Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ЛЕСОСИБИРСКИЙ ПЕДАГОГИЧЕСКИЙ ИНСТИТУТ –
филиал Сибирского федерального университета

Высшей математики, информатики, экономики и естествознания
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
Храмова Л.Н. Храмова
подпись инициалы, фамилия
«14» 06 2024 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 Информационные системы и технологии
код-наименование направления

ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНТЕРАКТИВНОЙ ИГРЫ «МАТЕМАТИЧЕСКИЙ КВЕСТ» ДЛЯ ПРОБЛЕМНОГО ОБУЧЕНИЯ

Руководитель 14.06.24 доцент, канд. пед. наук С.С. Ахтамова
подпись, дата должность, ученая степень инициалы, фамилия

Выпускник 14.06.24 Д.А. Кетов
подпись, дата инициалы, фамилия

Нормоконтролер 14.06.24 А.В. Фирер
подпись, дата инициалы, фамилия

Лесосибирск 2024

РЕФЕРАТ

Выпускная квалификационная работа по теме «Проектирование и разработка интерактивной игры «Математический квест» для проблемного обучения» содержит 64 страницы текстового документа, 42 использованных источника, 2 таблицы, 22 рисунка, 5 приложений.

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ, ИНТЕРАКТИВНАЯ ИГРА, ПРОБЛЕМНОЕ ОБУЧЕНИЕ, МАТЕМАТИЧЕСКИЙ КВЕСТ.

Актуальность данной темы обусловлена тем, что в научной и технической литературе недостаточно освещены вопросы проектирования и разработки интерактивных образовательных игр, таких как математические квесты.

Цель исследования – теоретически обосновать и разработать интерактивную игру «Математический квест» для проблемного обучения.

Объект исследования – разработка компьютерных игр.

Предмет исследования – разработка интерактивной компьютерной игры для проблемного обучения.

Основные задачи исследования:

– анализ теоретических основ компьютерной игры, проблемного обучения и математического квеста, а также их применение в игровой среде: провести классификацию интерактивных компьютерных игр, исследовать элементы и принципы проблемного обучения, интегрировать «Математический квест» в интерактивную игру, осуществить выбор инструментальных средств и алгоритма реализации проекта;

– планирование концепции игры «Математический квест»: определить базовые цели и задачи игры, разработать сюжет и геймплей, которые стимулируют проблемное мышление;

– реализация интерактивной игры «Математический квест» для проблемного обучения: осуществить программирование и дизайн игрового интерфейса, интегрировать системы подсказок для поддержки учащихся в процессе игры, провести функциональное тестирование.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Современное образование сталкивается с вызовом обеспечения активного и заинтересованного участия учащихся в учебном процессе, особенно в области математики. Однако, традиционные методы преподавания не всегда способствуют максимальному усвоению материала и развитию креативного мышления. В свете этого возникает необходимость в поиске новых подходов, способных сделать процесс обучения более увлекательным, продуктивным и интерактивным.

Выбор темы обусловлен актуальностью обучения математике, а также необходимостью разработки методов, способствующих более эффективному усвоению материала. Интерактивные игры доказали свою эффективность в привлечении внимания и мотивации учащихся, и поэтому создание «Математического квеста» является логичным шагом для современного образования. Такой подход способствует улучшению образовательного процесса и созданию позитивного отношения учащихся к изучению математики.

Современное образование сталкивается с вызовами в области привлечения внимания и заинтересованности учащихся, особенно в контексте изучения математики. Традиционные методы обучения могут оказаться недостаточно привлекательными для современных детей, учебный процесс может казаться им скучным и трудным для освоения. Разработка интерактивной игры «Математический квест» направлена на решение этой проблемы, предоставляя ученикам увлекательное образовательное средство, которое совмещает в себе игровой опыт и математические задачи. Актуализация данной темы не только способствует разнообразию методик обучения, но и поддерживает формирование позитивного отношения к математике.

Цель исследования – теоретически обосновать и разработать интерактивную игру «Математический квест» для проблемного обучения.

Объект исследования – разработка компьютерных игр.

Предмет исследования – разработка интерактивной компьютерной игры для проблемного обучения.

Основные задачи исследования:

– анализ теоретических основ компьютерной игры, проблемного обучения и математического квеста, а также их применение в игровой среде: провести классификацию интерактивных компьютерных игр, исследовать элементы и принципы проблемного обучения, интегрировать «Математический квест» в интерактивную игру, осуществить выбор инструментальных средств и алгоритма реализации проекта;

– планирование концепции игры «Математический квест»: определить базовые цели и задачи игры, разработать сюжет и геймплей, которые стимулируют проблемное мышление;

– реализация интерактивной игры «Математический квест» для проблемного обучения: осуществить программирование и дизайн игрового интерфейса, интегрировать системы подсказок для поддержки учащихся в процессе игры, провести функциональное тестирование.

Применяемые методы исследования:

– теоретические: анализ учебной и научно-технической литературы по теме исследования; обобщение; сравнительный анализ;

– эмпирические: опрос; моделирование; тестирование программного продукта.

Практическая значимость исследования заключается в том, что данная разработанная компьютерная интерактивная игра может быть использована педагогами для проведения внеклассных мероприятий, помимо этого, полученный материал исследования может использоваться студентами при написании статей, рефератов, курсовых и дипломных работ.

Структура работы – работа состоит из введения, двух глав, заключения, 5 приложений и списка использованных источников, включающего 42 источника. Результаты работы представлены в 2 таблицах, 22 рисунках. Общий объем работы – 64 печатных листа.

1 Анализ теоретических основ проблемного обучения и математического квеста, их применение в игровой среде

1.1 Сущность и классификация компьютерных игр

Компьютерная игра – это компьютерная программа, построенная с помощью вычислительной техники, служащая для организации игрового процесса (геймплея) и управления им. Игра предлагает взаимодействие игрока с виртуальным миром, игрок так же с помощью взаимодействия с пользовательским интерфейсом игры или устройством ввода получает визуальную обратную связь от устройства изображения, чаще всего показываемый на мониторах компьютера, телефона или телевизорам если игрок играет на игровой консоли. Игры предлагают игрок ряд возможностей, такие как управлять своими игровыми персонажами, решать сложные головоломки, сражаться с противниками и в целом исследовать виртуальный игровой мир и влиять на его ход событий.

Опираясь на работу М. Гейга [8], Ф. С. Гундольфа [12], выделим основные компоненты сущности компьютерных игр:

– геймплей: это компонент, который отвечает за механику игры. Геймплей определяет степень и характер интерактивности между игроком и игрой, как игрок взаимодействует с игровым миром и то как игра реагирует на его действия. От геймплея зависят правила игры, созданного или предложенного разработчиком игры, её испытания для игрока и способы их преодоления;

– интерактивность: это способность давать пользователю возможность влиять на что-то либо в произведении своими решения. Так же в играх интерактивность понимается как поведение других предметов, реагирующих на действия геймера, то есть реакция самой игры. Интерактивность в широком смысле понимается как самостоятельное управление процессом, например, учащийся может самостоятельно управлять процессом освоения каких-либо знаний;

– графика и звук: они напрямую влияют на атмосферу, что бы создать хорошие впечатления реалистичности для игрока. С помощью графики, музыки и звуков можно создать то настроение в игре, которое хочет передать автор игры. Например, создать напряжение и вызвать мурашки при прохождении тяжёлых уровней или же наоборот создать спокойствие, что бы игрок мог проникнуться уникальной атмосферой игры. Графика и звук вдыхают жизнь в игровой мир;

– искусственный интеллект: это мозг, воссозданный внутри компьютера. В играх же это набор алгоритмов, который управляет поведением неигровых персонажей. Он помогает создать иллюзию разума у неигровых персонажей через их поведение. Он используется во многих играх и является важным компонентом геймплея;

– многопользовательский режим: это режим компьютерной игры, в которую могут играть больше одного человека в одном и том же игровом мире одновременно. Играть в такой режим можно либо на одном компьютере, либо на нескольких компьютерах через локальную сеть или же через интернет. Так как этот режим позволяет взаимодействовать с другими игроками, то людям приходится работать совместно в команде для достижения общих целей, что добавляет элемент социальной коммуникации, либо же играть против других игроков, что добавляет элемент соревнования.

Классификация компьютерных игр может основываться на различных критериях, таких как игровой процесс, жанр, аудитория и технологические особенности.

Опираясь на работы Д. В. Денисова [13], С. Н. Ларковича [19], приведём несколько способов классификации компьютерных игр:

а) по игровому процессу:

– линейные игры: игры с заранее заданным сюжетом и последовательностью событий;

– открытый мир: игры, где игроки могут свободно исследовать игровой мир и выполнять задания по своему усмотрению;
– сетевые игры: многопользовательские игры, в которых игроки могут взаимодействовать друг с другом через сеть;
– казуальные игры: простые и легкие игры, предназначенные для кратковременного развлечения.

б) по жанрам:
– экшн (action);
– приключенческие (adventure);
– ролевые (RPG);
– стратегии (strategy);
– симуляторы (simulation);
– спортивные (sports);
– шутеры (shooter);
– головоломки (puzzle).

в) по аудитории:
– для детей;
– для подростков;
– для взрослых;
– семейные игры;
– образовательные игры.

г) по платформам:
– игры для персональных компьютеров;
– консольные игры;
– мобильные игры;
– игры для виртуальной реальности.

д) по технологическим особенностям:
– 2D игры;
– 3D игры;
– игры с поддержкой виртуальной реальности;

– аркадные игры;

– инди-игры.

Это лишь некоторые из способов классификации компьютерных игр, и различные игры могут сочетать в себе элементы из разных классификаций.

Опираясь на работы Б. П. Никитина [27], П. И. Пидкасистого [30], выделим различные жанры развивающих игр, представленных в таблице 1.

Таблица 1 – Жанры развивающих игр

Жанр	**Описание**
Образовательные (Educational)	Игры, разработанные с целью обучения игроков различным навыкам и знаниям
Математические (Mathematics)	Игры, ориентированные на развитие математических навыков и логического мышления
Языковые	Игры, направленные на изучение иностранных языков или развитие грамматических навыков
Научно-популярные (Edutainment)	Игры, сочетающие образовательные аспекты с развлекательным содержанием
Познавательные (Discovery)	Игры, которые ставят перед игроком задачу исследования и открытия новых фактов и явлений
Креативные (Creative)	Игры, которые стимулируют творческое мышление и развитие художественных навыков
Развивающие логику (Logic)	Игры, направленные на развитие логического мышления и решение головоломок

«Интерактивное пространство – это любое пространство, созданное дизайнером, ограниченное правилами, средой или технологиями и интерпретируемое людьми через игру.» [4, с. 55].

Интерактивная игра – это вид компьютерных игр, где игроки взаимодействуют с виртуальными объектами в игре для влияния на игровой процесс и события в игре.

С развитием игровой индустрии, появляются новые технологии и более мощные игровые платформы, благодаря этому разрабатываются новые интерактивные игры, которые становятся все более интересными и реалистичными. Интерактивные игры применяются в образовании, для более эффективного и лёгкого процесса обучения.

Анализ работ Н. И. Герасимова [9], М. В. Кларина [14], О. И. Райса [31] и существующих решений позволил выделить следующие преимущества интерактивных игр:

– позволяют мотивировать обучающихся: захватывающие и увлекательные игры могут стимулировать интерес обучающихся. Повышать их мотивацию к обучению можно благодаря выполнению ряду задач, которые нужно выполнить что бы перейти на следующий уровень и получать какие-либо достижения;

– помогают практиковать навыки: учащимся дают возможность практиковаться и развивать практические навыки что бы они могли закрепить и применить теоретические знания на практике;

– учат работать в команде: многие игры дают возможность поработать в команде, работа в команде в свою очередь способствует развитию навыков коммуникации. Ученики могут сотрудничать что бы совместно решать задачи и достигать общую цель;

– проверяют знания: проверять знания учащихся так же можно с помощью интерактивных игр, для определения их уровня понимания и успеваемости;

– развивают критическое мышление: некоторые интерактивные игры обучают учащихся анализировать ситуации и принимать решения, что способствует развитию их критического мышления.
В целом, интерактивные игры могут быть встроенные в учебные программы как дополнительная форма обучения.

1.2 Проблемное обучение. Математический квест

«Проблемное преподавание – мы определяем как деятельность учителя по созданию системы проблемных ситуаций, изложению учебного материала и его (полному или частичному) объяснению и управлению деятельностью учащихся по усвоению новых знаний как в виде готовых выводов, так и

путем самостоятельной постановки учебных проблем и их решения.» [23, с. 301]. Этот метод обучения способствует развитию критического мышления, аналитических способностей и умений решать проблемы в реальном мире.

«Проблемная ситуация означает, что в процессе деятельности человек натолкнулся на что-то непонятное, неизвестное, тревожащее, удивляющее.» [23, с. 126].

«Проблемное обучение – это современный уровень развития дидактики и педагогической практики.» [23, с. 11]. Так же проблемное обучение – это метод в обучении, когда учитель создаёт проблемную ситуацию что бы ученик мог построить своё понимание предмета и смог самостоятельно решить проблемную ситуацию.

«Проблемное учение – это учебно-познавательная деятельность учащихся по усвоению знаний и способов деятельности путем восприятия объяснений учителя в условиях проблемной ситуации, самостоятельного (или с помощью учителя) анализа проблемных ситуаций, формулировки проблем и их решения посредством (логического и интуитивного) выдвижения предположений, гипотез, их обоснования и доказательства, а также путем проверки правильности решения.» [23, с. 301]. Проблемное учение способствует развитию критического мышления, самостоятельности, коммуникативных и творческих навыков учащихся, что помогает им лучше осваивать учебный материал и успешно применять его на практике.

В отличии от традиционного обучения, ученик эмоционально погружается в предмет и концентрирует внимание на одном вопросе. Не просто энциклопедичность, а создание учителем проблемной обучающей ситуацию, которую ученик должен в свою очередь решить проблему, используя своё критическое мышление, использовать имеющиеся знания и умение находить нестандартные решения. Ученик должен анализировать проблему детально, анализируя опыт других, проводить исследования в

общем. После таких диагностик, ученик должен увидеть проблему и сформулировать её, после этого он должен решить её.

Опираясь на работы Ю. К. Бабанского [1], В. А. Ситарова [33], выделим различные виды проблемного обучения:

– проблема: ученикам предлагается, соответствующая и понятная для их возраста и опыта, проблема, на которую они не знают ответа, и они должны на основе своих базовых знаний самостоятельно решить данную проблему;

– теоретическое исследование: ученики решают теоретические проблемы приходя к новым правилам, законам и теоремам, можно сказать, что они являются первыми исследователями и первооткрывателями;

– поиск практического решения: ученики применяют уже свои имеющиеся знания, например, применяя свои практические навыки в лабораторных заданиях.

Методы проблемного обучения:

– проблемное изложение: этот метод применяется, когда учащиеся имеют недостаточные знания о предмете или впервые сталкиваются с ним, и им трудно установить связи между фактами. В таких случаях инициативу принимает учитель, формулируя проблемные вопросы и задачи, и решает их самостоятельно, в то время как ученики участвуют в процессе поиска решения скорее ментально, чем активно;

– поисковая беседа: этот метод используется в случае, когда школьники имеют базовые знания, необходимые для активного участия в решении учебной проблемы. Это форма обсуждения, в ходе которой учащиеся, опираясь на имеющийся у них материал, ищут и вырабатывают ответы на поставленные вопросы под руководством учителя. Проблемные вопросы предполагают вызов интеллектуальных трудностей и целенаправленный мыслительный поиск. Значительное внимание уделяется

подсказкам и направляющим вопросам. Роль педагога заключается в подведении итогов на основе ответов учеников;

– исследовательская деятельность учащихся: этот метод применяется, когда учащиеся уже обладают достаточными знаниями для формулирования предположений и создания гипотез. Он предполагает самостоятельное формулирование и решение проблемы с последующим контролем учителя. В рамках этого метода предполагается также постановка исследовательских задач: сначала учащиеся выполняют практическую работу по сбору фактов (эксперименты, наблюдения, анализ литературы), а затем проводят теоретический анализ и обобщение полученных результатов.

Анализируя работы С. И. Брызгалова [5], А. В. Брушлинского [6], Т. В. Кудрявцева [17], выявим достоинства проблемного обучения:

– развитие критического мышления: учащиеся сталкиваются с реальными или вымышленными проблемами, что способствует развитию их способности анализировать, оценивать и критически мыслить;

– самостоятельное обучение: этот метод обучения стимулирует самостоятельность учащихся в процессе поиска решений и формирования собственного понимания учебного материала;

– повышение мотивации: интересные и вызывающие трудности проблемные ситуации могут повысить мотивацию учащихся к изучению предмета, так как они чувствуют себя активными участниками процесса;

– практическое применение знаний: проблемное обучение позволяет учащимся применять свои знания на практике, решая реальные или вымышленные проблемы, что способствует их лучшему запоминанию и пониманию;

– развитие коммуникативных навыков: работа в группах над решением проблемных ситуаций способствует развитию коммуникативных навыков, таких как умение слушать и выражать свои мысли;

– активное обучение: проблемное обучение активизирует учащихся, делая их активными участниками учебного процесса, вместо пассивных слушателей;

– стимулирование творческого мышления: решение проблемных ситуаций требует от учащихся творческого мышления и поиска новаторских подходов к решению задач;

– адаптация к реальным ситуациям: проблемное обучение помогает учащимся развивать навыки работы с реальными проблемами, что может быть полезно в их будущей профессиональной деятельности.

Анализируя работы И. Я. Лернера [20], М. И. Махмутова [24], В. П. Романова [30], выявим недостатки проблемного обучения:

– временные затраты: подготовка и проведение проблемных ситуаций требует значительного времени со стороны учителя и учащихся, особенно при первоначальном освоении метода;

– неоднородность группы: ученики могут иметь разный уровень подготовки и способности к самостоятельному решению проблемных задач, что может затруднить совместную работу в группе;

– ограниченное покрытие материала: из-за акцента на решение проблемных ситуаций некоторые аспекты учебного материала могут быть недостаточно освещены, что может привести к недостаточному пониманию основных концепций;

– неэффективность в определенных предметах: в некоторых предметах, требующих строгой последовательности изучения материала или механического запоминания фактов, проблемное обучение может быть менее эффективным;

– не всегда подходит для всех учащихся: некоторые ученики могут испытывать трудности в самостоятельном решении проблемных задач или в работе в группе, что может снизить их мотивацию и эффективность обучения;

– не всегда эффективно для стандартизированных тестов: проблемное обучение может не обеспечивать достаточной подготовки к стандартизированным тестам, что важно для некоторых учебных программ и систем оценки;

– необходимость квалифицированного руководства: проблемное обучение требует от учителя глубокого понимания предметной области и умения эффективно руководить учебным процессом, что может быть сложно для неопытных преподавателей.

Математический квест – это увлекательный и интерактивный способ обучения математике, который включает в себя решение задач и головоломок в контексте приключенческого сюжета. Он может быть как виртуальным, так и физическим, и часто используется как дополнительный инструмент в образовательном процессе.

Анализируя работы Л. Б. Бараева [2], Л. Маврина [21], Л. П. Стасова [34], выделим роли математического квеста в обучении:

– мотивация к изучению математики: квесты предлагают увлекательный контекст, в котором математика становится неотъемлемой частью решения проблем и достижения целей в игровой вселенной. Это может стимулировать интерес учеников к предмету;

– развитие критического мышления: решение математических задач в рамках квеста часто требует от учеников анализа, логического мышления и поиска нестандартных решений. Это способствует развитию их критического мышления и проблемного мышления;

– контекстуализация математических понятий: квесты обычно строятся вокруг определенной темы или концепции математики, что помогает ученикам видеть связь между абстрактными математическими концепциями и их применением в реальном мире;

– коллаборативное обучение: математические квесты часто требуют сотрудничества и коммуникации между учениками для решения задач. Это

способствует развитию навыков командной работы и эффективного обмена идеями;

– индивидуализация обучения: квесты могут быть адаптированы под разные уровни сложности и потребности учеников, что позволяет индивидуализировать процесс обучения и предоставить каждому ученику задания, соответствующие его уровню подготовки.

В целом, математические квесты обогащают образовательный опыт, делая изучение математики увлекательным и интерактивным, а также способствуют развитию широкого спектра навыков и умений у обучающихся.

1.3 Общий алгоритм реализации проекта и выбор инструментальных средств

Выбор инструментальных средств является важным этапом в разработке программного обеспечения. От него зависит эффективность, качество и скорость выполнения проекта.

Основные шаги, которые учитываются при выборе инструментальных средств:

Общий алгоритм реализации проекта

а) определение концепции игры:

– определить основную идею игры: приключение, в котором ученики будут решать математические задачи для продвижения по сюжету;

– разработать общий сюжет и задания для каждого этапа игры.

б) проектирование уровней:

– разбить игру на уровни с различными сложностями и математическими задачами;

– определить последовательность заданий на каждом уровне.

в) создание игрового контента:

– разработать математические задачи для каждого уровня, учитывая их сложность и соответствие учебной программе;

– создать графику, анимации и музыку, чтобы сделать игру более привлекательной для учеников.

г) разработка игровой механики и интерфейса:

– определить механику игры: как ученики будут взаимодействовать с игровым миром и решать задачи;

– создать пользовательский интерфейс, который будет интуитивно понятен и удобен для использования.

д) тестирование и отладка:

– провести тестирование игры на устройстве и с разными группами пользователей, чтобы выявить и исправить возможные ошибки и недочеты;

– собрать обратную связь от тестировщиков и учеников и внесите необходимые изменения.

Выбор инструментальных средств:

– Движок игровой разработки. Unity: мощный движок с широкими возможностями для разработки 2D и 3D игр.

– Графический дизайн и анимация. Aseprite: для создания спрайтов, фонов и других графических элементов.

– Звуковые эффекты и музыка. FL Studio: для создания музыкального сопровождения игры.

– Средства для разработки задач. Языки программирования: C#.

Далее рассмотрим перечисленные инструментальные средства:

а) движок игровой разработки: Unity – это кроссплатформенный игровой движок и интегрированная среда разработки, используемая для создания интерактивных 2D и 3D приложений, включая компьютерные игры, виртуальную реальность и дополненную реальность. Он предоставляет разработчикам широкий спектр инструментов и ресурсов для создания

высококачественных игровых и мультимедийных проектов, а также поддерживает различные платформы, включая ПК, мобильные устройства, консоли и веб. Unity является одним из самых популярных и широко используемых игровых движков в индустрии разработки игр.

Опираясь на работы Д. С. Воронина [7], Е. А. Горячева [11], Д. В. Денисова [13], выделим ключевые особенности и функции Unity:

– кроссплатформенность: Unity позволяет разработчикам создавать игры и приложения, которые могут работать на различных платформах, включая ПК, мобильные устройства, консоли и веб;

– графика и визуализация: Unity обладает мощным движком графики, который поддерживает создание красивых и реалистичных 2D и 3D графических эффектов;

– интегрированная среда разработки (IDE): Unity предоставляет интуитивно понятную среду разработки с широким спектром инструментов для создания, редактирования и управления игровыми ресурсами и сценами;

– активное сообщество: Unity имеет огромное и активное сообщество разработчиков, которые предлагают множество учебных материалов, руководств, ассетов и плагинов для обмена знаниями и опытом;

– мощный движок физики: Unity включает в себя мощный движок физики, который позволяет реализовывать реалистичные физические эффекты, такие как столкновения, симуляция жидкостей и твердых тел;

– мультиплатформенная разработка: Unity обеспечивает возможность создания проектов, которые могут быть экспортированы на различные платформы с минимальными изменениями;

– скриптинг на C#: Unity использует C# в качестве основного языка программирования, что делает разработку более доступной и удобной для многих разработчиков;

– поддержка виртуальной и дополненной реальности: Unity предоставляет инструменты и ресурсы для создания проектов в виртуальной реальности (VR) и дополненной реальности (AR).

Логотип Unity представлен на рисунке 1.

Рисунок 1 – Логотип Unity

Unity является мощным инструментом для разработки игр, предлагая множество преимуществ, таких как кроссплатформенность, богатый набор инструментов и поддержка сообщества. Однако, как и любой другой инструмент, он имеет свои недостатки, включая высокие требования к ресурсам и потенциальные сложности с оптимизацией. Разработчики должны взвесить эти преимущества и недостатки при выборе Unity для своего проекта.

б) графический дизайн и анимация: Aseprite – это специализированное программное обеспечение для создания пиксельной графики и анимации, логотип представлен на рисунке 2. Оно широко используется в разработке игр и других цифровых медиа, благодаря своим мощным инструментам и удобному интерфейсу.

Ключевые особенности и функции Aseprite включают

1) создание пиксельной графики:

– интуитивные инструменты для рисования, позволяющие создавать детализированные пиксельные изображения;

– поддержка различных режимов цветности, включая RGBA и индексированные цвета.

2) анимация:

– инструменты для создания анимации кадр за кадром;

– возможность работы с таймлайном и слоями, что позволяет удобно управлять анимацией;
– поддержка циклов анимации и предварительного просмотра в реальном времени.

3) инструменты редактирования:
– разнообразные кисти, ластики и инструменты для трансформации;
– функции для выделения, копирования и вставки пикселей;
– возможность работы с несколькими слоями и эффектами наложения.

4) экспорт и импорт:
– поддержка различных форматов файлов, включая PNG, GIF, JPEG и другие;
– экспорт анимаций в форматах GIF и sprite sheets, что упрощает их интеграцию в проекты.

5) пользовательский интерфейс:
– настраиваемый интерфейс, позволяющий адаптировать рабочую среду под нужды пользователя;
– поддержка горячих клавиш для быстрого доступа к инструментам и функциям.

Рисунок 2 – Логотип Aseprite

Aseprite является мощным инструментом для художников и разработчиков, занимающихся пиксельной графикой и анимацией, предоставляя все необходимые средства для создания качественных и профессиональных работ.

в) звуковые эффекты и музыка

FL Studio – это мощная цифровая аудиостанция, широко используемая для создания, редактирования и сведения музыки. Она известна своим интуитивным интерфейсом и множеством функций, которые делают её популярной среди музыкантов и продюсеров. Логотип представлен на рисунке 3.

Ключевые особенности и функции FL Studio включают

1) создание и редактирование музыки:
   – многоуровневый секвенсор, позволяющий создавать сложные композиции;
   – Piano Roll для точного редактирования нот и мелодий;
   – поддержка автоматизации параметров, что позволяет создавать динамичные и выразительные треки.
2) инструменты и эффекты:
   – встроенные синтезаторы и сэмплеры для создания разнообразных звуков;
   – поддержка VST и AU плагинов, что позволяет расширять функциональность программы;
   – множество встроенных эффектов, таких как реверберация, задержка, эквалайзеры и компрессоры, для обработки звука.
3) сведение и мастеринг:
   – микшер с поддержкой многодорожечной обработки, что позволяет сводить треки с высокой точностью;
   – плагины для мастеринга, такие как Maximus, для повышения громкости и улучшения звука финального микса;
   – возможность работы с аудиоформатами высокого разрешения.

4) интерфейс и рабочий процесс:

– настраиваемый интерфейс, который можно адаптировать под личные предпочтения;

– поддержка мультитач, что упрощает работу на устройствах с сенсорным экраном;

– функции для быстрого создания аранжировок, такие как Playlist и Clip Tracks.

5) экспорт и импорт:

– экспорт проектов в различные аудиоформаты, включая MP3, WAV, OGG и другие;

– поддержка экспорта треков в формате MIDI для дальнейшей обработки в других DAW;

– импорт аудиофайлов и лупов для использования в проектах.

FL Studio предоставляет все необходимые инструменты для создания качественной музыки и звукового дизайна, делая процесс производства музыки доступным и удобным.

Рисунок 3 – Логотип FL Studio

г) средства для разработки задач: для написания скриптов в Unity используется язык программирования C#, разработанный компанией Microsoft. C# является основным языком для платформы Microsoft .NET и

применяется для создания различных типов приложений, включая настольные, веб-приложения, игры, мобильные приложения и многое другое.

C# – это мощный и универсальный язык программирования, используемый для разработки разнообразных приложений, включая компьютерные игры. Он поддерживает основные принципы объектно-ориентированного программирования, такие как наследование, полиморфизм и инкапсуляция.

Опираясь на работы В. А. Натка [26], Х. Ферроне [37], Д. С. Хокинг [38], выделим ключевые особенности и функции C#:

– простота изучения и использования: C# имеет простой и интуитивно понятный синтаксис, что делает его доступным для новичков в программировании и удобным для опытных разработчиков;

– обширная документация и поддержка: C# имеет обширную документацию, множество учебных материалов и активное сообщество разработчиков, что облегчает изучение и использование языка;

– мощные возможности объектно-ориентированного программирования (ООП): C# поддерживает концепции ООП, такие как наследование, полиморфизм, инкапсуляция, что делает его удобным для организации и структурирования кода в игровых проектах;

– интеграция с платформой Unity: Unity, один из самых популярных игровых движков, использует C# в качестве основного языка программирования, что делает его предпочтительным выбором для разработчиков, создающих игры на этой платформе;

– кроссплатформенность: благодаря платформе .NET и интеграции с различными операционными системами, приложения, написанные на C#, могут быть легко перенесены на различные платформы, включая ПК, мобильные устройства и консоли;

– большое количество библиотек и фреймворков: C# имеет обширную экосистему библиотек и фреймворков, которые могут значительно ускорить

разработку игровых проектов, предоставив разработчикам готовые решения для ряда задач.

В целом, C# представляет собой мощный и универсальный язык программирования, который идеально подходит для разработки игр благодаря своей простоте, гибкости и обширным возможностям.

В первой главе были рассмотрены основные концепции, связанные с разработкой интерактивной игры «Математический квест» на движке Unity. Была подчёркнута значимость интерактивных элементов в образовательных играх для повышения интереса и мотивации учащихся. Были раскрыты принципы проблемного обучения и его применение в формате математического квеста, который стимулирует логическое мышление и навыки решения задач. Был предложен общий алгоритм реализации проекта, включающий этапы от определения целей до тестирования и внедрения, с выбором Unity как основного инструмента разработки благодаря его широким возможностям и поддержке языка программирования C#.

2 Проектирование и разработка интерактивной компьютерной игры для проблемного обучения

2.1 Планирование разработки проекта

Планирование разработки проекта – это процесс определения целей, задач, ресурсов и временных рамок, необходимых для успешной реализации проекта. Он включает в себя определение этапов работы, распределение задач между участниками команды, а также оценку рисков и ресурсов, необходимых для достижения поставленных целей. План разработки проекта помогает управлять процессом выполнения задач, выявлять и решать проблемы на ранних стадиях, а также обеспечивать соблюдение сроков и бюджета проекта.

Описание целевой аудитории – это детальное описание группы людей или организаций, которым предназначен продукт или услуга. Оно включает в себя информацию о характеристиках и потребностях аудитории, ее поведенческих особенностях, демографических данных, интересах, проблемах и целях. Ключевой момент в создании описания целевой аудитории – это понимание, каким образом продукт или услуга могут удовлетворить ее потребности или решить ее проблемы. Это позволяет разработать эффективные маркетинговые стратегии и коммуникационные подходы, которые будут наиболее привлекательны и релевантны для данной аудитории. Для данного проекта целевой аудиторией выбраны дети в возрасте от 9 до 12 лет. Этот выбор обусловлен тем, что математические задачи, представленные в игре, подходят для их уровня знаний и способностей.

Проект представляет собой компьютерную интерактивную игру. Основная цель игры заключается в развитии математического мышления, а также сообразительности игрока.

Проект должен отвечать следующим требованиям:

– понятный игровой процесс: игра должна иметь простой и интуитивно понятный игровой процесс, который легко воспринимается игроками. Это включает в себя понятные правила игры, ясную цель и четкий способ взаимодействия с игровым миром;

– математические задачи в качестве основной механики: основной механикой игры является прохождение уровней путем решения математических задач. Требования к этой функции включают в себя разнообразие задач, доступных для решения, и обеспечение их достаточного уровня сложности для стимулирования игрового процесса;

– простое меню: меню игры должно быть простым и легким для использования. Оно не должно быть перегружено информацией и функциями, чтобы не отвлекать игроков от основного игрового процесса. Требования также могут включать в себя интуитивную навигацию по меню и доступ к основным функциям игры без лишних усилий;

– оптимизация для различных компьютеров: игра должна быть оптимизирована для работы на различных компьютерах и не требовать высоких технических характеристик.

Эти требования к функциональной части являются первостепенными и должны быть выполнены в первую очередь.

Моделирование и описание игрового процесса: UML (Unified Modeling Language) является очень полезным инструментом для моделирования и описания игрового процесса. Он предоставляет стандартные графические элементы и нотации для создания диаграмм, которые помогают визуализировать структуру и поведение игры.

Диаграмма последовательности (Sequence Diagram) в UML представляет собой визуализацию последовательности взаимодействия между объектами или компонентами системы во времени. Она отображает, как объекты взаимодействуют друг с другом в определенной последовательности для выполнения определенной функциональности или сценария.

При запуске компьютерной интерактивной игры «Математический квест» игроку предлагается игровое меню, где он должен будет выбрать режим игры и уровень, после этого игрок приступает к прохождению уровня. После прохождения уровня игроку предлагается выбрать следующий уровень или выйти из игры.

Диаграмма последовательности взаимодействия игрока с системой, созданная с помощью языка моделирования UML, для разрабатываемой интерактивной компьютерной игры «Математический квест» представлена на рисунке 4.

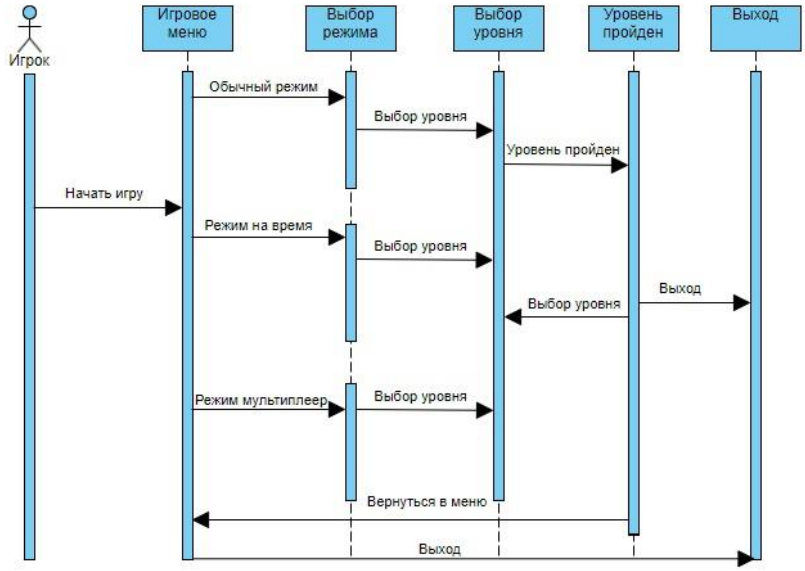

Рисунок 4 – Диаграмма последовательности игры «Математический квест»

Диаграмма последовательности призвана стать неотъемлемым инструментом для анализа и дизайна системы, благодаря своей способности к более глубокому осмыслению и визуализации последовательности взаимодействий между различными элементами и компонентами.

2.2 Реализация интерактивной игры «Математический квест»

Прежде чем начать разработку, первым шагом загрузим специализированное программное обеспечение. Для этого перейдём на официальный сайт Unity и зарегистрируемся. Создание учётной записи UnityID представлено на рисунке 5.

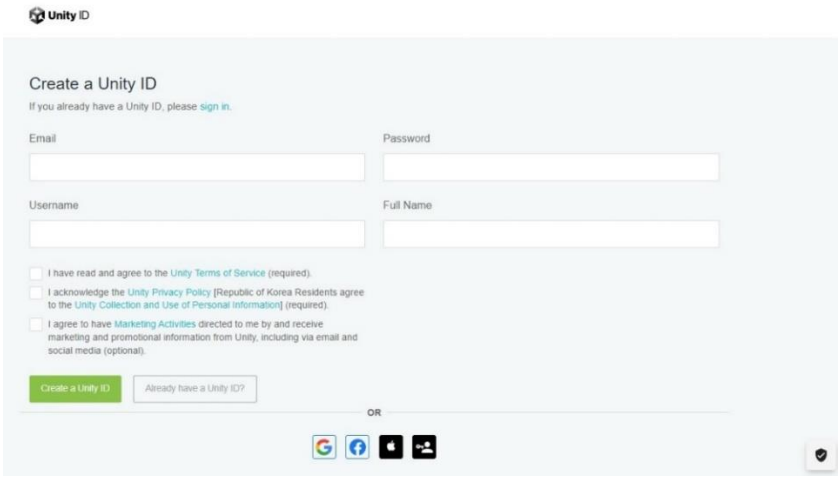

Рисунок 5 – Создание учётной записи UnityID

Для выбора подходящей версии Unity рекомендуется использовать стабильную и последнюю версию, чтобы воспользоваться всеми обновлениями и новыми функциями. При переходе на страницу загрузки Unity необходимо указать операционную систему, под которой будет использоваться Unity. Последняя версия Unity поддерживает различные операционные системы, включая Windows и macOS. В нашем случае будет выбрана версия для Windows. Процесс загрузки Unity изображен на рисунке 6.

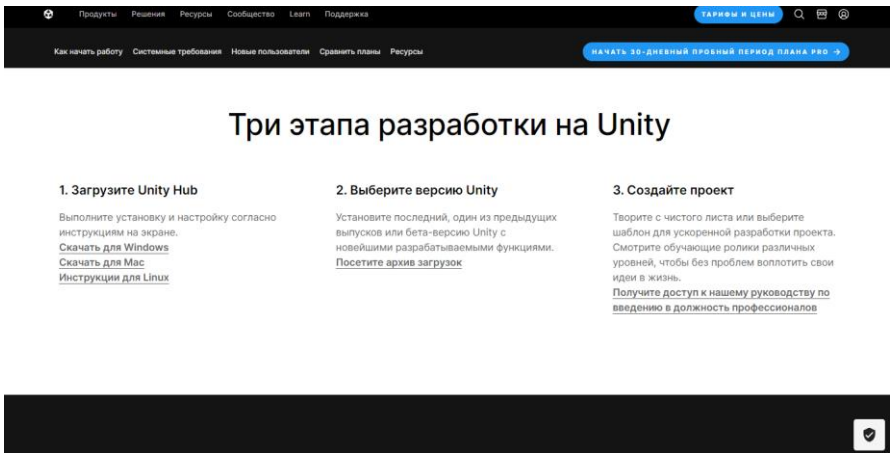

Рисунок 6 – Загрузка Unity

Аккаунт в UnityID позволяет использовать бесплатные ассеты из внутреннего магазина Unity. Unity Hub является платформой, на которой разработчики Unity могут делиться знаниями и опытом. На этом ресурсе пользователи могут получать поддержку, находить вдохновение и следить за последними новостями и трендами в сфере разработки игр на платформе Unity. Помимо этого, Unity Hub предназначен для создания и хранения игровых проектов. Окно UnityHub представлено на рисунке 7.

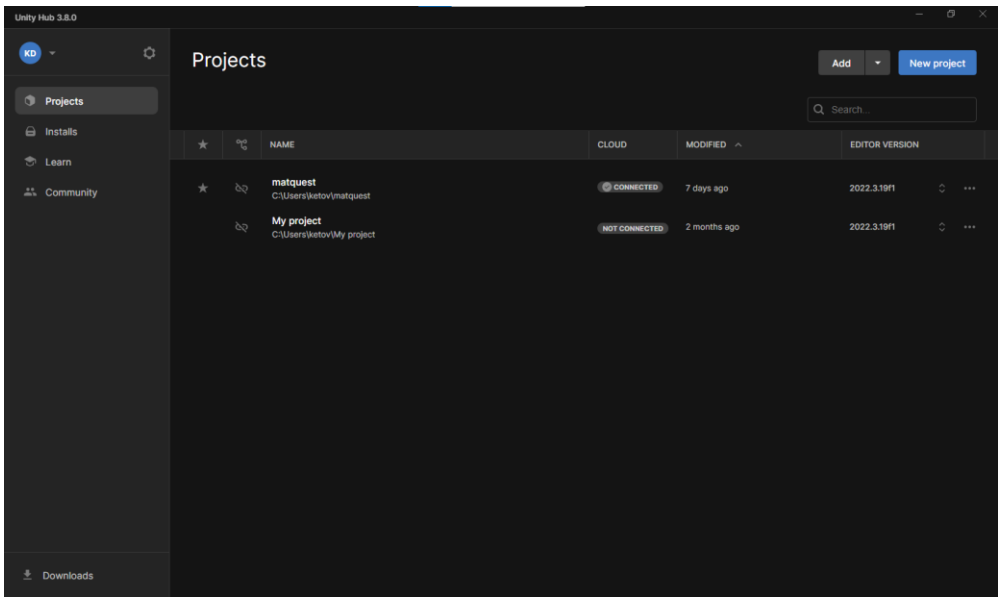

Рисунок 7 – Окно UnityHub

Для создания графики был выбран редактор Aseprite, поскольку он идеально подходит для рисования в стиле «PixelArt». Внешний вид интерфейса программы представлен на рисунке 8.

Для написания программного кода было принято решение использовать Microsoft Visual Studio 2022. На официальном сайте был подобран оптимальный тарифный план для пользователей. Процесс загрузки Microsoft Visual Studio 2022 представлен на рисунке 9.

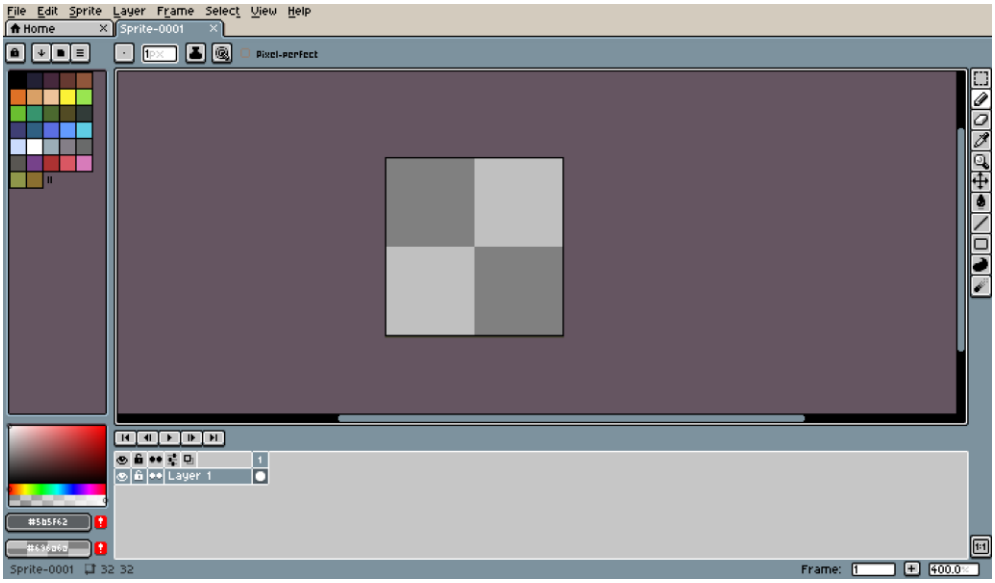

Рисунок 8 – Интерфейс Aseprite

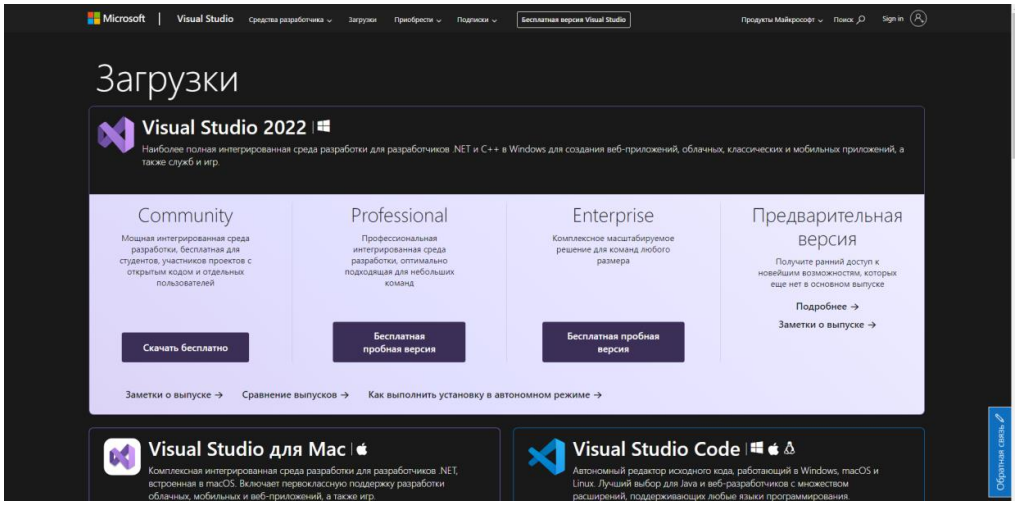

Рисунок 9 – Загрузка MVS 2022

Для создания музыки к игре был выбран FL Studio. Этот инструмент предлагает обширные возможности для создания и редактирования музыкальных композиций, что делает его идеальным выбором для разработки высококачественного звукового сопровождения для игры. Внешний вид интерфейса FL Studio представлен на рисунке 10.

Рисунок 10 – Интерфейс FL Studio

Графическое оформление игры выполняет важные функции:

– создание атмосферы проведения игры: графика помогает создать уникальную атмосферу и визуальный стиль игры, который задает тон и настроение игрового процесса. Это может включать в себя использование определенных цветовых палитр, стилей рисовки и визуальных эффектов;

– передача информации: графические элементы, такие как интерфейс пользователя (UI), значки и индикаторы, передают важную информацию игроку. Это включает в себя отображение здоровья персонажа, уровня энергии, количества очков, текущих задач и других игровых данных;

– улучшение игрового опыта: качественное графическое оформление делает игру более привлекательной и увлекательной для игроков. Оно помогает создать визуально приятный и запоминающийся игровой мир, который мотивирует игроков проводить больше времени в игре;

– поддержка повествования: графика играет ключевую роль в поддержке и развитии сюжета игры. Визуальные элементы, такие как кат-сцены, иллюстрации и анимации, помогают рассказать историю и передать эмоции персонажей;

– управление вниманием игрока: графические элементы направляют внимание игрока на важные аспекты игрового процесса, помогая ему ориентироваться в игровом мире и выполнять задачи. Это может быть достигнуто с помощью визуальных подсказок, выделения ключевых объектов и использования контраста;

– создание идентичности: уникальный графический стиль помогает игре выделиться среди множества других игр и создать запоминающийся бренд. Это важно для маркетинга и привлечения аудитории.

Графическое оформление является неотъемлемой частью общего дизайна игры и играет важную роль в создании увлекательного и привлекательного игрового опыта.

Перед разработкой игры нужно нарисовать спрайты, так как они являются основными графическими элементами, которые будут использоваться в игровом процессе. Спрайты могут включать в себя персонажей, объекты, фоны и другие визуальные элементы, которые создают атмосферу и визуальное представление игры.

Спрайт (Sprite) – это двухмерное изображение или анимация, используемое в компьютерных играх и других графических приложениях для представления объектов, персонажей, элементов интерфейса и других визуальных компонентов. Спрайты часто используются в играх с 2D-графикой, но могут также применяться в 3D-играх для отображения некоторых элементов интерфейса и эффектов.

Для интерактивной игры «Математический квест» было нарисовано главное меню:

1. Создать новый файл: спрайт был создан размером 330 на 124 пикселя, в качестве фона был выбран прозрачный фон.

2. Нарисовать фон: были нарисованы земля и небо, для того что бы добавить на них декоративные элементы, такие как здание, деревья.

3. Украсить и добавить детали: были добавлены дополнительные детали, такие как логотип игры и декоративные элементы, чтобы сделать главное меню более привлекательным и интересным.

4. Добавить кнопки и элементы интерфейса: были добавлены кнопки и другие элементы интерфейса на фоне. Кнопки такие как старт, параметры, выход из игры.

5. Добавить анимацию: была добавлена солнца и анимацию птицы.

6. Настроить размеры: изображение было настроено на размер 1920 на 720 пикселя.

7. Экспортировать спрайт: после завершения рисования спрайта фона, спрайт был экспортирован в формате PNG для сохранения качества изображения.

Спрайт игрового меню, представлен на рисунке 11.

Рисунок 11 – Спрайт игрового меню

Для интерактивной игры «Математический квест» был нарисован игровой персонаж:

1. Создать новый файл: был создан спрайт размером 32 на 32 пикселя, в качестве фона был выбран прозрачный фон.

2. Нарисовать персонажа: был нарисован персонаж и его иконка, персонаж был нарисован ярким, запоминающимся и легко узнаваемым.

3. Настроить размеры: изображение было настроено на размер 500 на 500 пикселей.

4. Экспортировать спрайт: после завершения рисования спрайта, спрайт был экспортирован в формате PNG для сохранения качества изображения.

Спрайты игрового персонажа, представлены на рисунке 12.

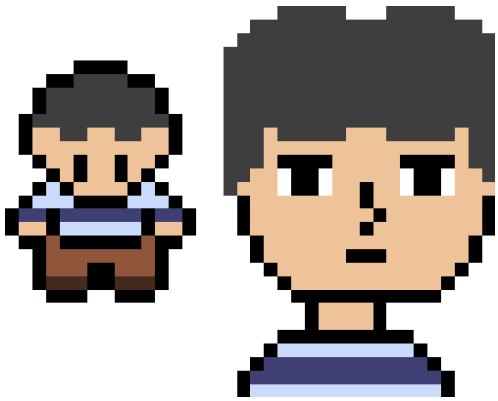

Рисунок 12 – Спрайты игрового персонажа

Для интерактивной игры «Математический квест» были нарисованы боссы:

1. Создать новый файл: был создан спрайт размером 32 на 32 пикселя, в качестве фона был выбран прозрачный фон.

2. Нарисовать боссов: были нарисованы иконки боссов, они были нарисованы уникальными, запоминающимся и легко узнаваемыми.

3. Настроим размеры: изображение было настроено на размер 500 на 500 пикселей.

4. Экспортировать спрайт: после завершения рисования спрайтов, спрайты были экспортированы в формате PNG для сохранения качества изображения.

Спрайты боссов, представлены на рисунке 13.

Рисунок 13 – Спрайты боссов

После того как мы нарисованы спрайты, необходимо загрузить их в Unity для дальнейшей разработки игры. Загрузка спрайтов в Unity представлена на рисунке 14.

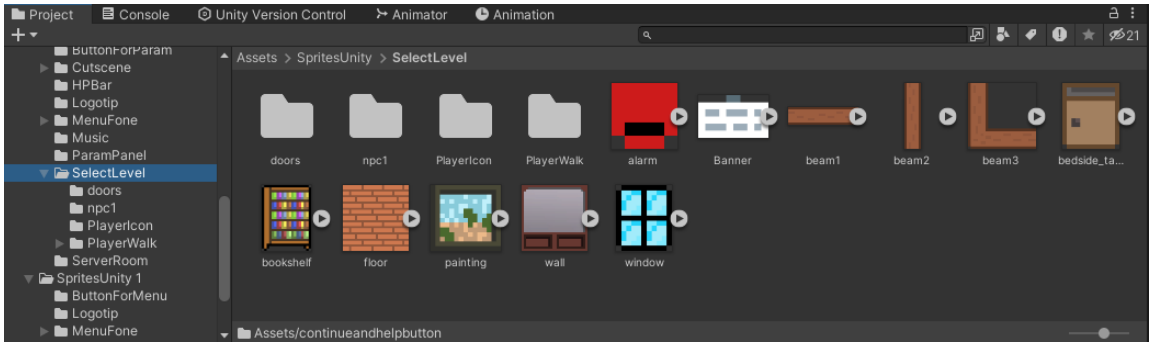

Рисунок 14 – Загрузка спрайтов в Unity

Музыка играет ключевую роль в играх, выполняя несколько важных функций, которые улучшают игровой опыт и делают его более захватывающим и эмоционально насыщенным. Вот основные функции музыки в игре:

– создание атмосферы и настроения: музыка помогает создать определенную атмосферу в игре, отражая настроение различных игровых сцен и ситуаций. Она может усиливать чувства тревоги, радости, напряжения или спокойствия, подчеркивая эмоциональные моменты;

– поддержка повествования: музыкальные треки могут сопровождать сюжетные элементы игры, помогая рассказать историю. Например, эпическая

музыка может играть во время важных событий, а грустная мелодия – во время трагических моментов;

– обратная связь для игрока: музыка может служить инструментом обратной связи для игрока, указывая на изменения в игровом состоянии. Например, быстрый темп музыки может сигнализировать о приближении врагов, а спокойная мелодия – о безопасной зоне;

– усиление игровой механики: музыка может поддерживать и усиливать игровые механики. Например, ритмическая музыка может помогать игроку синхронизировать свои действия с игровым процессом в ритм-играх;

– создание запоминающегося бренда: музыкальные темы и мелодии могут стать узнаваемыми и ассоциироваться с игрой. Хорошо подобранная музыка может стать частью игрового бренда и сделать игру более запоминающейся для игроков;

– поддержка игрового темпа: музыка помогает задавать и поддерживать темп игры. Быстрая музыка может стимулировать активные действия, а медленная – способствовать расслаблению и изучению игрового мира;

– улучшение пользовательского опыта: музыка делает игровой процесс более приятным и увлекательным. Хорошо подобранные музыкальные треки могут улучшить общее впечатление от игры и сделать её более привлекательной для игроков.

Музыка является мощным инструментом, который значительно влияет на восприятие игры и помогает создать более глубокий и увлекательный игровой опыт.

Для интерактивной игры «Математический квест» напишем музыку используя встроенные в FL Studio семплы ударных инструментов. Разработка музыки представлена на рисунке 15.

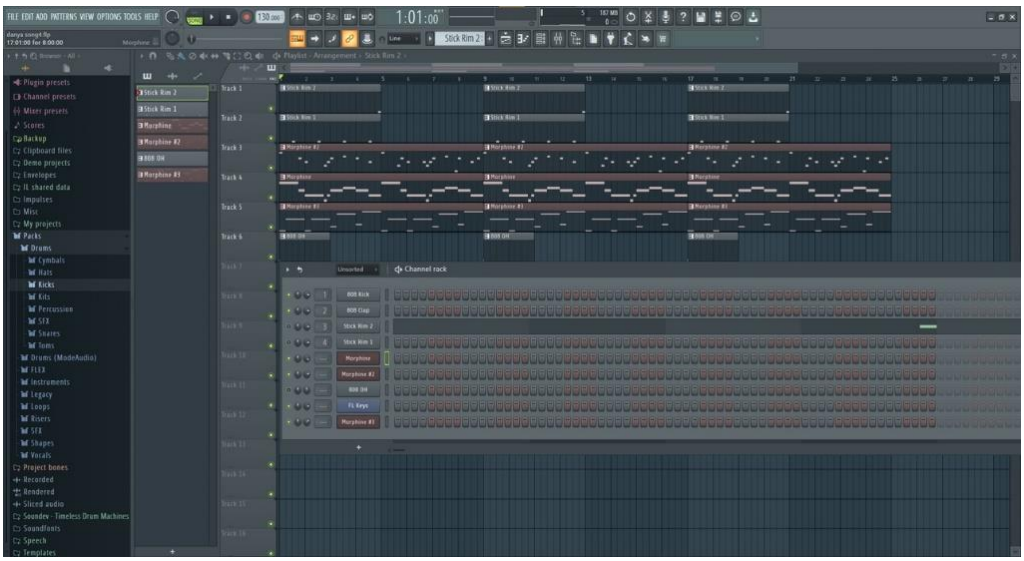

Рисунок 15 – Разработка музыки

Для разработки компьютерной игры мы будем использовать игровой движок Unity и Microsoft Visual Studio 2022 для написания программного кода.

После запуска UnityHub появляется окно создания проекта, окно создания проекта представлено на рисунке 16. В этом окне пользователю предлагается выбрать шаблон приложения. Доступны два варианта шаблонов:

– шаблон 2D предназначен для разработки игр с двухмерной графикой. Это может быть платформер, головоломка, аркада и многие другие жанры, где игровой мир и персонажи представлены в двумерной плоскости;

– шаблон 3D предназначен для разработки игр с трехмерной графикой. Это может быть шутер от первого лица, приключенческая игра, симулятор и многие другие жанры, где игровой мир и персонажи представлены в трехмерном пространстве.

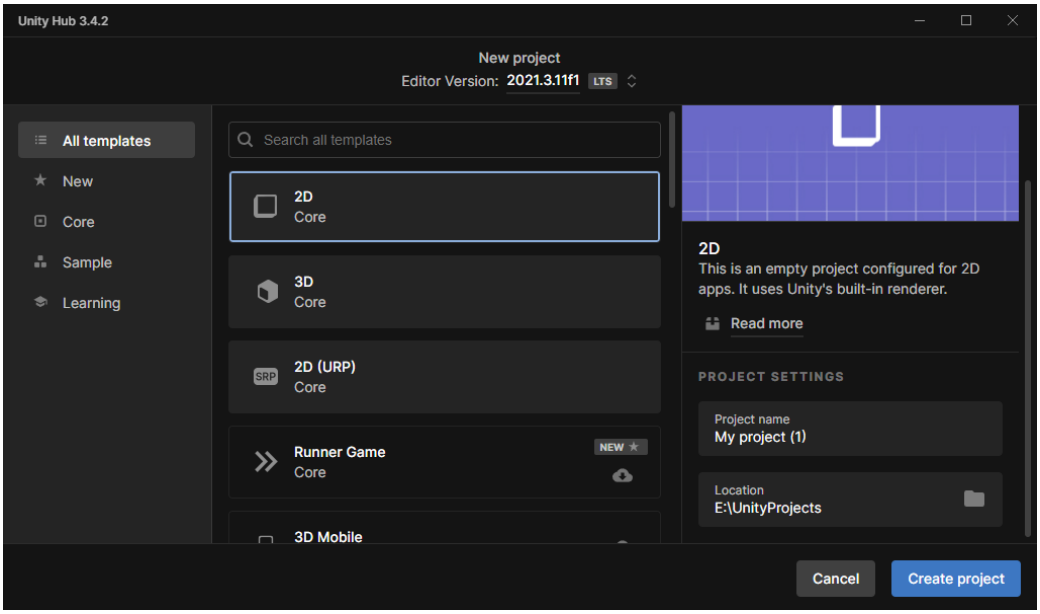

Рисунок 16 – Создание проекта

Выбираем шаблон 2D так как у него интуитивно понятный интерфейс для работы с 2D-объектами, на нём легко работать с двухмерными объектами, так же 2D-игры обычно менее требовательны к ресурсам, что облегчает их оптимизацию для различных устройств.

Создаём проект под названием «matquest». Пустой проект в Unity представлен на рисунке 17.

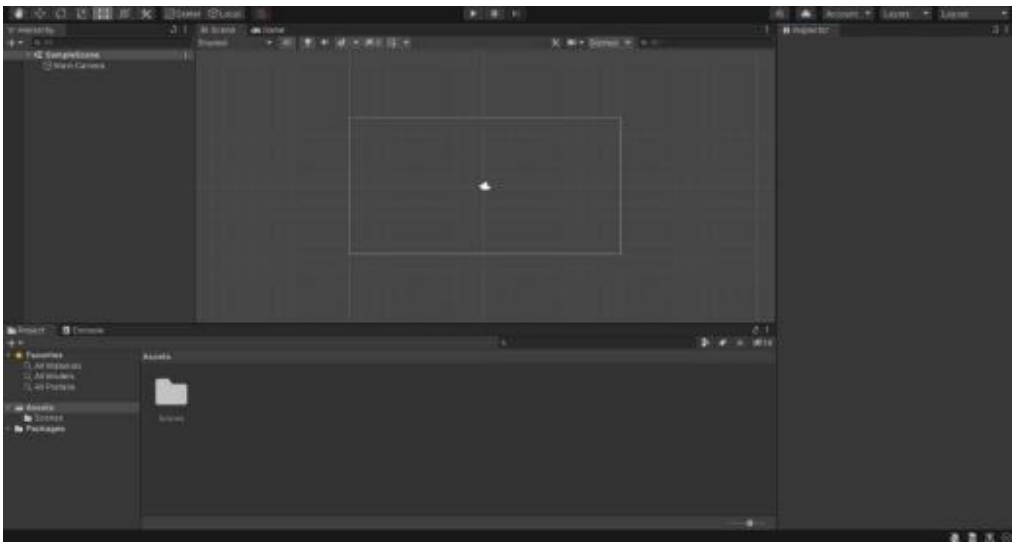

Рисунок 17 – Пустой проект в Unity

Для создания уровня используем ранее нами загруженные спрайты. Добавим иконку и рамку для персонажа и босса, так же добавим кнопки для выхода из игры, выключении музыки, возвращении в меню и подсказок.

Для улучшения взаимодействия пользователей с игрой «Математический квест», мы добавим интерфейсные элементы, такие как кнопки для выбора ответа и сердечки, отвечающие за систему жизней. Эти элементы не только повышают удобство использования, но и способствуют увлекательности игрового процесса, предоставляя визуальную обратную связь и стимулируя игроков. Сцена первого уровня представлен на рисунке 18.

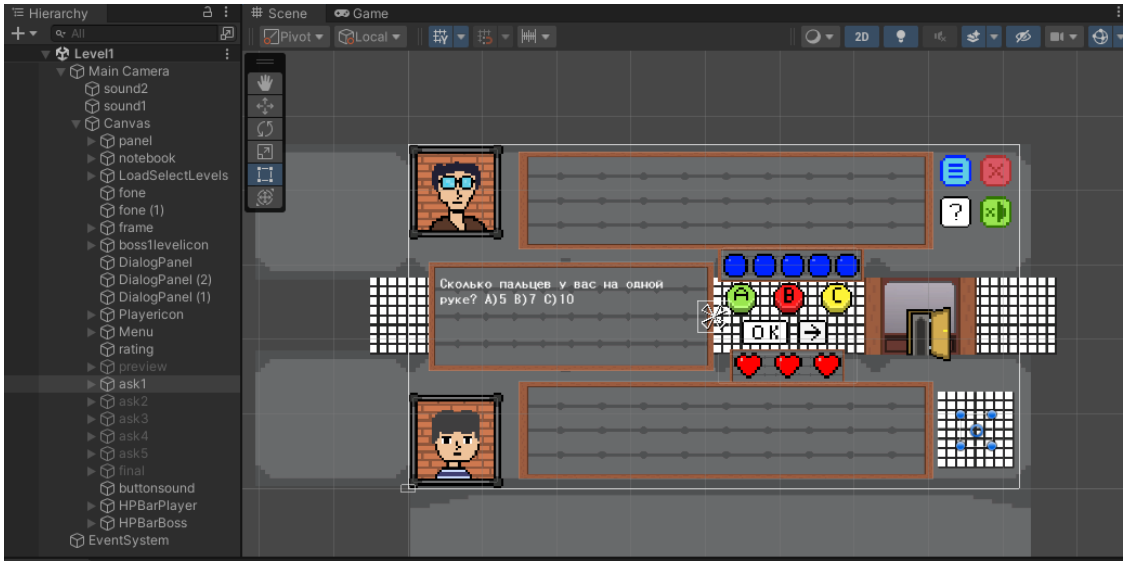

Рисунок 18 – Сцена первого уровня

Для увеличения уровня вызова и добавления интереса к игре «Математический квест», мы создадим сцену поражения, которая будет активироваться после трех ошибок игрока. Это создаст дополнительный элемент в игровом процессе и стимулирует игрока к более внимательному и точному выполнению заданий. Сцена поражения представлена на рисунке 19.

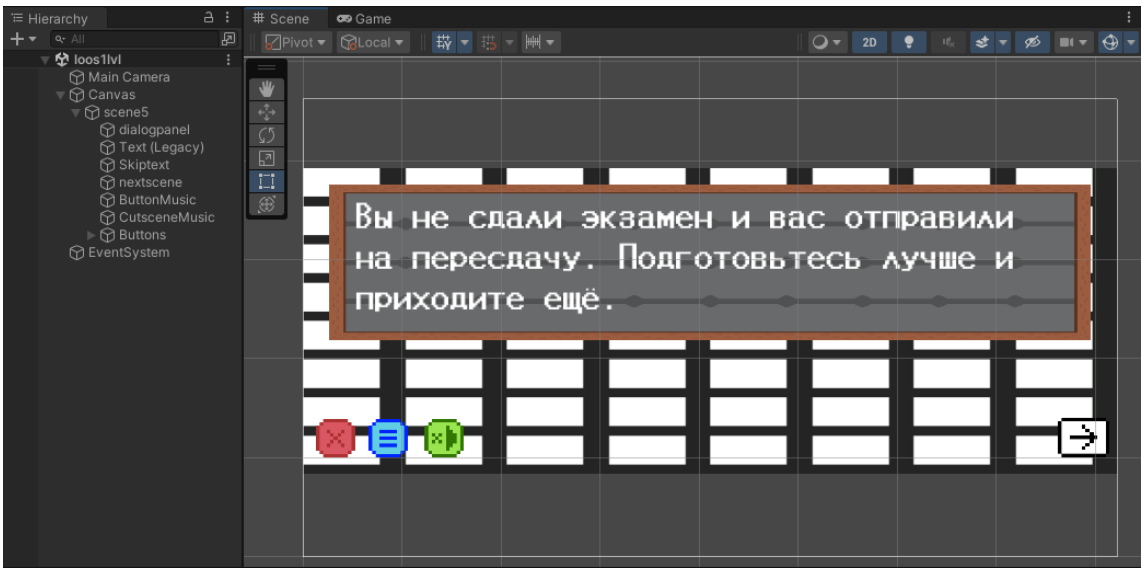

Рисунок 19 – Сцена поражения

Для улучшения игрового процесса и предоставления игрокам возможности выбора уровня, мы создадим сцену выбора уровней. В этой сцене игрок сможет выбрать один из десяти уровней, нажав на дверь, которая будет служить кнопкой перехода к соответствующему уровню. Сцена выбора уровня представлена на рисунке 20.

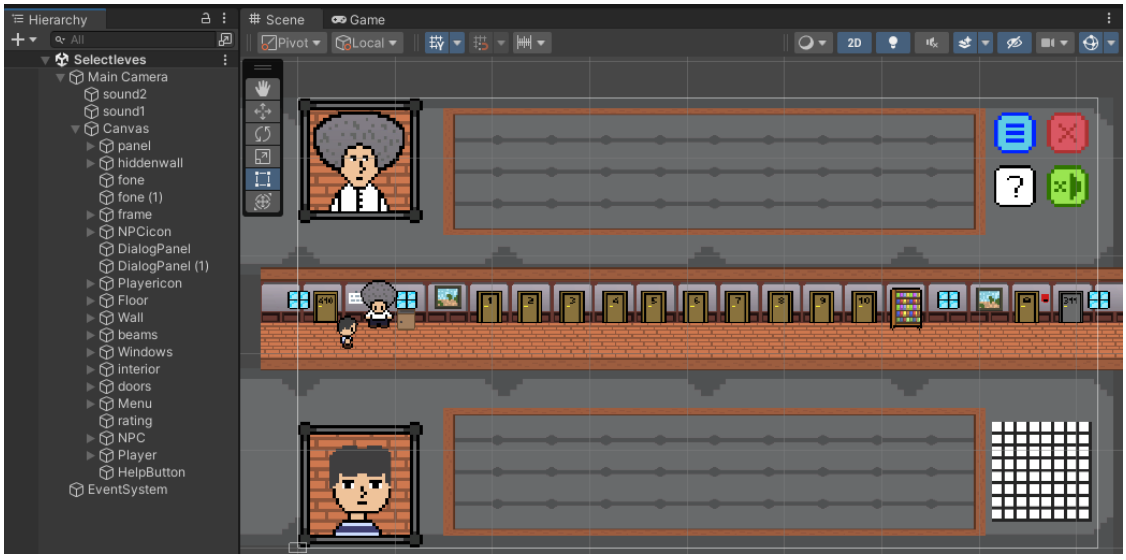

Рисунок 20 – Сцена выбора уровня.

Для повышения интерактивности и разнообразия игрового процесса в игре «Математический квест» мы добавим два дополнительных режима

игры: «На время» и «Мультиплеер». Эти режимы позволят игрокам выбрать стиль игры, который соответствует их предпочтениям, и сделают игру более увлекательной и многогранной.

Режим «На время» вносит элемент срочности и повышает уровень вызова для игроков. В этом режиме игроку предоставляется тридцать секунд для того, чтобы ответить на пять вопросов. В отличие от обычного режима, где игрок может раздумывать неограниченное время, здесь необходимо быстро принимать решения. Если игрок не успеет ответить на все вопросы в отведенное время, ему засчитывается поражение. Уровень с режимом «На время» представлен на рисунке 21.

Этот режим стимулирует игроков к быстрой концентрации и мгновенному решению математических задач, что развивает навыки быстрого мышления и увеличивает напряженность игрового процесса.

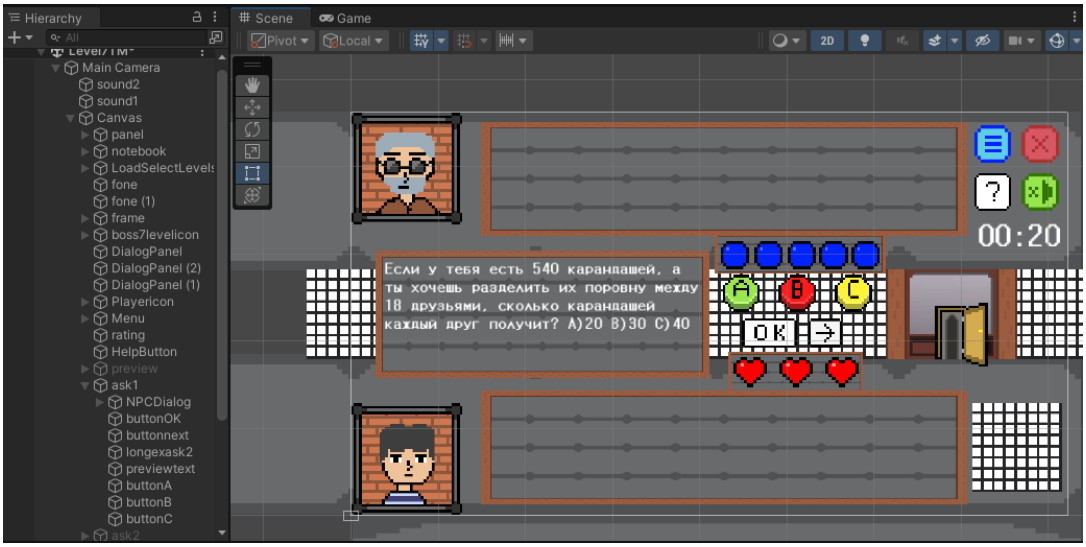

Рисунок 21 – Уровень с режимом «На время»

Режим «Мультиплеер» вносит элемент соревнования между игроками, делая игровой процесс более динамичным и захватывающим. В этом режиме каждый игрок начинает с тремя жизнями. Игра продолжается до тех пор, пока у одного из игроков не закончатся все жизни. Игрок, первым потерявший все жизни, считается проигравшим.

Первому игроку предлагается управление с помощью компьютерной мыши, в то время как второй игрок использует клавиатуру, нажимая кнопки цифр. Такая система управления позволяет легко и удобно разделить управление между двумя игроками, делая процесс игры комфортным для обоих участников. Уровень с режимом «Мультиплеер» представлен на рисунке 22.

Мультиплеерный режим способствует развитию духа соперничества, а также улучшает социальные навыки игроков. Соревнование с друзьями сделает игровой процесс более захватывающим и социально значимым.

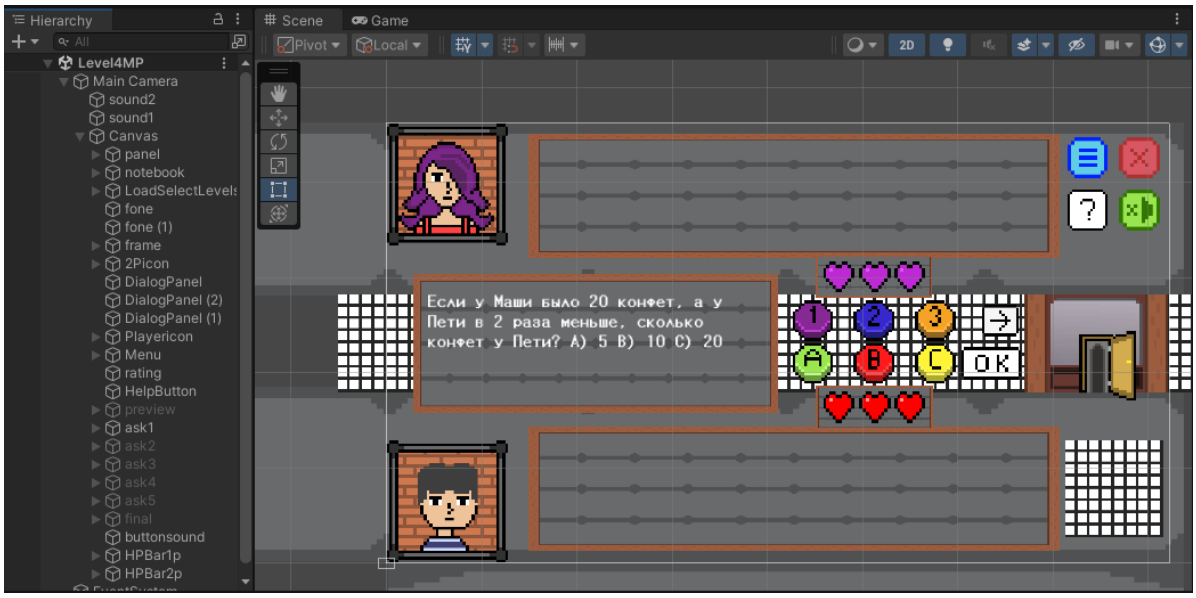

Рисунок 22 – Уровень с режимом «Мультиплеер»

2.3 Функциональное тестирование проекта

Функциональное тестирование представляет собой процесс проверки программного обеспечения, направленный на подтверждение правильности выполнения его функциональных требований. Главной целью функционального тестирования является обеспечение соответствия разработанной системы ожиданиям и спецификациям, установленным на этапе проектирования.

Основные характеристики функционального тестирования:

– проверка функций: функциональное тестирование проверяет все функции и возможности программного обеспечения. Каждая функция рассматривается отдельно, и тесты проводят для каждой из них, чтобы подтвердить, что она работает правильно;

– использование тестовых случаев: для функционального тестирования разрабатываются специальные тестовые случаи, которые охватывают все аспекты функциональности программы. Эти тестовые случаи включают в себя различные сценарии использования, чтобы удостовериться, что система ведет себя корректно в различных ситуациях;

– валидация ввода и вывода: в ходе функционального тестирования проверяется правильность обработки входных данных и получения ожидаемых выходных данных. Это включает проверку всех форм ввода и вывода, включая пользовательский интерфейс, базы данных, файлы и API;

– имитация пользовательского поведения: функциональное тестирование часто включает имитацию действий реальных пользователей, чтобы проверить, как система реагирует на различные действия и сценарии. Это позволяет выявить возможные проблемы и ошибки;

– черный ящик: функциональное тестирование часто рассматривается как тестирование «чёрного ящика», поскольку тестировщики не обязательно должны знать внутреннюю структуру и код программы. Они сосредоточены на проверке функциональности на основе спецификаций и требований.

Функциональное тестирование играет критическую роль в процессе разработки программного обеспечения. Оно позволяет выявлять и устранять ошибки на ранних стадиях разработки, обеспечивая высокое качество и надежность конечного продукта. Проверяя соответствие системы функциональным требованиям и ожиданиям пользователей, функциональное тестирование способствует созданию программного обеспечения, удовлетворяющего потребности и ожидания конечных пользователей.

Результаты тестирования представлены в таблице 2.

Таблица 2 – Функциональное тестирование

№	Тест	Описание	Ожидаемый результат	Прохождение теста
1	Запуск игры	Проверка запуска игры и загрузки главного меню	Главная сцена загружается, отображается главное меню.	Да
2	Навигация по меню	Проверка переходов между различными пунктами меню	Все пункты меню работают корректно и возвращаются в главное меню.	Да
3	Решение математических задач	Проверка корректности обработки решений задач	Корректные решения принимаются, некорректные решения отклоняются.	Да
4	Таймер в режиме «На время»	Проверка работы таймера в режиме «На время»	Таймер отображается и работает корректно, по истечении времени — поражение	Да
5	Система жизней	Проверка работы системы жизней	Количество жизней уменьшается при ошибках, игра заканчивается при потере всех жизней	Да
6	Управление в режиме «Мультиплеер»	Проверка работы управления для двух игроков	Управление корректно работает для обоих игроков	Да
7	Выход из игры	Проверка корректного выхода из игры	Игра корректно закрывается	Да
8	Музыкальное сопровождение	Проверка включения/выключения музыкального сопровождения	Музыка включается/выключается корректно.	Да

Во второй главе дипломной работы была представлена практическая реализация проекта по созданию интерактивной игры «Математический квест» на движке Unity. Процесс разработки включал в себя несколько ключевых этапов, начиная с концептуального проектирования и заканчивая функциональным тестированием.

Графические элементы игры были созданы с помощью программы Aseprite, что позволило достичь высокого качества пиксель-арта и придать игре уникальный визуальный стиль. Музыкальное сопровождение было написано в FL Studio, обеспечив высокое качество звукового оформления,

которое дополняет игровую атмосферу и способствует вовлечению игроков. В игре были добавлены режимы мультиплеера и игры на время, что значительно расширило её функциональные возможности и повысило уровень взаимодействия между игроками.

Интеграция всех компонентов в Unity прошла успешно, что позволило создать целостный и привлекательный игровой продукт. Программирование игровых механик на языке C# обеспечило гибкость и надёжность игрового процесса. Тестирование показало, что игра функционирует стабильно, а её образовательные цели были достигнуты, подтверждая эффективность использования разработанных методов и инструментов. Режимы мультиплеера и игры на время сделали игровой процесс более динамичным и соревновательным, что дополнительно повышает мотивацию учащихся к обучению.

ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломного проекта была разработана интерактивная игра «Математический квест», реализованная в режиме проблемного обучения. В процессе работы над проектом были выполнены следующие задачи:

1. Проведена классификация интерактивных компьютерных игр.
2. Исследованы элементы и принципы проблемного обучения.
3. Интегрирован «Математический квест» в интерактивную игру.
4. Проведен анализ требований к игре и целевой аудитории, что позволило определить основные функции и механики игрового процесса.
5. Разработаны и протестированы спрайты персонажей и окружения, созданные с использованием графического редактора Aseprite, оптимизированного для рисования в стиле PixelArt.
6. С использованием среды разработки Unity и интегрированного редактора Microsoft Visual Studio 2022 был создан функциональный и гибкий игровой проект.
7. Внедрены два дополнительных режима игры: «На время» и «Мультиплеер», что значительно расширило возможности и вариативность игрового процесса.
8. Разработаны различные сцены игры, включая главное меню, сцену выбора уровней, игровые уровни, сцену поражения и дополнительные интерфейсы для настройки и управления.
9. Созданы и интегрированы музыкальные сопровождения и звуковые эффекты, написанные в FL Studio, которые улучшили общее восприятие игры и повысили уровень вовлеченности пользователей.

Функциональное тестирование показало, что игра «Математический квест» соответствует заявленным требованиям и обеспечивает пользователю комфортный и увлекательный игровой процесс. Проведенное функциональное тестирование подтвердило корректность работы всех

ключевых компонентов игры, включая систему жизней, таймер, управление персонажами и интерфейсные элементы.

Результаты работы над проектом продемонстрировали успешное применение современных инструментов и технологий для разработки интерактивных обучающих игр. Разработанная игра способствует развитию логического мышления, скорость реакции. В дальнейшем, проект может быть дополнен новыми уровнями, задачами и функциональными возможностями, что обеспечит его актуальность и востребованность среди широкой аудитории.

Таким образом, цели дипломного проекта достигнуты, а полученные результаты подтверждают эффективность выбранного подхода и методов разработки.

Результаты исследований представлены на следующих научных мероприятиях:

1. II Всероссийский молодежный научный форум «Современное педагогическое образование: теоретический и прикладной аспекты» (г. Лесосибирск, ЛПИ – филиал СФУ, 10 апреля 2023 г.).

2. VII Всероссийская научно-практическая конференция преподавателей, учителей, студентов и молодых ученых «Актуальные проблемы преподавателей дисциплин естественнонаучного цикла» (г. Лесосибирск, ЛПИ – филиал СФУ, 1 ноября 2023 г.).

Принята к публикации статья:

1. Кетов, Д.А. Проектирование и разработка интерактивной игры «Математический квест» для проблемного обучения / Д.А. Кетов // Всероссийская научно-практическая конференция «Современное педагогическое образование: теоретический и прикладной аспекты» – Лесосибирск, 2024 – 3 стр.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Бабанский, Ю. К. Проблемное обучение как средство повышения эффективности учения школьников / Ю. К. Бабанский. – Ростов-на-Дону : Феникс, 1970. – 300 с.

2. Бараева, Л. Б. Математика для дошкольников в играх и упражнениях / Л. Б. Бараева, С. Ю. Кондратьева. – Москва : Карго, 2012. – 500 с.

3. Богуславская, З. М. Развивающие игры для детей дошкольного возраста / З. М. Богуславская, Е. О. Смирнова. – Москва : Просвещение, 2018. – 143 с.

4. Бонд Г. Д. Unity и C# геймдев от идеи до реализации / Д. Г. Бонд. – Санкт-Петербург : Питер, 2017. – 640 с.

5. Брызгалова, С. И. Проблемное обучение в школе: учеб. пособие / С. И. Брызгалова. – Калининград : Калининградский университет, 2007 – 91 с.

6. Брушлинский, А. В. Психология мышления и проблемное обучение / А. В. Брушлинский. – Москва : Знание, 1983. – 96 с.

7. Воронин, Д. С. Unity 2D. Создание игр и приложений / Д. С. Воронин. – Москва : ДМК Пресс, 2019. – 250 с.

8. Гейг, М. Разработка игр на Unity за 24 часа / М. Гейг. – Москва : Бомбора, 2020. – 466 с.

9. Герасимова, Н. И. Деловая игра как интерактивный метод обучения речевой деятельности / Н. И. Герасимова. – М : Среднее профессиональное образование, 2020. – 54 с.

10. Гик, Е. Я. Занимательные математические игры / Е. Я. Гик. – Москва : Наука, 1987. – 160 с.

11. Горячева, Е. А. Unity 2D: создание аркадных игр с нуля / Е. А. Горячева. – Санкт-Петербург : Питер, 2020. – 218 с.

12. Гундольф, Ф. С. Игры. Геймдизайн. Исследование игр / Ф. С. Гундольф. – Москва : Гуманитарный центр, 2021. – 250 с.

13. Денисов, Д. В. Разработка игры в Unity. С нуля до реализации / Д. В. Денисов. – ЛитРес:Самиздат, 2021.

14. Кларин, М. В. Интерактивное обучение - инструмент освоения нового опыта / М. В. Кларин. – Спб : Педагогика, 2018. – 156 с.

15. Климкович, Е. В. Квест как современная форма популяризации гуманитарного знания / Е. В. Климкович. – Москва : Вестник университета правительства Москвы, 2019. – 357 с.

16. Корнилов, А. В. UNITY. Полное руководство / А. В. Корнилов. – Санкт-Петербург : Наука и техник, 2020. – 432 с.

17. Кудрявцев, Т. В. Проблемное обучение: истоки, сущность, перспективы / Т. В. Кудрявцев. – Москва : Знание, 2001. – 85 с.

18. Ларкович, С. Н. Справочник Unity. Кратко, быстро, под рукой / С. Н. Ларкович. – Санкт-Петербург : Наука и техника, 2020. – 288 с.

19. Ларкович, С. Н. Привет, Unity! Моя первая книга по созданию игр / С. Н. Ларкович. – Россия : Наука и техника, 2021. – 288 с.

20. Лернер, И. Я. Проблемное обучение / И. Я. Лернер. – Москва : Знание, 1974. – 64 с.

21. Маврина, Л. Математические игры для дошкольников / Л. Маврина. – Москва : Стрекоза, 2012. – 665 с.

22. Майк, Г. Разработка игр на Unity 2018 за 24 часа / Майк Г. – Москва : Эксмо, 2020. – 464 с.

23. Махмутов, М. И. Проблемное обучение / М. И. Махмутов. – Москва : Педагогика, 1975. – 368 с.

24. Махмутов, М. И. Проблемное обучение. Основные вопросы теории / М. И. Махмутов. – Москва : Педагогика, 2016. – 456 с.

25. Мэннинг, Д. Unity и для разработчика / Д. Мэннинг. – Санкт-Петербург : Питер, 2018.

26. Натка, В. А. Unity в действии с примерами на C#. Создание игр и приложений / В. А. Натка, Н. Г. Кожаев. – Санкт-Петербург : Питер, 2019. – 212 с.

27. Никитин, Б. П. Ступеньки творчества или развивающие игры / Б. П. Никитин. – Москва : Педагогика, 1990. – 160 с.

28. Орлов, С. А. Теория и практика языков программирования / С. А. Орлов. – СПб. : Питер, 2013. – 688 с.

29. Петров, Е. П. Unity 2D: создание вашей первой игры / Е. П. Петров. – Санкт-Петербург : ДМК Пресс, 2017. – 167 с.

30. Пидкасистый, П. И. Технология игры в обучении и развитии / П. И. Пидкасистый, Ж. С. Хайдаров. – М : Эксмо, 1996. – 169 с.

31. Райс, О. И. Интерактивные технологии в обучении. Педагогика нового времени / О. И. Райс. – Москва : Ridero, 2020. – 80 с.

32. Рубачева, Е. Б. Сборник тематических квестов для учащихся 1-9 классов / Е. Б. Рубачева. – СПб : ГБУ ДО «У Вознесенского моста», 2019. – 42 с.

33. Ситаров, В. А. Проблемное обучение как одно из направлений современных технологий обучения / В. А. Ситаров. – Москва : Знание. Понимание. Умение, 2009. – 270 с.

34. Стасова, Л. П. Развивающие математические игры-занятия в ДОУ / Л. П. Стасова. – Воронеж : ЧП Лакоценин С. С., 2008. – 108 с.

35. Суслов, В. Квесты для обучения и развлечения / В. Суслов. – Москва : Солон-Пресс, 2020. – 96 с.

36. Торн, А. Основы анимации в Unity / А. Торн. – Москва : ДМК Пресс, 2016. – 176 с.

37. Харрисон, Ф. Изучаем C# через разработку игр на Unity / Ф. Харрисон. – Санкт-Петербург : Питер, 2022. – 400 с.

38. Хокинг, Д. С. Unity в действии. Мультиплатформенная разработка на C# / Д. С. Хокинг. – Питер : Питер, 2019. – 352 с.

39. Швабер, К. Исчерпывающее руководство по Скраму: Правила Игры / К. Швабер. – Санкт-Петербург : Питер, 2017. – 26 с.

40. Шмаков, С. А. Игры учащихся – феномен культуры / С. А. Шмаков. – М : Новая школа, 1994. – 203 с.

41. Троелсен Э. Язык программирования C# 6.0 и платформа .NET 4.6 / Э. Троелсен, Ф. Джепикс. – М. : Вильямс, 2017. – 1440 с.

42. Яблоков, К. М. Исторические компьютерные игры как способ моделирования исторической информации // История и математика: Анализ и моделирование социально–исторических процессов / К. М. Яблоков. – Москва : КомКнига, 2020. – 204 с.

ПРИЛОЖЕНИЕ А

Листинг C# кода, кнопки для перехода на другие сцены

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class doors : MonoBehaviour
{
   public void StartLevel1()
   {
      SceneManager.LoadScene("Level1");
   }
   public void StartLevel2()
   {
      SceneManager.LoadScene("Level2");
   }
   public void StartLevel3()
   {
      SceneManager.LoadScene("Level3");
   }
   public void StartLevel4()
   {
      SceneManager.LoadScene("Level4");
   }
   public void StartLevel5()
   {
      SceneManager.LoadScene("Level5");
   }
   public void StartLevel6()
   {
      SceneManager.LoadScene("Level6");
   }
   public void StartLevel7()
   {
      SceneManager.LoadScene("Level7");
   }

   public void StartLevel8()
   {
```

```
        SceneManager.LoadScene("Level8");
    }
    public void StartLevel9()
    {
        SceneManager.LoadScene("Level9");
    }
    public void StartLevel10()
    {
        SceneManager.LoadScene("Level10");
    }
public void StartFinalCutscene()
{
    SceneManager.LoadScene("CutsceneFinal");
}
    public void StartLevelSelectFinal()
    {
        SceneManager.LoadScene("Selectlevesfinal");
    }
    public void passed1lvl()
    {
        SceneManager.LoadScene("1lvlpassed");
    }
    public void passed2lvl()
    {
        SceneManager.LoadScene("2lvlpassed");
    }
    public void passed3lvl()
    {
        SceneManager.LoadScene("3lvlpassed");
    }
    public void passed4lvl()
    {
        SceneManager.LoadScene("4lvlpassed");

    public void passed5lvl()
    {
        SceneManager.LoadScene("5lvlpassed");
    }
    public void passed6lvl()
    {
        SceneManager.LoadScene("7lvlpassed");
    }
    public void passed7lvl()
    {
```

```
    SceneManager.LoadScene("7lvlpassed");
}
public void passed8lvl()
{
    SceneManager.LoadScene("8lvlpassed");
}
public void passed9lvl()
{
    SceneManager.LoadScene("9lvlpassed");
}
public void ServerRoom()
{
    SceneManager.LoadScene("ServerRoom");
}
public void SelectlevesTM()
{
    SceneManager.LoadScene("SelectlevesTM");
}
public void Level1TM()
{
    SceneManager.LoadScene("Level1TM");
}
public void Level2TM()
{
    SceneManager.LoadScene("Level2TM");
}
public void Level3TM()
{
    SceneManager.LoadScene("Level3TM");
}
public void Level4TM()
{
    SceneManager.LoadScene("Level4TM");
}
public void Level5TM()
{
    SceneManager.LoadScene("Level5TM");
}
public void Level6TM()
{
    SceneManager.LoadScene("Level6TM");
}
public void Level7TM()
{
```

```
    SceneManager.LoadScene("Level7TM");
}
public void Level8TM()
{
    SceneManager.LoadScene("Level8TM");
}
public void Level9TM()
{
    SceneManager.LoadScene("Level9TM");
}
public void Level10TM()
{
    SceneManager.LoadScene("Level10TM");
}
public void SelectlevesMP()
{
    SceneManager.LoadScene("SelectlevesMP");
}
public void Level1MP()
{
    SceneManager.LoadScene("Level1MP");
}
public void Level2MP()
{
    SceneManager.LoadScene("Level2MP");
}
public void Level3MP()
{
    SceneManager.LoadScene("Level3MP");
}
public void Level4MP()
{
    SceneManager.LoadScene("Level4MP");
}
public void Level5MP()
{
    SceneManager.LoadScene("Level5MP");
}
public void Level6MP()
{
    SceneManager.LoadScene("Level6MP");
}
public void Level7MP()
{
```

```
        SceneManager.LoadScene("Level7MP");
    }
    public void Level8MP()
    {
        SceneManager.LoadScene("Level8MP");
    }
    public void Level9MP()
    {
        SceneManager.LoadScene("Level9MP");
    }
    public void Level10MP()
    {
        SceneManager.LoadScene("Level10MP");
    }
}
```

ПРИЛОЖЕНИЕ Б

Листинг C# кода, таймер для режима «На время»

```
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;
public class TimerScript : MonoBehaviour
{
   public float timerDuration = 30f;
   private float currentTime;
   private bool timerActive = false;
   public Text timerText;
   public Button startButton;
   void Start()
   {
      currentTime = timerDuration;
      startButton.onClick.AddListener(StartTimer);
   }
   void Update()
   {
      if (timerActive)
      {
         currentTime -= Time.deltaTime;
         int minutes = Mathf.FloorToInt(currentTime / 60f);
         int seconds = Mathf.FloorToInt(currentTime % 60f);
         timerText.text = string.Format("{0:00}:{1:00}", minutes, seconds);
         if (currentTime <= 0)
         {
            timerActive = false;
            timerText.text = "00:00";
            SceneManager.LoadScene("loosTM");
         }
      }
   }
   void StartTimer()
   {
      timerActive = true;  }
}
```

ПРИЛОЖЕНИЕ В

Листинг C# кода, система жизней

```
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class TimeModHPBarPlayer : MonoBehaviour
{
    public int hearts = 3;
    public Image[] heartImages;
    public Sprite fullHeartSprite;
    public Sprite emptyHeartSprite;

    void Start()
    {
        UpdateHeartsUI();
    }

    public void OnWrongButtonPressed()
    {
        hearts--;
        UpdateHeartsUI();

        if (hearts <= 0)
        {
            SceneManager.LoadScene("SelectlevesTM");
        }
    }
    void UpdateHeartsUI()
    {
        for (int i = 0; i < heartImages.Length; i++)
        {
            if (i < hearts)
            {
                heartImages[i].sprite = fullHeartSprite;}
            else
            {
                heartImages[i].sprite = emptyHeartSprite;}
        }
    }
}
```

ПРИЛОЖЕНИЕ Г

Листинг C# кода, управление для второго игрока

```
using UnityEngine;
using UnityEngine.UI;

public class KeyboardButtonController: MonoBehaviour
{
    public Button button1;
    public Sprite button1NormalSprite;
    public Sprite button1PressedSprite;

    public Button button2;
    public Sprite button2NormalSprite;
    public Sprite button2PressedSprite;

    public Button button3;
    public Sprite button3NormalSprite;
    public Sprite button3PressedSprite;

    public AudioClip buttonSound;

    public GameObject ask1;
    public GameObject ask2;

    public GameObject p1HPBarObject;
    public GameObject p2HPBarObject;

    private P1HPBar p1HPBar;
    private P2HPBar p2HPBar;

    private bool isButton1Pressed = false;
    private bool isButton2Pressed = false;
    private bool isButton3Pressed = false;

    private void Start()
    {
        // Получаем ссылки на компоненты P1HPBar и P2HPBar
        p1HPBar = p1HPBarObject.GetComponent<P1HPBar>();
        p2HPBar = p2HPBarObject.GetComponent<P2HPBar>();

        button1.image.sprite = button1NormalSprite;
        button2.image.sprite = button2NormalSprite;
```

```
    button3.image.sprite = button3NormalSprite;

    button1.onClick.AddListener(() => OnButton1Click());
    button2.onClick.AddListener(() => OnButton2Click());
    button3.onClick.AddListener(() => OnButton3Click());

    button1.interactable = false;
    button2.interactable = false;
    button3.interactable = false;
}

private void Update()
{
    if (Input.GetKeyDown(KeyCode.Alpha1))
    {
        isButton1Pressed = true;
        button1.image.sprite = button1PressedSprite;
    }
    else if (Input.GetKeyDown(KeyCode.Alpha2))
    {
        isButton2Pressed = true;
        button2.image.sprite = button2PressedSprite;
    }
    else if (Input.GetKeyDown(KeyCode.Alpha3))
    {
        isButton3Pressed = true;
        button3.image.sprite = button3PressedSprite;
    }
    else if (Input.GetKeyUp(KeyCode.Alpha1))
    {
        if (isButton1Pressed)
            OnButton1Click();
        isButton1Pressed = false;
        button1.image.sprite = button1NormalSprite;
    }
    else if (Input.GetKeyUp(KeyCode.Alpha2))
    {
        if (isButton2Pressed)
            OnButton2Click();
        isButton2Pressed = false;
        button2.image.sprite = button2NormalSprite;
    }
    else if (Input.GetKeyUp(KeyCode.Alpha3))
    {
```

```
            if (isButton3Pressed)
                OnButton3Click();
            isButton3Pressed = false;
            button3.image.sprite = button3NormalSprite;
        }
    }

    private void OnButton1Click()
    {
        PlayButtonSound();
        if (p2HPBar != null)
        {
            p2HPBar.OnWrongButtonPressed();
        }
    }

    private void OnButton2Click()
    {
        PlayButtonSound();
        ask1.SetActive(false);
        ask2.SetActive(true);
        if (p1HPBar != null)
        {
            p1HPBar.OnWrongButtonPressed();
        }
    }

    private void OnButton3Click()
    {
        PlayButtonSound();
        if (p2HPBar != null)
        {
            p2HPBar.OnWrongButtonPressed();
        }
    }
    private void PlayButtonSound()
    {
        AudioSource.PlayClipAtPoint(buttonSound,
Camera.main.transform.position);
    }
}
```

ПРИЛОЖЕНИЕ Д

Листинг C# кода, диалоговая система

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class DialogSystem : MonoBehaviour
{

    public string[] lines;
    public float speedText;
    public Text dialogText;

    public int index;

    void Start()
    {
        index = 0;
        StartDialog();
    }

    void StartDialog()
    {
        dialogText.text = string.Empty;
        StartCoroutine(TypeLine());
    }

    IEnumerator TypeLine()
    {
        foreach (char c in lines[index].ToCharArray())
        {
            dialogText.text += c;
            yield return new WaitForSeconds(speedText);
        }
    }

    public void scipTextClick()
    {
        if (dialogText.text == lines[index])
        {
            NextLines();
```

```
        }
        else
        {
            StopAllCoroutines();
            dialogText.text = lines[index];
        }
    }

    public void NextLines()
    {
        if (index < lines.Length - 1)
        {
            index++;
            StartDialog();
        }
        else
        {
            index = 0;
            StartDialog();
        }
    }
}
```