$$

h_o - высота окна.

l_o - ширина окна.

N_o - количество окон.

$$F_{ост} = 2 \times 1.5 \times 2 = 6\text{м}^2$$

$Q_{ост}$ - количество тепла, поступающего с излучением через 1 м2 поверхности остекления

$$Q_{ост} = 45 \text{ Вт/м}^2$$

$A_{ост}$ - коэффициент, зависящий от вида остекления

$$Аост = 1.15$$

k - коэффициент загрязнённости окон

$$k = 0.5$$

$$Q_{ср} = 6 \times 45 \times 1.15 \times 0.5 = 155 \text{ Вт}$$

И используя это значения вычислим $Q_{изб}$:

$$Q_{изб} = 360 + 84.8 + 155 = 599.8 \approx 600 \text{ Вт}$$

Теперь определим требуемый для комфортных условий воздухообмен:

$$V = 3.6 \times 600 / 1.042 \times 1.2 \times (22 – 16) = 2160 / 7.5 = 288 \text{ м}^3/\text{час}$$

Вывод

Проведя расчеты воздухообмена на рабочем месте, можно сделать вывод, что для того, чтобы нормализовать параметры микроклимата, на рабочем месте оператора ПК требуется выбрать кондиционер, мощность которого хватит для воздухообмена в нудном помещении, а также поддерживает оптимальные, приемлемые параметры температуры, влажности, скорости воздуха. Наиболее удобными являются кондиционеры со сплит-системой. Кондиционер не так сильно шумит и не мешает рабочему процессу, поскольку внутренняя часть кондиционера не имеет источника шума и действует как устройство управления системой.

В качестве примера можно выбрать любой современный кондиционер с подачей воздуха не менее 288 м3 / ч с наименьшим шумоотдачей и контролем (зима-лето).

Заключение

Создание, разработка и реализация мобильных игр на операционной системе Android – увлекательная и одновременно сложная задача, реализация которой может быть универсальной и технологически инновационной. Это требует логического мышления и структурного подхода. При выполнении этого проекта я столкнулся с рядом задач и приобрел новые знания, выполняя требования при использовании новых технологий, поближе ознакомился с языком программирования C#, игровыми движками, средствами анимации и бесплатными площадками 3D моделей. Следует отметить, что результат выполненной работы и рабочая версия игры соответствует современным требованиям мобильных игр.

При выполнении этого дипломного проекта я изучил принципы, технологии, требования и задачи разработки мобильных игр. Я ознакомился с принципами разработки и создания приложений для операционной системы Android, учёл весь жизненный цикл разработки ПО, тщательно изучил современные межплатформенные технологии и методологию разработки мобильных приложений, анализируя игровые движки и изучил основы реализации кода в проекте.

Кроме того, было установлено, что рассматриваемый предмет является наиболее важным в реалиях современного рынка мобильных устройств. Игры в настоящее время находятся на пике своей популярности, и они приносят очень большую прибыль как разработчикам, так и спонсорам, в зависимости, от качества конечного продукта.

Созданная мною игра основана на всех основных шаблонах жанра Раннера и включает в себя все функции и функции игрока в этом жанре. Это приложение соответствует всем стандартам и требованиям, не нарушая права площадок мобильных игр.

В четвертой части дипломного проекта была рассчитана общая стоимость продукта на основе сводки различных видов затрат, которые могут быть понесены в процессе производства. Этот расчет соответствует реализации любого типа продукта, как для развития клиентов, так и для самостоятельной разработки, для последующей публикации в интернет-магазине приложений (Play Market). Исследования аналогичных продуктов на рынке показали, что стоимость разработки полностью соответствует качеству разрабатываемого мобильного приложения, что позволяет окупить себя и приносить прибыл уже в ближайшее время.

В пятой части дипломного проекта, а именно в разделе «Безопасность жизнедеятельности» были рассчитаны условия организации всех согласованных стандартных рабочих мест для сотрудников, участвующих в реализации проекта, и влияние вредных факторов на программистов.

Список использованной литературы

1 Сайт http://startandroid.ru/ru/

2 Сайт http://developer.android.com

3 Сайт https://habr.com/ru/company/plarium/blog/447820/

4 Ретабоуил Сильвен. Android SDK. Разработка приложений под Android на C/C++; ДМК Пресс -Москва, 2012. -496 с.5

5 https://www.labirint.ru/books/533397/

6 Хокинг Джозеф. Unity в действии. Мультиплатформенная разработка на C#. 2-е межд. Издание, Питер, 2019. -352 с

7 https://books.google.kz/books/about/Unity

8 Скит Джон. C# для профессионалов. Тонкости программирования 2019 Издание Вильямс

9 https://vk.com/wall-51126445_5977

10 Д. Гриффито Head First. Программирование для Android 2016, 704 с.

11 https://ru.pdfdrive.com/head-first.html

12 Latypov V. Android NDK Game Development Cookbook 2013, 320 p

13 https://www.amazon.com/Android-NDK-Game-Development-Cookbook/dp/1782167781

14 David M.B Physics for Game Developers 2015, 336 p

15 https://www.amazon.com/gp/product/0596000065

16 Коробко, В.И. Охрана труда: Учебное пособие для студентов вузов / В.И. Коробко. – М.: ЮНИТИ-ДАНА, 2013. – 239 с.

17 Майер Рето Android 4. Программирование приложений для планшетных компьютеров и смартфонов. Эксмо, 2013–816 с.

18 Джереми Гибсона Бонд. Unity и C#. Геймдев от идеи до реализации. 2-е изд, Издательский дом Питер. 2019–755 с

19 Джесси Шелл. Геймдизайн. Как создать игру, в которую будут играть все. 2015–389 с.

20 Джозеф Хокинг. Unity в действии. Мультиплатформенная разработка на C#. 2013–430 с.

Приложенте Б

Листинг программы

```
package com.example.jwtauthreg.service;

import com.example.jwtauthreg.Entity.RoleEntity;
import com.example.jwtauthreg.Entity.TaskEntity;
import com.example.jwtauthreg.Entity.UserEntity;
import com.example.jwtauthreg.repository.RoleEntityRepository;
import com.example.jwtauthreg.repository.TaskEntityRepository;
import com.example.jwtauthreg.repository.UserEntityRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Service;

import java.time.LocalDate;
import java.util.List;

@Service
public class UserService {

    @Autowired
    private TaskEntityRepository taskEntityRepository;

    @Autowired
    private UserEntityRepository userEntityRepository;

    @Autowired
    private RoleEntityRepository roleEntityRepository;

    @Autowired
    private PasswordEncoder passwordEncoder;

    public UserEntity saveUser(UserEntity userEntity) {
        RoleEntity userRole = roleEntityRepository.findByName("ROLE_USER");
        userEntity.setRoleEntity(userRole);
        userEntity.setPassword(passwordEncoder.encode(userEntity.getPassword()));
        return userEntityRepository.save(userEntity);
    }

    public UserEntity findByLogin(String login) {
        return userEntityRepository.findByLogin((login));
    }
```

```
    public UserEntity findByLoginAndPassword(String login, String password) {
        UserEntity userEntity = findByLogin(login);
        if (userEntity != null) {
            if (passwordEncoder.matches(password, userEntity.getPassword())) {
                return userEntity;
            }
        }
        return null;
    }

    public TaskEntity post(TaskEntity taskEntity){
        return taskEntityRepository.save(taskEntity);
    }

    public String del(TaskEntity taskEntity){
        taskEntityRepository.deleteById(taskEntity.getId());
        return "Задача успешно удалена :)";
    }

    public TaskEntity getTaskById(int id) {
        return taskEntityRepository.findById(id).orElse(null);
    }

    public List<TaskEntity> getTaskByDate(LocalDate date) {
        return taskEntityRepository.findAllByDate(date);
    }

    public List<TaskEntity> getTasks(){
        return taskEntityRepository.findAll();
    }

    public TaskEntity put(TaskEntity taskEntity){
        TaskEntity task =
taskEntityRepository.findById(taskEntity.getId()).orElse(null);
        task.setProcess(taskEntity.getProcess());
        return taskEntityRepository.save(task);
    }
}

package com.example.jwtauthreg.config;

import com.example.jwtauthreg.Entity.UserEntity;
import org.springframework.security.core.GrantedAuthority;
```

```
import org.springframework.security.core.authority.SimpleGrantedAuthority;
import org.springframework.security.core.userdetails.UserDetails;

import java.util.Collection;
import java.util.Collections;

public class CustomUserDetails implements UserDetails {

    private String login;
    private String password;
    private Collection<? extends GrantedAuthority> grantedAuthorities;

    public static CustomUserDetails
fromUserEntityToCustomUserDetails(UserEntity userEntity){
        CustomUserDetails c = new CustomUserDetails();
        c.login = userEntity.getLogin();
        c.password = userEntity.getPassword();
        c.grantedAuthorities = Collections.singletonList(new
SimpleGrantedAuthority(userEntity.getRoleEntity().getName()));
        return c;
    }

    @Override
    public Collection<? extends GrantedAuthority> getAuthorities() {
        return grantedAuthorities;
    }

    @Override
    public String getPassword() {
        return password;
    }

    @Override
    public String getUsername() {
        return login;
    }

    @Override
    public boolean isAccountNonExpired() {
        return true;
    }

    @Override
    public boolean isAccountNonLocked() {
```

```
        return true;
    }

    @Override
    public boolean isCredentialsNonExpired() {
        return true;
    }

    @Override
    public boolean isEnabled() {
        return true;
    }
}

package com.example.jwtauthreg.config;

import com.example.jwtauthreg.config.jwt.JwtFilter;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import
org.springframework.security.config.annotation.web.configuration.EnableWebSecu
rity;
import
org.springframework.security.config.annotation.web.configuration.WebSecurityCo
nfigurerAdapter;
import org.springframework.security.config.http.SessionCreationPolicy;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;
import
org.springframework.security.web.authentication.UsernamePasswordAuthentication
Filter;

@Configuration
@EnableWebSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter {

    @Autowired
    private JwtFilter jwtFilter;

    @Override
    protected void configure(HttpSecurity http) throws Exception {
        http.
```

```
            httpBasic().disable()
            .csrf().disable()

.sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS)
.and()
            .authorizeRequests()
            .antMatchers("/admin/*").hasRole("ADMIN")
            .antMatchers("/user/*").hasAnyRole("USER", "ADMIN")
            .antMatchers("/register/", "/auth/").permitAll()
            .and()
            .addFilterBefore(jwtFilter, UsernamePasswordAuthenticationFilter.class);
    }

    @Bean
    public PasswordEncoder passwordEncoder() {
        return new BCryptPasswordEncoder();
    }
}
```