Санкт-Петербургский государственный университет

***ХАНДЕШИНА Анастасия Андреевна***

**Выпускная квалификационная работа**

***Разработка обучающих приложений «Интерактивные задачи по математике»***

Уровень образования: бакалавриат

Направление 09.03.03 *«Прикладная информатика»*

Основная образовательная программа СВ.5078

*«Прикладная информатика в области искусств и гуманитарных наук»*

Профиль «Прикладная информатика в области искусств и гуманитарных наук»

Научный руководитель: доцент кафедры информационных систем в искусстве и гуманитарных науках, кандидат физико-математических наук

Щербаков Павел Петрович

Консультанты:

доцент кафедры информационных систем в искусстве и гуманитарных науках, кандидат педагогических наук

Гаевская Елена Георгиевна,

доцент кафедры информационных систем в искусстве и гуманитарных науках, кандидат физико-математических наук

Захаркина Валентина Валентиновна

Рецензент: профессор СПб ГУАП, доктор технических наук

Гордеев Александр Владимирович

Санкт-Петербург

2022

# **Аннотация**

выпускной квалификационной работы

Хандешиной Анастасии Андреевны

**«Разработка обучающих приложений «Интерактивные задачи по математике»»**

Целью выпускной квалификационной работы является создание многоплотформенного образовательного приложения «Интерактивные задачи по математике».

ВКР состоит из 3 глав, каждая из которых имеет 3 подглавы. В 1 главе рассказывается о исследовании педагогических приёмов и подборе математических задач. Во 2 главе описаны разработка структуры приложения и проработка внешнего вида приложения. В 3 главе освещен процесс реализации прототипов приложения.

Прикладным результатом ВКР является две версии образовательного приложения с игровыми элементами. Подбор математических задач, создание графических элементов и написание программного кода выполнены автором ВКР.

В процессе работы использованы программы diagrams.net, Aseprite, Unity и Visual Studio.

Объем работы: 71 страниц текста, 35 рисунков, 8 источников литературы и 1 приложение.

Ключевые слова: мобильное приложение, таксономия Блума, Unity, начальная школа, задачи по математике, пиксель-арт, компьютерная игра.

Автор работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Хандешина Анастасия Андреевна

подпись (фамилия, имя, отчество)

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_ Щербаков Павел Петрович

подпись (фамилия, имя, отчество)

# **Оглавление**

[**Аннотация** 2](#_Toc104330925)

[**Оглавление** 3](#_Toc104330926)

[**Определения** 4](#_Toc104330927)

[**Введение** 5](#_Toc104330928)

[**I. Педагогические приёмы и подбор задач** 6](#_Toc104330929)

[1.1 Таксономия Бенджамина Блума 6](#_Toc104330930)

[1.2 Эдьютеймент 7](#_Toc104330931)

[1.3 Математические задачи 8](#_Toc104330932)

[**II. Разработка структуры и внешнего вида приложения** 13](#_Toc104330933)

[2.1 Аналоги 13](#_Toc104330934)

[2.2 Интерфейс 17](#_Toc104330935)

[2.3 Визуальные решения и аудиодизайн 19](#_Toc104330936)

[**III. Реализация прототипа приложения** 21](#_Toc104330937)

[3.1 Aseprite 21](#_Toc104330938)

[3.2 Unity 22](#_Toc104330939)

[3.3 Тестирование 43](#_Toc104330940)

[**Заключение** 45](#_Toc104330941)

[**Использованные источники** 46](#_Toc104330942)

[**Приложения** 47](#_Toc104330943)

# **Определения**

UI (англ. - User Interface) - пользовательский интерфейс, дизайн кнопок, полей ввода текста;

User Flow (aнгл.) – схема передвижения пользователя между экранами приложения;

Ассет (англ. - asset) – внешний загружаемый ресурс, импортированный в проект и используемый в его реализации;

Игровая сцена (англ. – scene) – крупнейшая единица разделения проекта, отделяющая игровые объекты друг от друга;

Игровой движок (англ. - game engine) – способ, содержащий в себе инструменты для создания и реализации игры на определённой платформе;

Кастомизация (англ. - to customize) - изменение игроком вешнего вида персонажа или локации;

Компонент (англ. – component) – характеристика поведения объекта в игре;

Маскот (англ. - mascot) – узнаваемый персонаж, не участвующий в процессе игры, но являющийся её олицетворением и осуществляющий взаимодействие с пользователей через прямую речь;

Писксель-арт (англ. – pixel art) – вид цифрового изображения с четко выделенными отдельными пикселями;

Рендер (англ. - render) - процесс получения изображения или видеозаписи с помощью компьютерной программы;

Скрипт (англ. – script) – программный код, описывающий поведение игровых объектов;

Таксономия (др.-греч. τάξις — строй, порядок и νόμος — закон) - теория систематизации сложноорганизованных областей действительности и знания, имеющих иерархическое строение;

Эдьютейнмент (англ. – edutainment) - бучение в формате развлечений;

# **Введение**

Обучение детей представляет из себя сложный и трудоёмкий процесс, требующий полного вовлечения не только педагога, но и ученика. Для повышения уровня заинтересованности ребенка в учебном процессе используются различные игровые механики. Не смотря на противоречивое отношение к обучению через развлечение теоретиков педагогики, данная практика получает всё большее распространение, особенно после компьютеризации общества.

Именно с помощью цифровых технологий реализована выпускная квалификационная работа - обучающие приложения «Интерактивные задачи по математике». Приложения для персонального компьютера на операционной системе Windows и мобильных устройств на операционной системе Android являются синтезом педагогических наук, игровых механик и цифровой реализации.

Целевой аудиторией приложений являются дети младшего школьного возраста, в повседневной жизни которых сочетаются необходимость в обучении и потребность в игре. Стоит учитывать и возросшую значимость электронных устройств в воспитании и досуге детей, что делает цифровые приложение актуальным способом обучения и развлечения.

Целью выпускной квалификационной работы является создание многоплотформенного образовательного приложения «Интерактивные задачи по математике». Для её достижения были поставлены следующие задачи:

* изучение педагогических приёмов подбор подходящих математических задач;
* изучение аналогов;
* разработка структуры и аудиовизуального стиля приложения;
* реализация прототипов приложений.

# **I. Педагогические приёмы и подбор задач**

В педагогике используются разнообразные приемы и техники, направленные на улучшение качества образования. Для того, чтобы повысить полезность приложений, а результат их использования максимально эффективным, необходимо изучить педагогические аспекты донесения информации до ребёнка и ее закрепления.

# **1.1 Таксономия Бенджамина Блума**

Бенджамин Блум – американский ученый, руководитель группы, разработавшей схему педагогических целей – таксономию Блума. Рассмотрим выделяемые области деятельности и их специфику:

* Когнитивная (познавательная) область отвечает за изучение и последующее применение материала для решения проблем, в ходе которого происходит комбинирование имеющихся знаний и навыков.
* Аффективная (эмоционально-ценностная) область включает в себя чувства и эмоции, с помощью которых формируется эмоционально-личностное отношение к окружению, интересы и ценности.
* Психомоторная область связана с физической деятельностью и развитием практических навыков устной и письменной речи, а также использованию дополнительных инструментов. Исследования Блума затрагивают в основном когнитивную область, выделяя в ней иерархически выстроенные учебные цели. Перейдем непосредственно к иллюстрации таксономии.

Рис. 1 – Таксономия Блума

Основой иерархии является знание. Оно включает в себя изучение, запоминание и воспроизведение информации. На этом этапе формируется знания об основных понятиях, теориях, правилах, терминах, фактах и принципах; ученик может назвать, перечислить, определить, найти, указать, записать и повторить необходимое. Далее идет понимание, характеризующиеся способностью ученика к интерпретации полученных знаний. Например, к понимаю относится пересказ, перефразирование, перевод, чтение графиков и схем, обобщение и сравнение. Следующей категорией является применение. На этом этапе ученик применяет полученные знания в практических ситуациях, в том числе новых для себя. Так он демонстрирует высокий уровень владения информацией и способность ее применения все зависимости от условий. Выше в таксономии находится анализ, включающий в себя возможность структурирования материала, с его последующим разделением на составные части. На этом этапе ученик способен к логическим суждениям по имеющимся данным, поиску ошибок и взаимосвязей. Следующей ступенью является синтез, ознаменованный умением комбинировать полученные знания и навыки для создания новых концептов. Вершиной иерархии считается оценивание. Для достижения этой учебной цели ученику необходимо уметь давать оценку материалу, судить о нем по критериям и аргументировать свое мнение.

С помощью таксономию Блума можно определить уровни учебных целей, которые необходимо пройти для полного усвоения информации. Таким образом, при составлении порядка задач необходимо учитывать отношение каждой конкретной задачи к тому или иному уровню и располагать их от основания иерархии до ее вершины.

# **Эдьютеймент**

Так как обучающие приложения подразумевают под собой сочетание образовательных и развлекательных элементов, рассмотрим понятие, характеризующее симбиоз этих направлений. В зарубежной педагогике распространен термин эдьютеймент – от английских слов education (обучение) и entertainment (развлечение). Кандидат философских наук, доктор социологических наук, профессор кафедры философской антропологии и общественных коммуникаций РГПУ им. А.И. Герцена Ольга Леонидовна Гнатюк дает следующее определение: «Эдьютеймент - цифровой контент, соединяющий образовательные и развлекательные элементы». Ольга Олеговна Дьяконова, автор статьи «Понятие «Эдьютеймент» в зарубежной и отечественной педагогике», выводит следующие критерии успешной реализации эдьютеймента:

1) необходимо побудить учащихся обратить внимание на изучаемое явление, вызвать их интерес, побудить принять активное участие в процессе приобретения знаний;

2) во время самого процесса приобретения знаний, необходимо доставить учащимся удовольствие от полностью занять их и отвлечь от сторонних мыслей или переживаний;

3) необходимо сильно заинтересовать учащихся, заставит их целиком   
отдаться занятию или идее.

Теперь следует определить, как каждый из пунктов может реализоваться в обучающих приложениях «Интерактивные задачи по математике». Первый пункт достижим с помощью музыкального сопровождения, звуковых эффектов и яркого дизайна, используемого при создании главного меню, меню выбора игровых уровней, самого игрового уровня, внутриигрового магазина и дизайна главного персонажа. Второй пункт реализуем с помощью системы поощрений за пройденный уровень в виде внутиигровой валюты, которая может быть потрачена на кастомизацию главного персонажа. Третий пункт строится на желании игрока пройти все уровни, чтобы получить максимальное количество валюты для покупки каждого понравившегося элемента в магазине.

# **1.3 Математические задачи**

**Каждому уровню в обучающих приложениях соответствует одно задание, выбранное из учебников по математике издательства «Просвещение» соответствующего года обучения. Исходя из количества учебных целей в таксономии Блума и разделов математики, изучаемых в школах, следует, что всего количество уровней составляет 48, по 12 для каждого года обучения, 6 из которых соответствует учебной программе по алгебре и 6 по геометрии. Для упрощения подбора задач, следует опираться на глаголы, соответствующие каждому из уровней таксономии Блума.**

**Задачи из программы 1 класса:**

* Задачи на знание (вспомнить, распознать, идентифицировать, добыть):

1) Какими знаками обозначаются сложение, вычитание и равенство?

2) Напиши номера картинок, на которых нарисованы луч и отрезок. (задача с иллюстрацией)

* Задачи на понимание (интерпретировать, реорганизовывать, перекомпоновать, дифференцировать, достичь, сделать, установить, объяснить):

1) Запиши задачу с помощью математических знаков: Сначала был 1 ёжик. Потом к нему подбежали ещё 4 ёжика. Всего стало 5 ёжиков.

2) Мысленно поменяй местами каждую пару фигур одинаковой формы и запиши, какое слово получилось. (задача с иллюстрацией)

* Задачи на применение (применить, обобщить, установить связи, выбрать, развить, организовать, использовать, руководствоваться, преобразовать):

1) Какие математические знаки надо подставить в пример 7 \_ 3 \_ 4?

2) Как можно одним словом назвать все фигуры на рисунке? (задача с иллюстрацией)

* Задачи на анализ (анализировать, констатировать, сравнивать, выводить):

1) Найди ошибку: а) 5 < 8 б) 3 – 2 = 1 в) 5 + 2 > 8 г) 2 + 4 = 6. В ответ выпиши букву неправильного примера.

2) Сравни длины отрезков и запиши длину самого короткого. (задача с иллюстрацией)

* Задачи на синтез (производить, развивать, комбинировать, организовать, сформулировать, модифицировать):

1) Составь и реши пример по тексту: В коробке было 7 карандашей. Вынули 2 карандаша. Сколько карандашей осталось в коробке? Запиши само выражение и ответ к нему с помощью математических знаков.

2) Какого цвета листочек надо нарисовать в свободной клетке, чтобы все строки и столбцы были разными? (задача с иллюстрацией)

* Задачи на оценивание (рассуждать, аргументировать, оценивать, решать):

1) Обсуди с родителями или друзьями, как ты используешь цифры в своей жизни. Запиши свои выводы.

2) Обсуди с родителями или друзьями, как ты используешь меры длины в своей жизни. Запиши свои выводы.

**Задачи из программы 2 класса:**

* Задачи на знание (вспомнить, распознать, идентифицировать, добыть):

1) Какими знаками обозначаются умножение и деление?

2) Впиши недостающие слова в определение периметра многоугольника. Периметр многоугольника – это \_ длин всех его \_.

* Задачи на понимание (интерпретировать, реорганизовывать, перекомпоновать, дифференцировать, достичь, сделать, установить, объяснить):

1) Реши пример: 2 \* 7 + 2 \* 3

2) Найди периметр этого треугольника. (задача с иллюстрацией)

* Задачи на применение (применить, обобщить, установить связи, выбрать, развить, организовать, использовать, руководствоваться, преобразовать):

1) Выбери вариант, в котором все числа делятся на 3: а) 8, 3, 14 б) 13, 23, 33 в) 20, 60,10 г) 21, 39, 15

2) Сколько острых, прямых и тупых углов на чертеже? (задача с иллюстрацией)

* Задачи на анализ (анализировать, констатировать, сравнивать, выводить):

1) Сравни произведение чисел 7 и 3 и разность чисел 20 и 1. Запиши нужный символ.

2) Какая фигура лишняя? Запиши ее номер. (задача с иллюстрацией)

* Задачи на синтез (производить, развивать, комбинировать, организовать, сформулировать, модифицировать):

1) Составь и реши математическое выражение по тексту: За партами сидели 18 учеников, по 2 ученика за каждой партой. Сколько парт заняли эти ученики? Запиши само выражение и ответ к нему с помощью математических знаков.

2) Запиши номера тупых углов в этих треугольниках. (задача с иллюстрацией)

* Задачи на оценивание (рассуждать, аргументировать, оценивать, решать):

1) Обсуди с родителями или друзьями, как ты используешь деление в своей жизни. Запиши свои выводы.

2) Обсуди с родителями или друзьями, как ты используешь знание о площади фигур в своей жизни. Запиши свои выводы.

**Задачи из программы 3 класса:**

* Задачи на знание (вспомнить, распознать, идентифицировать, добыть):

1) Запиши пропущенное слово: Неизвестное в уравнении называют \_ уравнения.

2) Какие существуют единицы измерения длины? Запиши ответ через запятую начиная с наименьшей.

* Задачи на понимание (интерпретировать, реорганизовывать, перекомпоновать, дифференцировать, достичь, сделать, установить, объяснить):

1) Преобразуй 1 час 39 минут в минуты.

2) Сколько квадратных сантиметров в 5 дм² ?

* Задачи на применение (применить, обобщить, установить связи, выбрать, развить, организовать, использовать, руководствоваться, преобразовать):

1) Укажи частное при делении 72 на 12.

2) Какие из треугольников равнобедренные? (задача с иллюстрацией)

* Задачи на анализ (анализировать, констатировать, сравнивать, выводить):

1) Прочитай текст и определи, на сколько самое глубокое озеро глубже самого мелкого. Озера имеют различную глубину. Глубина Ладожского озера – двести тридцать метров, Онежского озера – сто двадцать семь метров, а озера Иссык-Куль – шестьсот шестьдесят восемь метров.

2) Сколько треугольников на картинке? (задача с иллюстрацией)

* Задачи на синтез (производить, развивать, комбинировать, организовать, сформулировать, модифицировать):

1) Верно ли утверждение, что в числе 601 содержится 6 сотен и 1 десяток?

2) Аня, Денис и Коля начертили по одной фигуре: Аня и Денис начертили фигуры с одинаковым числом сторон, а Коля и Денис начертили фигуры с одинаковым периметром. В ответ запиши числа в порядке, соответствующем фигурам Ани, Дениса и Коли. (задача с иллюстрацией)

* Задачи на оценивание (рассуждать, аргументировать, оценивать, решать):

1) Обсуди с родителями или друзьями, в какой ситуации в жизни тебе понадобится умение делить с остатком. Запиши свои выводы.

2) Обсуди с родителями или друзьями, где в повседневном окружении можно встретить прямоугольный треугольник. Запиши свои выводы

**Задачи из программы 4 класса:**

* Задачи на знание (вспомнить, распознать, идентифицировать, добыть):

1) Какая величина измеряется в км/ч?

2) Дополни определение: Тупой угол – угол больше \_ градусов.

* Задачи на понимание (интерпретировать, реорганизовывать, перекомпоновать, дифференцировать, достичь, сделать, установить, объяснить):

1) Вычисли: 34 \* (120 – 3920 : 70)

2) Какой угол является прямым? (задача с иллюстрацией)

* Задачи на применение (применить, обобщить, установить связи, выбрать, развить, организовать, использовать, руководствоваться, преобразовать):

1) Раскрой скобки 8 \* (345 – 49) и запиши полученное выражение.

2) Какие из высказываний являются верными? (задача с иллюстрацией)

1) Если фигура не жёлтого цвета, то это многоугольник.

2) Если фигура синего цвета, то это четырехугольник.

3) Если фигура не закрашена, то это прямоугольный треугольник.

* Задачи на анализ (анализировать, констатировать, сравнивать, выводить):

1) Какое из выражений решено правильно? а) 70537 : 54 = 1306 (ост 17) б) 33367 : 164 = 203 (ост 75) в) 155364 : 604 = 257 (ост 135)

2) Сколько частей каждого круга закрашено? Запиши ответ используя знак /. (задача с иллюстрацией)

* Задачи на синтез (производить, развивать, комбинировать, организовать, сформулировать, модифицировать):

1) Площадь кухни 9 м^2, что составляет одну восьмую часть площади всей квартиры. Найди площадь квартиры.

2) Под углом в сколько градусов пересекаются диагонали квадрата?

* Задачи на оценивание (рассуждать, аргументировать, оценивать, решать):

1) Обсуди с родителями или друзьями что можно измерить в миллиардах. Запиши свои выводы.

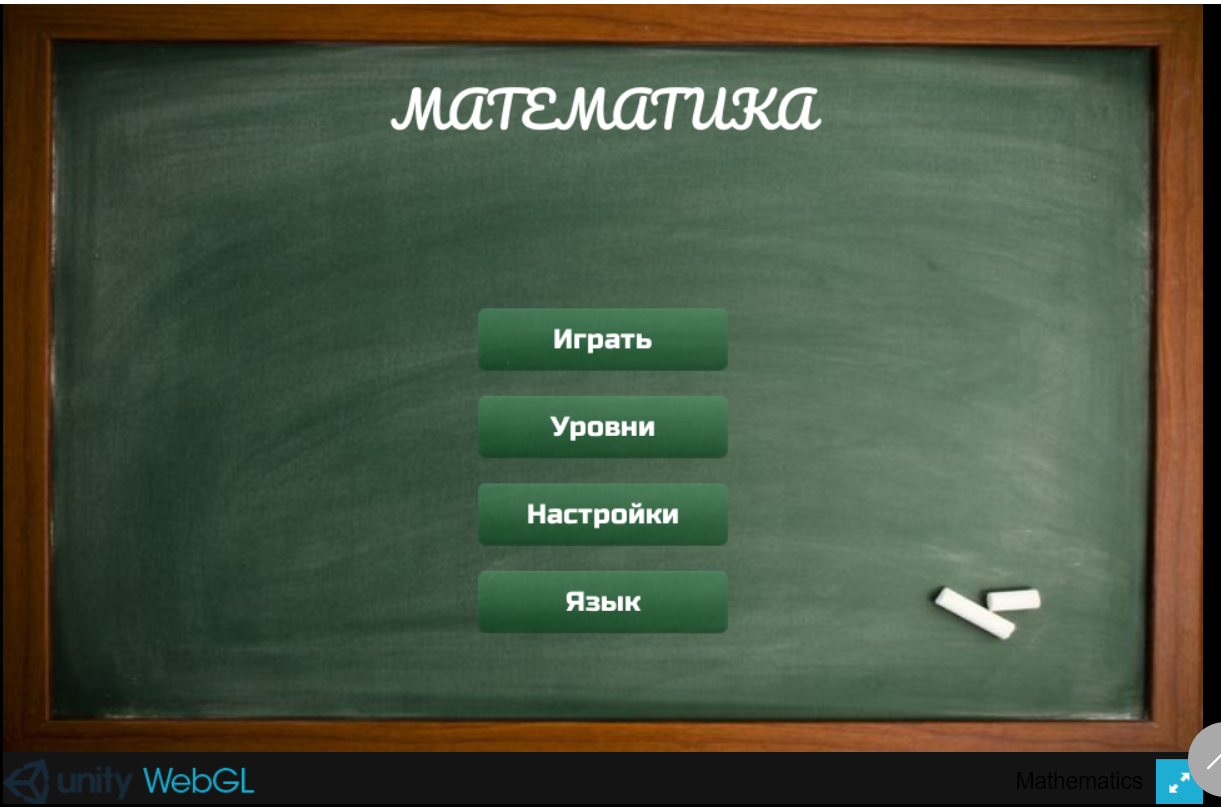
2) Обсуди с родителями или друзьями, где в повседневной жизни могут использоваться знания о площади. Запиши свои выводы.

# **II. Разработка структуры и внешнего вида приложения**

# **2.1 Аналоги**

Индустрия разработки развлекательного программного обеспечения достаточно молода, и в ней нет устоявшихся стандартов по реализации функционала приложений и его внешнего вида. Поэтому для создания приложений необходимо принимать во внимание уже созданные продукты и выделять наиболее удачные и неудачные идеи. Рассмотрим несколько программ с похожим функционалом.

Программа «Игра По математике 1 класс» свободно распространяется в сети Интернет, следовательно, её может приобрести любой желающий без финансовых вложений. Игра разработана на игровом движке Unity. На каждом из меню приложения воспроизводится приятная мелодия.

Рис. 2 – Главное меню программы «Игра По математике 1 класс»

Главный экран приложения стилизован под школьную доску, что соответствует обучающей тематике, однако является непривлекательным для большинства учеников.

Рис. 3 – Меню уровней программы «Игра По математике 1 класс»

Меню уровней представляет собой квадраты с обозначением номера уровня. Заблокированные уровни, для разблокировки которых необходимо пройти предыдущие, изображаются замками, что метафорично отсылает к их недоступности.

Рис. 4 – Меню уровня программы «Игра По математике 1 класс»

Каждый уровень представлен одним математическим примером. Ответ вводится с помощью клавиатуры пользователя, после нажимается кнопкам подтверждения ответа. Если ответ правильный, на экран выводится текст «Отлично», в обратном случае – «Неправильно. Попробуй ещё раз».

Резюмируя краткий обзор приложения, можно выделить следующие удачные идеи:

* Разделение главного меню приложения на разделы «Играть», «Уровни» и «Настройки»
* Условное обозначение замка для недоступных уровней
* Соответствие одной задачи одному уровню игры
* Наличие музыкального сопровождения

К неудачным можно отнести:

* Задачи включают в себя только вычисление правильного ответа
* Общая невыразительность интерфейса приложения и использование серой и болотно-зеленой цветовой гаммы
* Недостаточная мотивация пользователя из-за отсутствия награды

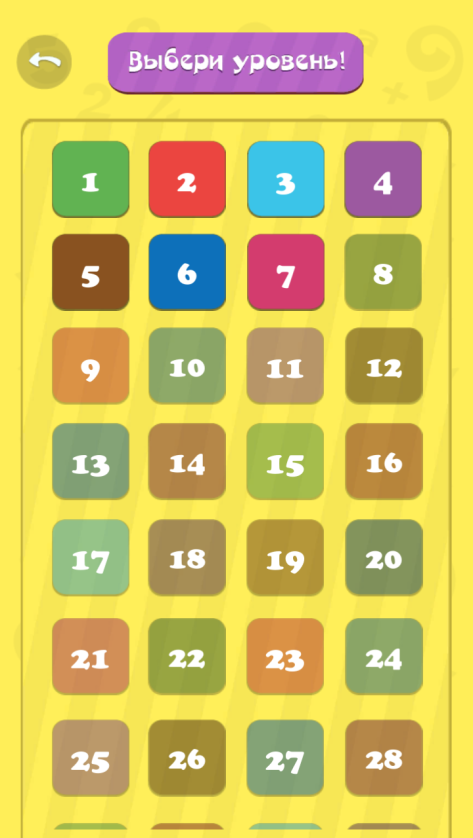
Приложение «100 Задачек по математике» свободно распространяется в цифровом магазине Google Play. Игра разработана на игровом движке Unity.

Рис. 5 – Главное меню приложения Рис. 6 – Меню уровней приложения Рис. 7 – Меню уровня приложения

«100 Задачек по математике» «100 Задачек по математике» «100 Задачек по математике»

Каждое меню приложения яркое и цветное, что способствует привлечению внимания ребёнка. В меню уровней также присутствуют отличия в обозначениях доступных для прохождения уровней. Однако они выражены лишь в незначительной насыщенности цвета. Ответ на каждую задачу водится с помощью кнопок, соответствующих варианту ответа.

Рис. 8 – Введён Рис. 9 – Введён неверны й

верный ответ ответ

Ввод правильного и неправильного ответа сопровождается появлением всплывающего элемента в виде звезды с кнопками выхода в главное меню и перехода к следующему уровню, или кнопка повтора введения ответа.

Резюмируя краткий обзор приложения, можно выделить следующие удачные идеи:

* Яркий дизайн
* Иллюстрации к текстам задач
* Всплывающие элементы для правильного и неправильного ответа

К неудачным можно отнести:

* Задачи включают в себя только вычисление правильного ответа
* Недостаточная мотивация пользователя из-за отсутствия награды

После обзора нескольких аналогичных приложений можно сделать вывод о векторе развития разработки обучающих приложений «Интерактивные задачи по математике» на основе удачных и неудачных идей.

# **2.2 Интерфейс**

Техническая реализация сразу на двух платформах позволяет пользователям самим выбрать удобный формат взаимодействия с приложением: версия для персонального компьютера под операционной системой Windows обладает более широким разрешением, что позволит в процессе разработки разместить больше элементов на экране. Версией для устройств с операционной системой Android можно пользоваться в любом удобном для пользователя месте ввиду специфики мобильных устройств. Название «Интерактивные задачи по математике» отражает контент, содержащийся в приложениях. Интерактивность заключается в прямом взаимодействии пользователя с задачами в игровой форме с помощью устоявшихся игровых механик и графического интерфейса.

Поскольку приложение рассчитано на учеников с 1 по 4 классы, то появляется необходимость в разделение задач по годам обучения. Таким образом, оправдана реализация 4 блока уровней. Каждый блок содержит 12 задач, а, следовательно, 12 уровней. За выполнение каждого задания игрок получает награду – внутриигровую валюту, стилизованную под монету для прямой ассоциации с возможностью их потратить во внутриигровом магазине.

В качестве маскота выбран кот из-за положительных ассоциаций с домашним питомцем, который есть во многих семьях. С помощью диалогового окна для стилизации прямой речи персонажа можно реализовать инструкции по взаимодействию с приложениями и тексты заданий. Следовательно, формулировка инструкций и текстов заданий содержит обращение на «ты» к пользователю.

В магазине можно приобрести способы кастомизации персонажа, который будет присутствовать на многих экранах приложения. Чем значимее изменение во внешности, тем дороже стоит изменение.

Также для мотивирования игрока проходить уровни введена система внутриигровых достижений, которые отображены в виде кубков на экране выбора блока уровней. Игрок будет награжден по прохождению каждого блока задач.

Из анализа других приложений и общей идеи приложения следует:

Список необходимых экранов:

* Главное меню;
* Меню выбора уровня;
* Магазин;
* Настройки;
* Руководство пользователя;
* Меню уровня;
* Экран правильного ответа;
* Экран неправильного ответа;
* Экран оповещения о порождении всех уровней блока;

Главное меню должно содержать кнопки перехода на последний запущенный уровень, перехода на меню уровней, перехода на страницу магазина, меню настроек и кнопку для выхода из приложения. Страница запущенного уровня должна содержать сам уровень. Меню уровней должно содержать кнопки перехода к конкретному уровню и выхода в главное меню. Страница магазина должна содержать кнопки для покупки лотов и кнопку выхода в главное меню. Меню настроек должно содержать инструмент для регулировки громкости звука и кнопку выхода в главное меню.

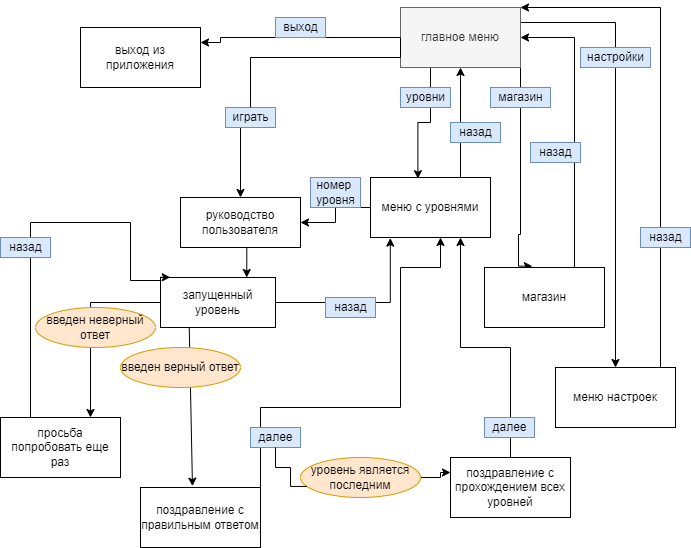
Для лучшего понимания взаимодействия пользователя с приложением необходимо построить схему User Flow, опыт создания которого получен на учебной дисциплине «Проектирование информационных систем» на 3 году обучения. 

Рис. 10 – схема User Flow обучающего приложения

На схеме белыми прямоугольниками обозначаются страница приложения, оранжевыми эллипсами – условия перехода на другую страницу, а синими прямоугольниками – кнопки на экране, ведущие к смене страницы. Стрелки являются условным обозначением перехода на другую страницу.

# **2.3 Визуальные решения и аудиодизайн**

В качестве визуального решения для приложения выбран яркий и контрастный стиль, захватывающий внимание пользователя и упрощающий считывание информации с экрана устройства. Для реализации элементов UI выбран стиль писксель-арт. Причинами этого послужили его распространенность в других медиа-продуктах и внешний вид, подходящий для детей младшего школьного возраста.

Поскольку в приложениях используется текст для отображения названий страниц приложений, текста заданий и указаний игроку, то возникает потребность в его визуализации. В связи с этим была поставлена цель найти подходящий шрифт, отвечающий перечисленным ниже критериям:

1) поддержка кириллических буквенных символов

2) визуальная стилизация под пискель-арт

3) свободное некоммерческое использование

4) поддержка игровым движком Unity

После продолжительных поисков был найден шрифт, отвечающий всем критериям. Им стал Sevenet 7 Cyr, правообладателями которого являются Sevenet & Dubina Nikolay. Шрифт распространяется свободно и подходит для некоммерческого использования.

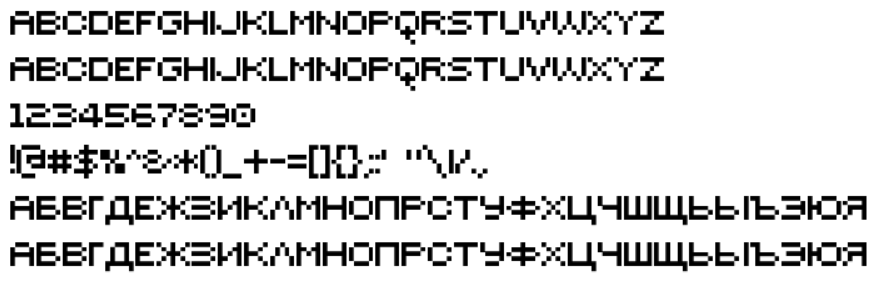


Рис. 11 – шрифт Sevenet 7 Cyr

Для реализации музыкального сопровождения было решено воспользоваться услугами стороннего композитора. Для него было написано техническое задание следующего содержания:

Требуется создание двух музыкальных композиций. Первая предназначается для главного меню, меню выбора уровня и магазина. Музыкально она должна быть более весёлой и быстрой (мажорной). Это необходимо, поскольку эти экраны – первые, и с них начинается взаимодействие пользователя с приложением. Музыкальная тема должна зацепить его и заставить остаться в приложении для дальнейшего использования. Вторая композиция будет использована во время взаимодействия с уровнями и непосредственным решением задач. Следовательно, она должна быть более спокойной и приглушенной (минорной), чтобы не отвлекать пользователя от мыслительного процесса. Окончание обеих композиций должно быть одинаковым с их началом, для возможности реализации зацикливания их воспроизведения. Стиль композиций должен соответствовать детской направленности приложения.

Композитор использовал программу Ableton Live и включенные в неё инструменты для создания музыкальных файлов формата mp3. Длина мажорной композиции составила 2 минуты 14 секунд, а размер файл – 5,12 Мб. Длина минорной композиции составила 2 минуты 47 секунд, размер файла – 6,56 Мб.

# **III. Реализация прототипа приложения**

# **3.1 Aseprite**

Для реализации графических элементов приложения была использована программа Aseprite, разработанная специально для создания пиксель-арта. Программа распространяется платно в цифровом магазине Steam и на официальном сайте разработчика.

С помощью инструментов «карандаш» (Pencil Tool), «заливка» (Paint Bucket Tool), «четырёхугольник» (Rectangle Tool) и «эллипс» (Ellipse Tool) были реализованы все графические элементы. Для упрощения создания некоторых элементов (элементы UI и иллюстрации к задачам) была использована функция симметричного отображения (Symmetry Options), доступная как для горизонтальной плоскости, так и для вертикальной.

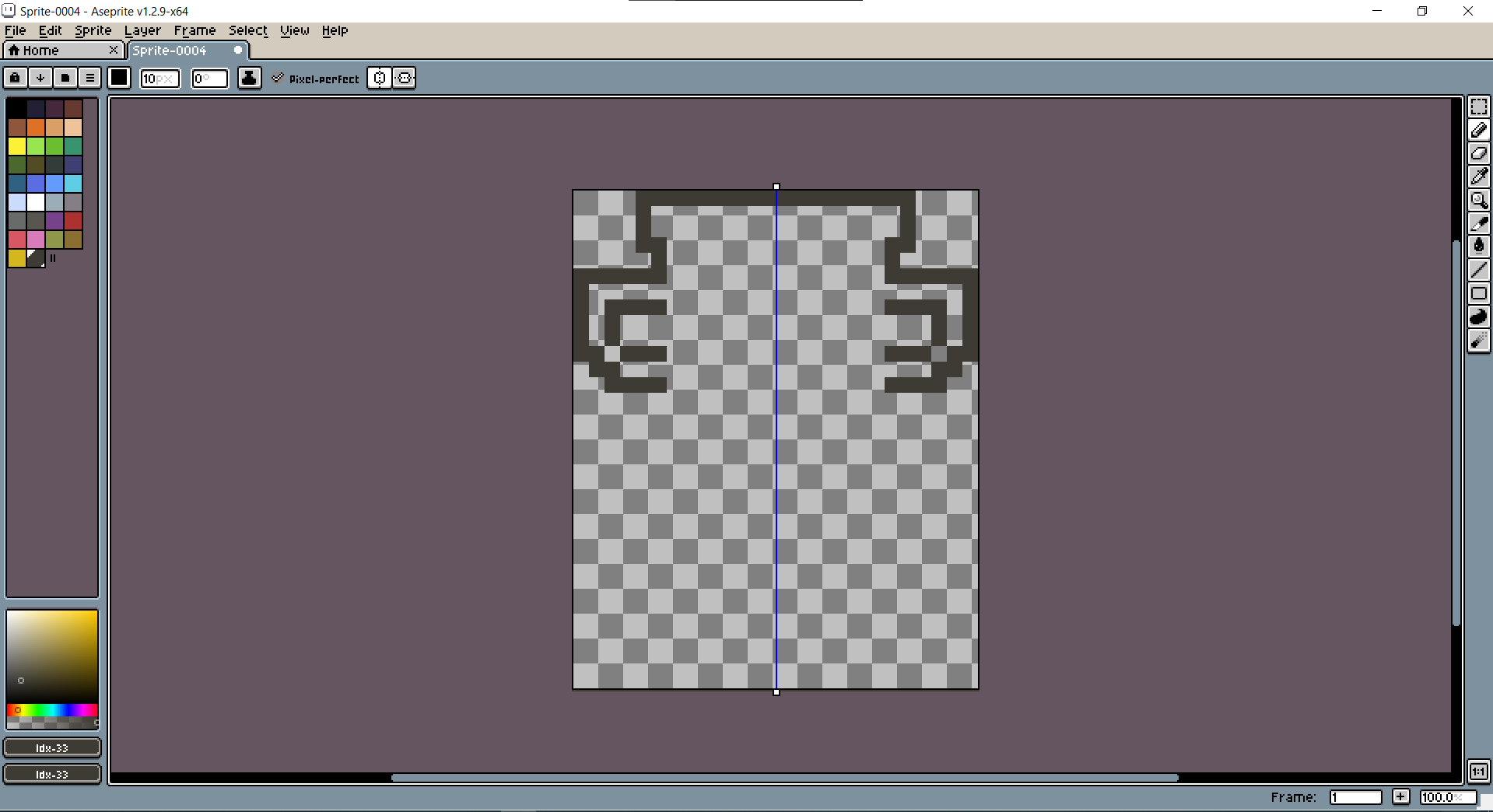


Рис. 11 – создание призового кубка в Aseprite с помощью инструмента «карандаш» и функции симметричного отображения

Итогом работы в программе стало создание графических элементов:

* элементы UI;
* призовой кубок и монета;
* персонаж – кот и вариации его внешнего вида;
* визуализации к задачам с геометрическим уклоном и иллюстрации к текстам заданий.

Все файлы сохранены в формате png, а их размер варьируется от 1 до 12 Кб.



Рис. 12 – некоторые из созданных в программе Aseprite графических элементов

# **3.2 Unity**

Для реализации обучающих приложений был выбран игровой движок Unity, освоенный в рамках учебного курса «Разработка систем виртуальной реальности» на втором году обучения. Основными преимуществами Unity, которые повлияли на выбор данной среды разработки, являются возможность работы сразу с несколькими платформами (Windows, iMac, Android, iOS, VR и т.д.), простота и понятность интерфейса, модульная система компонентов и свободное распространение для обучения и создания игр, прибыль которых составляет меньше 100 000 долларов США в год. Модульная система компонентов позволяет создавать игровые объекты, к которым можно добавить функциональные элементы, отвечающие за внешний вид объекта, его характеристики, поведение на игровой сцене и т.д.

Для разработки была использована версия 2020.1.4, на которой были реализованы предыдущие проекты, созданные в рамках учебного процесса.

Написание скриптов в Unity происходит на языке C#. С# - это объектно-ориентированный язык программирования, разработанный на рубеже XX и XXI веков американской компанией Microsoft. Данный язык относится к C-подобным языкам и наследует многое от своего предшественника, языка Java, освоенного в рамках учебных курсов «Технологии программирования» на втором году обучения.

Для написания программного кода была использована среда разработки Microsoft Visual Studio версии 17.1.5, поддерживающая несколько языков, в том числе и C#. Данная среда разработки была выбрана из-за понятного и простого интерфейса и наличия бесплатной лицензии для индивидуального использования и полного пакета инструментов для реализации программного кода на C#.

Для реализации обучающих приложений «Интерактивные задачи по математике» было создано 2 Unity-проекта в режиме 2D, так как на текущих стадиях разработки не предполагается наличие трёхмерных элементов.

В каждом проекте реализовано по 5 игровых сцен:

1. Главное меню (Menu)
2. 1 Класс (Class1)
3. 2 Класс (Class2)
4. 3 Класс (Class3)
5. 4 Класс (Class4)

Главное меню

Каждая игровая сцена состоит из нескольких объектов, служащих различным целям. Например, определение положения элементов UI, их отображение и обработка событий. Разберем каждый из существующих объектов в данной сцене.

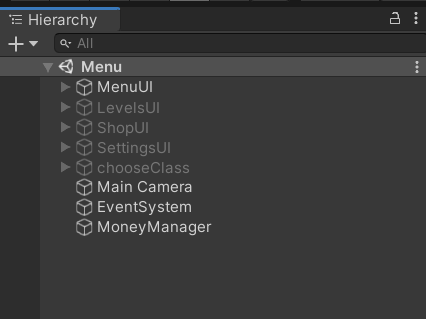


Рис. 13 – иерархия объектов в игровой сцене Menu

Как упоминалось ранее, в Unity разработке применяется модульная система компонентов. Каждая сцена состоит из нескольких игровых объектов (GameObject) с компонентом «Холст» (Canvas), являющийся пространством, в котором расположены все объекты UI.

Компоненты отображаются в панели «Инспектор» (Inspector). Каждому игровому объекту по умолчанию присваивается компонент «Трансформация» (Transform), который определяет положение игрового объекта в пространстве и его масштаб. Помимо этого, к игровому объекту добавить другие компоненты, отвечающие за отображение объекта на экране, его функции и поведение.

Рассмотрим компоненты холста MenuUI.

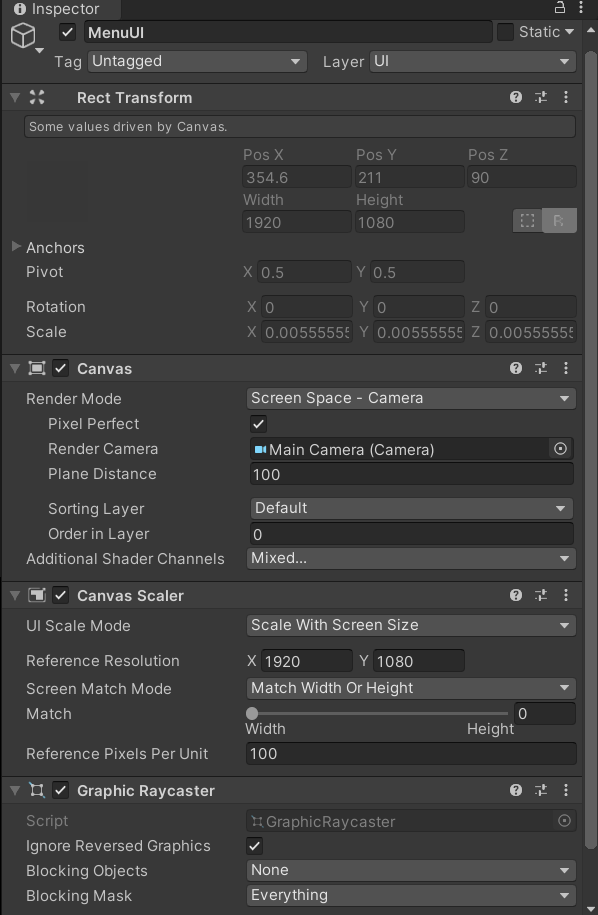


Рис. 14 – компоненты и их свойства холста MenuUI

Помимо компонента «Превращение» (в данном случае он называется «Превращение четырехугольника» (Rect Transform), поскольку проект выполнен в двухмерной плоскости), для игрового объекта используются ранее упомянутый компонент «Холст» (Canvas). Среди свойств этого компонента можно выделить «Способ рендера» (Render Mode), который отвечает за способ представления UI (в данном случае использует рендер через основную камеру (Main Camera), которая была создана автоматически при добавлении новой игровой сцены в проект), и «Порядок слоёв» (Order in Layer), отвечающий за порядок слоёв в игровой сцене (в данном случае значение составляет минимальное возможное - 0, поскольку главное меню является самым нижним слоем в игровой сцене).

Также используется компонент «Масштаб холста» (Canvas Scaler). Некоторые из его свойств: «Режим масштабирования UI» (UI Scale Mode) отвечает за способ отображения UI на холсте (в данном случае используется способ «Масштабирование по размеру экрана» (Scale With Screen Size), размер которого соотносится с популярным разрешением экрана Full HD (1920 x 1080 пикселей)).

Компонент «Графическое отображение» (Graphic Raycaster) используется автоматически для всех элементов холсте. Он определяет какие графические объекты есть холсте и какие из них используются для ввода.

Аналогичные компоненты используются и для холстов LevelsUI, ShopUI, SettingsUI и chooseClass. Однако стоит отметить, что в холсте chooseClass значение «Порядок слоёв» составляет 1, поскольку данный холст является всплывающим окном и располагается над всеми другими холстами.

Рассмотрим прочие игровые объекты в данной сцене: «Основная камера» (Main Camera), «Система событий» (EventSystem) и «Менеджер денег» (MoneyManager).

Игровой объект «Основная камера», как было ранее упомянуто, был создан автоматически при добавлении новой сцены. Этот объект отвечает за захват игрового поля и демонстрацию его пользователю приложения. В данном случае использованы стандартные настройки всех компонентов («Камера» (Camera), отвечающий за настройку параметров камеры и «Слушатель аудио» (AudioListener), захватывающий звук в игровой сцене), поскольку они полностью соответствуют задачам реализации и не нуждаются в дополнительных настройках.

«Система событий», аналогично «Основной камере», добавляется на игровую сцену автоматически. Тут также сохранены стандартные значения компонентов «Система событий» (Event System) и «Модуль ввода» (Standalone Input Module), однако был добавлен компонент «Источник аудио» (Audio Source). В свойствах данного компонента стоит отменить «Аудио дорожку» (Audio Clip), которая отвечает за конкретный мультимедийный файл, проигрываемый на данной сцене, и «Цикличность» (Loop), отвечающей за зацикленность аудио файла. Таким образом, в данном примере «Аудио дорожкой» является музыкальный файл «figure i.mp3, написанный специально для обучающих приложений согласно техническому заданию, составленному ранее, а галочка напротив «Цикличности» означает, что файл будет воспроизведён заново после окончания.

Игровой объект «Менеджер денег» является контейнером для хранения скрипта Money Manager, отвечающего за обработку денежных средств на холсте ShopUI.

Холст MenuUI отвечает за отображение элементов пользовательского интерфейса на главном меню.

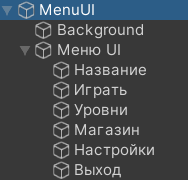


Рис. 15 – иерархия объектов на холсте MenuUI



Рис. 16 – MenuUI

Игровой объект Background является элементом UI типа «Изображения» (Image), растянутым с помощью компонента «Превращение четырехугольника» на всю площадь холста. В компоненте «Изображение» (Image) «Источником Изображения» (Source Image) является файл «bg1.png», отвечающий за фон данного холста. Следующим игровым объектом является Menu UI - контейнер для хранения скрипта Main Menu, отвечающего за отображение холста chooseClass. Свойства компонента будут рассмотрены при обсуждении скрипта MainMenu. Одним из потомков Меню UI является «Название». Это игровой объект типа «Текст» (Text), используемый для отображения текстовых элементов UI. Обратим внимание на компонент «Текст» (Text), где представлены свойства отображения текста. Среди них содержание текстового поля (text), шрифт (font), способ начертания (font style), размер шрифта (font size), выравнивание (alignment), выровнять по геометрии текстового поля (align by geometry) и лучший размер шрифта для данного текстового поля (best fit).

Далее следуют объекты Играть, Уровни, Магазин, Настройки и Выход. Все они принадлежат типу «Кнопка» (Button) и отвечают за обработку взаимодействия пользователя с элементами UI.

Некоторыми компонентами кнопки Играть являются «Изображение» (Image), «Кнопка» (Button), «Обработка нажатия» (On Click ()) и контейнер для скрипта Main Menu. «Изображение» отвечает за визуальную часть кнопки, одним из свойств которого является «Источник изображения» (Source Image). В данном случае «Источником изображения» служит файл «играть.png». В «Обработке нажатия» указано, что при нажатии на кнопку запустится функция Play из скрипта MainMenu, который, в свою очередь, находится в контейнере Main Menu.

Похожие компоненты представлены и для кнопки Уровни. Отличия заключается в «Источнике изображения» и «Обработке нажатий». В данном случае «Источником изображения» является «уровни.png», а при нажатии холст MenuUI отключается (GameObject.SetActive = false), а холст LevelsUI становится активным (GameObject.SetActive = true). Таким образом происходит смена отображаемых элементов UI на экране пользователя.

Аналогичная ситуация с объектами Магазин и Настройки, которые отвечают за переход из MenuUI в ShopUI и SettingsUI соответственно. Кнопка Выход отвечает за выполнение функции Exit из скрипта Main Menu.

Игровой объект LevelsUI отвечает за отображение элементов пользовательского интерфейса на меню выбора класса.

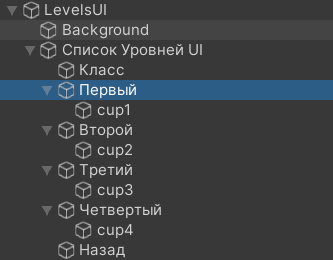


Рис. 17 – иерархия объектов на холсте LevelsUI



Рис. 18 – LevelsUI

Объект Background аналогичен рассмотренному ранее. Как и объект Список Уровней UI, который является контейнером для хранения скрипта Main Menu. Игровой объект Класс типа Текст и содержанием текстового поля «ВЫБЕРИ КЛАСС» отображается на экране пользователя и дает указания по взаимодействию с экраном. Далее следуют кнопки Первый, Второй, Третий и Четвертый, отвечающее за выполнение функций Class1, Class2, Class3 и Class4 скрипта Main Menu соответственно. Объекты cup1, cup2, cup3 и cup4, относящиеся к типу «Изображение», отвечают за отображение призовых кубков для каждого из доступных классов. Игровой объект Назад типа «Кнопка» отвечает за деактивацию холста LevelsUI и активацию холста MenuUI.

Игровой объект ShopUI отвечает за отображение элементов пользовательского интерфейса на экране магазина.

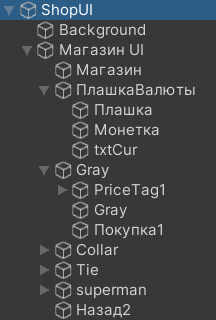


Рис. 19 – иерархия объектов на холсте ShopUI



Рис. 20 – ShopUI

Фоновое изображение Background, контейнер для скрипта Магазин UI и текст Магазин аналогичны подобным объектам в других холстах.

Группа объектов ПлашкаВалюты отвечает за графическое отображение заработанных пользователем монет и состоит из изображений Плашка и Монета и текста TxtCur.

Далее следуют группы объектов Gray, Collar, Tie и superman, отвечающие за графическое отображение изображений персонажа (изображение, на рис. 29 - Gray), его стоимости (текст, на рис. 15 – содержимое группы объектов PriceTag1) и кнопки покупки/выбора персонажа (кнопка, на рис. 15 – Покупка1).

Группа объектов PriceTag отвечает за отображение ценника каждого из доступных объектов, изображение с названием доступного объекта отвечает за графическое отображение данного объекта и служит контейнером для скрипта Shop с данными каждого объекта.

Кнопка Покупка реализует функцию buy в скрипте Shop. Кнопка Назад2 отвечает за деактивацию холста ShopUI и активацию холста MenuUI.

Игровой объект SettingsUI отвечает за отображение элементов пользовательского интерфейса на экране настроек.

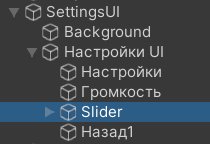


Рис. 21 – иерархия объектов на холсте SettingsUI



Рис. 22 – SettingsUI

Фоновое изображение Background, контейнер для скрипта Настройки UI и текст Настройки аналогичны подобным объектам в других холстах. Текст Громкость отображает на экране пользователя текстовое содержание «ГРОМКОСТЬ ЗВУКА».

«Слайдер» Slider (Slider) отвечает за объект UI, позволяющий с помощью скрипта Audio и его функции ChangeVolume менять громкость музыкального сопровождения на всех игровых сценах.

Кнопка Назад1 отвечает за деактивацию холста SettingsUI и активацию холста MenuUI.

Игровой объект chooseClass появляется при отсутствии информации о последнем запущенном уровне и содержит изображение popup, текст warning с содержанием «Пожалуйста, выбери свой класс в разделе уровней!» и копку back, деактивирующую холст chooseClass и активирующую холст MenuUI.

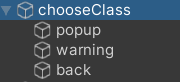


Рис. 23 – иерархия объектов на холсте chooseClass



Рис. 24 – chooseClass и MenuUI

Перейдем к рассмотрению скриптов, относящихся в данной игровой сцене.

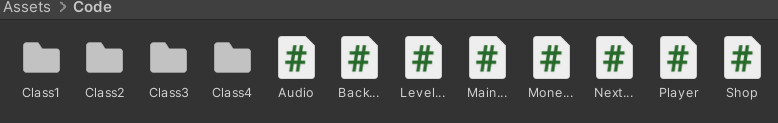


Рис. 25 – скрипты в папке ассетов Unity-проекта

Скрипт MainMenu обрабатывает взаимодействие пользователя с главным меню. Внутренний API приложения содержит следующие методы:

* Функция Play;
* Функция Class1;
* Функция Class2;
* Функция Class3;
* Функция Class4;
* Функция Exit;

Функция Play вызывается при нажатии кнопки Играть на холсте MenuUI.

Далее в цикле если-то (if-else) спрашивается, есть ли у пользователя сохранённая информация о последнем запущенном уровне. Для этого используется встроенная функция сохранения PlayerPrefs и её метод HasKey, которой по ключу «CurrentClass» определяет его значение. Если ключ отсутствует, иными словами если пользователь ещё не запустил ни один из уровней, то активируется холст chosseClass. Если же ключ есть, то с помощью оператора выбора switch с условием, соответствующим значению ключу «CurrentClass», происходит переход на другую игровую сцену. Иными словами, если игрок ранее уже второго класса, то значение «CurrentClass» равно 2 и оператор switch рассмотрит случай 2 (case 2) и перенаправит игрока на игровую сцену “Class2”.

Функции Class1, Class2, Class3 и Class4 вызываются при нажатии соответствующей кнопки на холсте LevelsUI. Функция переключает игрока на соответствующую кнопке игровую сцену и сохраняет на устройстве значение текущего класса (PlayerPrefs.SetInt). Таким образом при нажатии кнопки Четвертый на LevelsUI игрок перейдет на игровую цену 4 класса, а в памяти устройства сохранится соответствующее значение ключа «CurrentClass».

Функция Exit вызывается при нажатии кнопки Выход на MenuUI. Она сохраняет игровые данные на устройстве пользователя с помощью функции PlayerPrefs.Save и закрывает приложение.

Скрипт LevelsManager обрабатывает отображение призовых кубков на холсте LevelsUI. Внутренний API приложения содержит следующие методы:

* Функция Start;
* Функция Update;
* Функция updateCupImage;

Функция Start вызывается с первым кадром и отображает состояние первого захода в приложение. По умолчанию все объекты cup деактивированы. Функция Update каждый кадр вызывает функцию updateCupImage. В свою очередь, данная функция с помощью оператора выбора switch с условием, соответствующим значению ключу «CurrentClass», запускает цикл с условием «Был ли пройден 12 уровень в соответствующем классе?». Если да, то объект cup активируется поверх кнопки выбора класса.

Скрипт MoneyManager обрабатывает все действия с денежными средствами пользователя. Внутренний API приложения содержит следующие методы:

* Функция Start;
* Функция Update;
* Функция earnMoney;
* Функция returnMoney;

Функция Update каждый кадр получает информацию о количестве денежных средств на его счете и выводит её на холст ShopUI с помощью текста TxtCur. Для этого значение ключа «Money» переводится из целого числа (int) в строку (string) с помощью метода ToString(). Функция earnMoney отвечает за заработок денег. При правильном ответе на вопрос эта функция вызывается и увеличивает значение переменной денег на 1, а также сохраняет её новое значение. Функция returnMoney возвращает значение переменной money для его дальнейшего использования в других скриптах.

Скрипт Shop обрабатывает все действия на ShopUI. Внутренний API приложения содержит следующие методы:

* Функция Start;
* Функция Update;
* Функция buy;

Функция Start вызывается с первым кадром и отображает состояние первого захода в приложение. Цикл «если» запускается, если на устройстве нет информации о персонаже по умолчанию, тогда для каждого персонажа из массива изображений сохраняется значение, соответствующее имени файла и строки «buy» с числовым значением 0, кроме персонажа по умолчанию с именем Gray, которому присваиваются значения «Graybuy» и «Grayselected» и числовые значения 1. Иными словами, при первом заходе в приложение персонаж Gray сохраняется как купленный и выбранный, а все другие персонажи как некупленные.

Функция Update каждый кадр получает информацию о статусе покупки каждого персонажа и о том, какой персонаж выбран игровым. Если значение сохранения по ключу имя персонажа + информация о покупке равно 0, то персонаж считается не купленным и соответствующая кнопка на ShopUI отображает изображение «купить». Если значение сохранения по ключу имя персонажа + информация о покупке равно 1, то персонаж считается купленным. Тогда идет проверка на условие, а выбран ли этот персонаж в качестве игрового. Если значение сохранения по ключу имя персонажа + информация о выборе равно 1, то персонаж считается выбранным и соответствующая кнопка на ShopUI отображает изображение «выбрано». Если значение сохранения по ключу имя персонажа + информация о выборе равно 0, то персонаж считается не выбранным и соответствующая кнопка на ShopUI отображает изображение «выбрать».

Функция buy вызывается при нажатии кнопки Покупка на ShopUI. При условии, что значение переменной money из скрипта MoneyManager больше чем переменная price, отвечающая за цену каждого из персонажей, и персонаж не куплен, то запускается цикл покупки персонажа. Изображение кнопки Покупка на ShopUI меняется на «выбрано», переменная, отвечающая за хранение информации о денежных средствах пользователя становится равным разности предыдущего значения и цены персонажа, сохранение по ключу имя персонажа + информация о покупке становится равным 1, а номер персонажа сохраняется как номер игрового персонажа. Далее для каждого изображения в массиве персонажей меняются значения ключей сохранения, отвечающие за информацию о выборе персонажа.

Если персонаж уже был приобретён, то происходит смена выбранного персонажа. По аналогии с предыдущем описанием происходит смена ключей сохранения, отвечающих за информацию о выборе персонажа.

Скрипт Audio отвечает за изменение и сохранение значения громкости звука на SettingsUI. Внутренний API приложения содержит следующие методы:

* Функция Start;
* Функция ChangeVolume;

Функция Start вызывается с первым кадром и отображает состояние первого захода в приложение. Если в сохранении отсутствует ключ «musicVolume» с информацией и предпочитаемом уровне громкости, то устанавливается значение по умолчанию типа число с плавающей точкой (float) равное 0,5 и вызывается функция загрузки значения Слайдера на SettingsUI, которой становится равен значению ключа «musicVolume» сохранения PlayerPrefs. В противном случае функция запускается сразу.

Функция ChangeVolume изменят значение компонента AudioListener игрового объекта Main Camera, таким образом изменяя громкость получаемого камерой звука и сохраняет значению ключа «musicVolume» сохранения PlayerPrefs.

Меню уровней и меню уровня

Перейдем к рассмотрению игровых сцен Class1, Class2, Class3 и Class4. Поскольку для реализации каждой сцены использованы аналогичные механики, а сами сцены отличаются между собой только текстовыми элементами UI, отображающими текст каждой задачи, то рассмотрим лишь одну из игровых сцен.

Объекты tutorial и Congrats являются холстами и отвечают за взаимодействие с пользователем с помощью текстовых элементов UI.

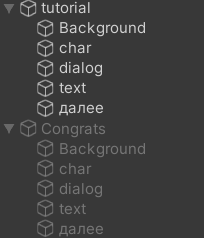


Рис. 26 – иерархия объектов на холстах tutorial и Congrats



Рис. 27 – tutorial и Congrats

Оба холста содержат фоновое изображение Background, спрайт персонажа char, изображение диалогового окна dialog, текст text с указаниями для пользователя и кнопки далее, отвечающие за переход к другому холсту. Холст tutorial появляется при открытии игровой сцены и содержит текст «Правильно отвечай на вопросы и зарабатывай монеты, чтобы порадовать меня обновками!», который объясняет пользователю принципы получения монет и их использования. Кнопка далее на tutorial деактивирует данный холст и активирует холст Menu. Холст Congrats появляется после прохождения последнего, 12 уровня, и содержит текст «Поздравляю с успешным решением задач! Ты сможешь найти свой кубок в разделе уровней». Кнопка далее на Congrats деактивирует холст и вызывает функцию BackToMainMenu скрипта Back to Menu.

Рассмотрим объекты char на обоих холстах. Объекты являются 2D элементами типа «Спрайт» (Sprite) и отвечают за отображение графических элементов на экране. Обратим внимание на компонент «Отображение спрайта» (Sprite Renderer), который отвечает показ элемента, и его свойства «Спрайт» (Sprite) и «Порядок слоёв» (Order in Layer). «Спрайт» отвечает за источник изображения, в данном примере он соответствует файлу «Gray.png», а «Порядок слоёв» номер слоя, на котором находится объект, в данном примере значение равно 1. В контейнере для скрипта Player собраны файлы, являющиеся источниками изображений других версий персонажа.

Игровой объект Menu отвечает за отображение элементов пользовательского интерфейса на экране выбора уровня.



Рис. 28 – иерархия объектов на холсте Menu в Class1



Рис. 29 – Menu в Class1

Menu содержит фоновое изображение Background, текст 1 klass и группу объектов Levels1. Группа содержит 12 кнопок для перехода на холст соответствующего уровня и 12 изображений замка, обозначающих доступность или недоступность данного уровня для пользователя.

Рассмотрим принцип работы всех кнопок на кнопке 1 уровня. При нажатии на кнопку происходит деактивация холста Menu и активация холста Level1. Кнопка также является контейнером для скрипта LevelManager1 и содержит необходимую для работы этого скрипта уникальную информацию.

Изображение замка 1 содержит компонент «Изображение», отвечающий в том числе за источник изображения, по умолчанию являющийся файлом «lockClosed.png».

Кнопка Назад1 по нажатию переходит с текущей сцены на сцену Menu.

Игровой объект Level1 отвечает за отображение элементов пользовательского интерфейса на экране выбора уровня.

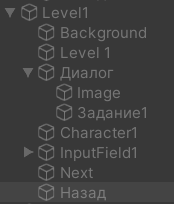


Рис. 30 – иерархия объектов на холсте Level1 в Class1

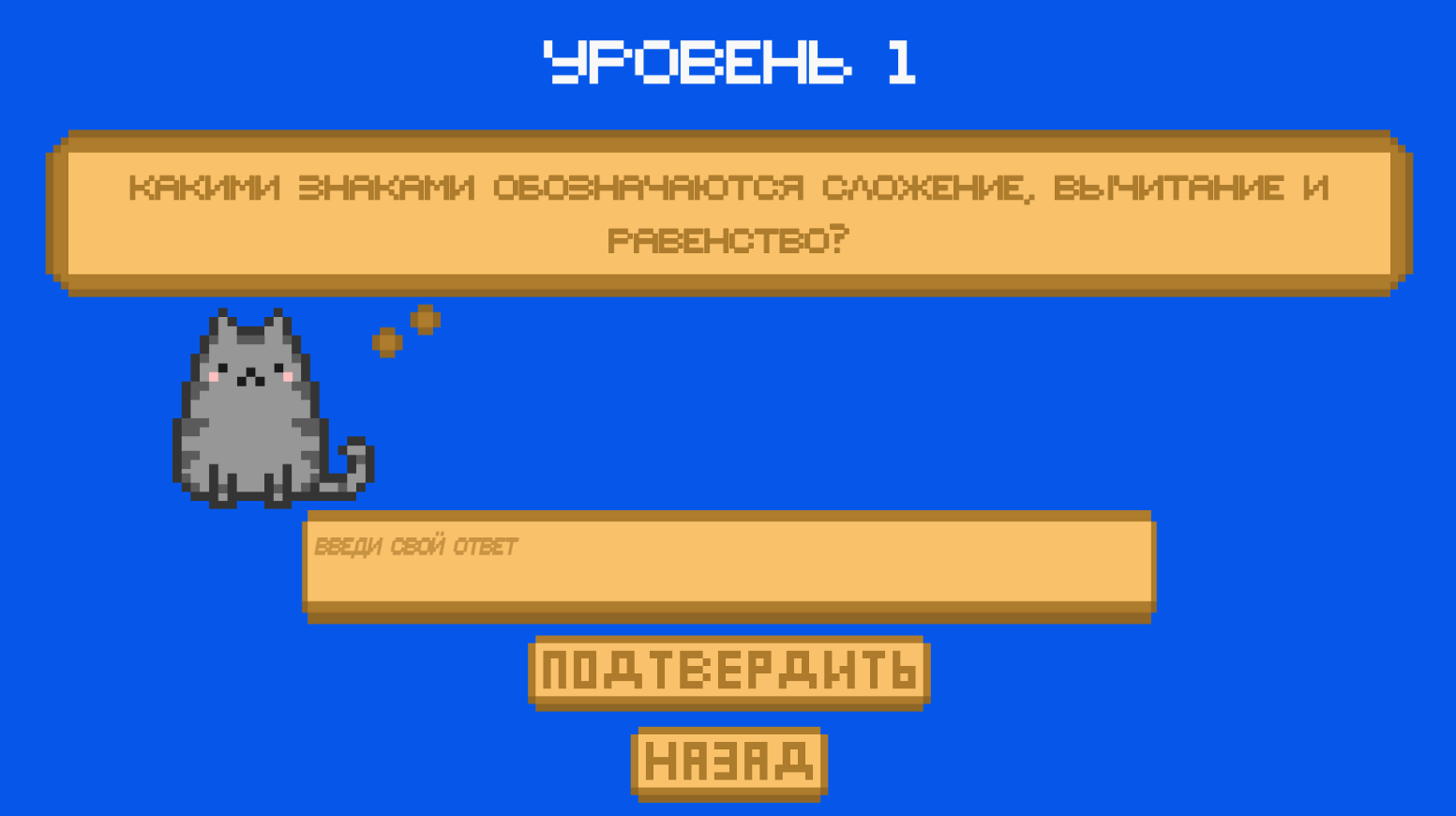


Рис. 31 – Level1 в Class1

Он содержит фоновое изображение Background, текст Level, группу объектов Диалог, состоящую из изображения Image и текста Задание, спрайта персонажа Character, поля ввода текста InputField и кнопок Next и Назад.

Изображение Image отображает файл «dialog.png» с помощью компонента «Изображение» и его свойства «Источник изображения», а текст Задание содержит в себе текст задачи, соотносящейся к данному уровню. Например, для уровня 1 содержание текста Задание1 будет следующим: «Какими знаками обозначаются сложение, вычитание и равенство?». Спрайт Character имеет такие же компоненты и их свойства, как ранее описанный спрайт char. С помощью поля ввода InputField пользователь вводит свой ответ на поставленный вопрос. Кнопка Next вызывает функцию inputedAnswer в скрипте LevelsButtons на основе уникальных свойств каждого уровня.

Кнопка Назад по нажатию закрывает текущий холст и отрывает холст Menu.

По такому же сценарию реализованы и другие уровни в данной сцене.

Холст correctAnswer становится активным, если пользователь правильно отвечает на задание.

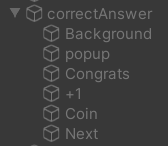


Рис. 32 – иерархия объектов на холсте correctAnswer



Рис. 33 – correctAnswer

Объектами холста являются фоновое изображение, текст с поздравлениями, текст «+1» и изображение монеты, символизирующих увеличение счета пользователя на одну монету, а также кнопка Next, которая закрывает данный холст, активирует холст Menu и вызывает функцию earnMoney из скрипта MoneyManager.

Холст uncorrectAnswer активируется поверх холста уровня и с помощью текстового элемента wrong сообщает пользователю, что он ввел неверный ответ. Кнопка back закрывает холст.

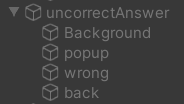


Рис. 34 – иерархия объектов на холсте uncorrectAnswer



Рис. 35 – uncorrectAnswer и Level1

Рассмотрим скрипты, используемые в данной сцене.

Скрипт BackToMenu обрабатывает взаимодействие пользователя с интерфейсом игровой сцены. Внутренний API приложения содержит следующие методы:

* Функция BackToMainMenu;

Функция BackToMainMenu из скрипта BackToMenu вызывается при нажатии нескольких кнопок Назад и меняет текущую сцену на сцену «Menu».

Скрипт Player изменят каждый объект типа Спрайт в данной сцене в зависимости от значения сохранения по ключу «character». Иными словами, при смене внешнего вида персонажа в сцене Menu, его значение сохраняется, и при открытии другой сцены, персонаж отображается в зависимости от выбранного внешнего вида. Внутренний API приложения содержит следующие методы:

* Функция Start;

Скрипт LevelsManager1 отвечает за корректное отображение кнопок на холсте Menu в игровой сцене Class1 и другие свойства холста. С первым кадром звук, получаемый объектом MainCamera от объекта EventSystem регулируется согласно значению ключа сохранения. Внутренний API приложения содержит следующие методы:

* Функция Start;
* Функция Update;
* Функция UpdateLevelImage;

Функция UpdateLevelImage использует оператор выбора switch с логической переменной unlocked, которая несёт в себе информацию доступен тот или иной уровень для прохождения. Если переменная соответствует истине и уровень разблокирован, то кнопка уровня становится доступной для нажатия, изображение замка сменяется на открытый замок или становится неактивным, если уровень уже пройден. Если переменная ложна, то кнопка недоступна для нажатия и изображение замка становится закрытым. Функция UpdateLevelStatus вычисляет номер предыдущего уровня и если значение ключа сохранения для него соответствует значению пройденного уровня, то текущий уровень становится разблокированным. Функция Update вызывает функции UpdateLevelImage и UpdateLevelStatus каждый кадр.

Скрипт LevelsButtons1 обрабатывает ответы пользователя и переход между холстами уровней в игровой сцене Class1. Внутренний API приложения содержит следующие методы:

* Функция inputedAnswer;
* Функция ToTheNextLevel;

Функция inputedAnswer вызывается при нажатии кнопки Next на холсте уровня. В ней содержимое поля ввода, которое заполнил, пользователь, приводится к стандартному виду с помощью методов ToLower и Replace(" ", string.Empty), которые приводят текст к нижнему регистру и удаляют пробелы. Дальше следует проверка, какой уровень запущен, и если введённый пользователем ответ является верным, то холст уровня закрывается, открывается холст correctAnswer и вызывается функция ToTheNextLevel. Если введён любой другой ответ кроме правильного, то активизируется холст uncorrectAnswer.

Функция ToTheNextLeАvel использует оператор выбора для номера текущего уровня, и если он больше нуля, то меняет значение ключа сохранения на значение пройденного уровня. Если номер текущего уровня равен 12, то логическая переменная isItTwelveLevel становится равной истине. Если пользователь еще не прошёл ни один уровень, то значение ключа сохранения становится равным значению непройденного уровня.

Отличия в версии для операционной системы Android

Изменение в разрешении экрана привели к пересмотру способа расположения элементов на холсте ShopUI в сцене Menu и холстах Menu в сценах Class1, Class2, Class3 и Class4. Для размещения всех необходимых изображений и кнопок на разрешении экрана в 720 x 1280 писклей была реализована возможность проматывать элементы с помощью прикосновения к экрану устройства (скролл).

Для этого был создан игровой объект Scroll, который является контейнером для другого объекта – ScrollContent. Компонент «Прямоугольная маска 2D» (Rect Mask 2D) отвечает на наложение маски на всю площадь игрового объекта. Таким образом контент, относящийся к потомкам данного объекта, будет виден только в ограниченном пространстве. Компонент «Прямоугольник скролла» (Scroll Rect) отвечает за возможность скроллить обекты. В свойстве «Контент» (Content) хранится информация о группе объектов, которые можно пролистывать. Свойства «Горизонталь» (Horizontal) и «Вертикаль» (Vertical) определяют по какой оси реализуется возможность пролистывать объекты.

ScrollContent также содержит в себе игровой объект Content, отвечающий за группировку всех элементов лотов магазина на холсте ShopUI и кнопок выбора уровня холстах Menu. Группировка всех объектов в таблицу осуществляется с помощью компонента «Группа компоновки сетки» (Grid Layout Group).

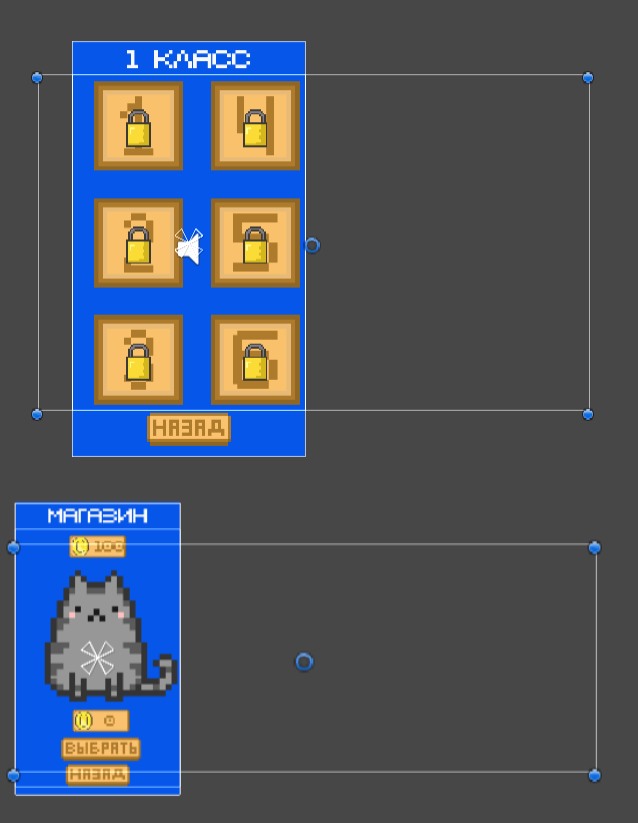


Рис. 36 – Границы объектов Content на хостах ShopUI и Menu

# **3.3 Тестирование**

Во время тестирования обеих версий приложений были выявлены и устранены проблемы с размещением объектов, их размеров и границ. Программный код версии для устройств с операционной системой Android был переработан для оптимизации производительности. Для всех функций Update введено условие активности соответствующего холста с помощью метода. activeSelf. Это помогло снизить количество обновлении экрана приложения, следовательно, снизить нагрузку на устройства пользователя.

Обе версии приложения были успешно протестированы на различных физических и виртуальных устройствах. Итоговые характеристики приложений:

Windows:

* Общий размер проекта Unity: 299 Мб
* Размер папки с ассетами: 24,6 Мб
* Размер исходного файла в формате exe: 625 Кб

Android:

* Общий размер проекта Unity: 356 Мб
* Размер папки с ассетами: 14,7 Мб
* Размер исходного файла в формате apk: 26,6 Мб

# **Заключение**

В ходе реализации выпускной квалификационной работой были выполнены все поставленные цели и задачи. Были изучены педагогические приёмы и на их основе был произведен подбор подходящих математических задач. Изучение аналогов позволило отобрать наиболее удачные идеи для разработки приложения и избежать неудачных решений при проектировании. Были разработаны структура и аудиовизуальный стиль приложения, определившие дальнейший вектор разработки. Заключительным этапом стала разработка и тестирование прототипов приложений.

Результатом достижения всех целей является создание многоплотформенного образовательного приложения «Интерактивные задачи по математике», готовое для реализации в цифровых магазинах.

# **Использованные источники**

1. Мурзагалиева А.Е., Утегенова Б.М. - Сборник заданий и упражнений. Учебные цели согласно таксономии Блума - Астана: АОО «Назарбаев Интеллектуальные школы» Центр педагогического мастерства, 2015. – 54с.;
2. Дьяконова О. О. - Понятие «Эдьютейнмент» в зарубежной и отечественной педагогике – Сибирский педагогический журнал № 6 – 2012;
3. Gaevskaya E. - Education or Edutainment? - Electronic Imaging &amp; the Visual Arts: EVA 2012 Florence, 9-11 May 2012;
4. Моро М. И., Волкова С. И., Степанова С. В. - Математика. 1 класс. В 2-х ч. - Издательство «Просвещение» - 2015;
5. Моро М. И., Бантова М. А., Бельтюкова Г. В. и др. - Математика. 2 класс. В 2-х ч. - Издательство «Просвещение» - 2014;
6. Моро М. И., Бантова М. А., Бельтюкова Г. В. и др. - Математика. 3 класс. В 2-х ч. - Издательство «Просвещение» - 2015;
7. Моро М. И., Бантова М. А., Бельтюкова Г. В. и др. - Математика. 4 класс. В 2-х ч. - Издательство «Просвещение» - 2015;
8. «Unity User Manual» [Электронный ресурс] / Unity Documentation – Режим доступа: docs.unity3d.com/Manual/, свободный. Загл. с экрана. – Яз. англ.;

# **Приложения**

Скрипт MainMenu:

|  |
| --- |
| public class MainMenu : MonoBehaviour  {  public GameObject mainMenu;  public GameObject chooseClass;  public void Play()  {  if (PlayerPrefs.HasKey("CurrentClass") == false)  {  chooseClass.SetActive(true);  }  else  {  switch (PlayerPrefs.GetInt("CurrentClass"))  {  case 1:  SceneManager.LoadScene("Class1");  break;  case 2:  SceneManager.LoadScene("Class2");  break;  case 3:  SceneManager.LoadScene("Class3");  break;  case 4:  SceneManager.LoadScene("Class4");  break;  }  }  }  public void Class1()  {  SceneManager.LoadScene("Class1");  PlayerPrefs.SetInt("CurrentClass", 1);  }  public void Class2()  {  SceneManager.LoadScene("Class2");  PlayerPrefs.SetInt("CurrentClass", 2);  }  public void Class3()  {  SceneManager.LoadScene("Class3");  PlayerPrefs.SetInt("CurrentClass", 3);  }  public void Class4()  {  SceneManager.LoadScene("Class4");  PlayerPrefs.SetInt("CurrentClass", 4);  }  public void Exit()  {  PlayerPrefs.Save();  Application.Quit();  }  } |

Скрипт LevelsManager:

|  |
| --- |
| public class LevelsManager : MonoBehaviour  {  public Image cupImage1;  public Image cupImage2;  public Image cupImage3;  public Image cupImage4;  void Start()  {  cupImage1.gameObject.SetActive(false); ;  cupImage2.gameObject.SetActive(false);  cupImage3.gameObject.SetActive(false);  cupImage4.gameObject.SetActive(false);  }  private void Update()  {  updateCupImage();  }  private void updateCupImage()  {  switch (PlayerPrefs.GetInt("CurrentClass"))  {  case 1:  if (PlayerPrefs.HasKey("1Class12"))  {  cupImage1.gameObject.SetActive(true);  }  break;  case 2:  if (PlayerPrefs.HasKey("2Class12"))  {  cupImage2.gameObject.SetActive(true);  }  break;  case 3:  if (PlayerPrefs.HasKey("3Class12"))  {  cupImage3.gameObject.SetActive(true);  }  break;  case 4:  if (PlayerPrefs.HasKey("4Class12"))  {  cupImage4.gameObject.SetActive(true);  }  break;  }  }  } |

Скрипт MoneyManager:

|  |
| --- |
| public class MoneyManager : MonoBehaviour  {  public static MoneyManager Instance;  public Text moneyText;  int money;  void Start()  {  if (Instance == null)  {  Instance = this;  }  }  void Update()  {  money = PlayerPrefs.GetInt("Money");  moneyText.text = money.ToString();  }  public void earnMoney()  {  money++;  PlayerPrefs.SetInt("Money", money);  }  public int returnMoney()  {  return money;  }  } |

Скрипт Shop:

|  |
| --- |
| public class Shop : MonoBehaviour  {  public int charNum;  public Button buyButton;  public int price;  public Sprite purch;  public Sprite selected;  public Sprite select;  public Image[] chars;  private void Start()  {  if (PlayerPrefs.GetInt("Gray" + "buy") == 0)  {  foreach (Image i in chars)  {  if ("Gray" == i.name)  {  PlayerPrefs.SetInt("Gray" + "buy", 1);  PlayerPrefs.SetInt("Gray" + "select", 1);  }  else  {  PlayerPrefs.SetInt(GetComponent<Image>().name + "buy", 0);  }  }  }  }  private void Update()  {  if (PlayerPrefs.GetInt(GetComponent<Image>().name + "buy") == 0)  {  buyButton.GetComponent<Image>().sprite = purch;  }  else if (PlayerPrefs.GetInt(GetComponent<Image>().name + "buy") == 1)  {  if (PlayerPrefs.GetInt(GetComponent<Image>().name + "select") == 1)  {  buyButton.GetComponent<Image>().sprite = selected;  }  else if (PlayerPrefs.GetInt(GetComponent<Image>().name + "select") == 0)  {  buyButton.GetComponent<Image>().sprite = select;  }  }  }  public void buy()  {  if (PlayerPrefs.GetInt(GetComponent<Image>().name + "buy") == 0)  {  if (MoneyManager.Instance.returnMoney() >= price)  {  buyButton.GetComponent<Image>().sprite = selected;  PlayerPrefs.SetInt("Money", PlayerPrefs.GetInt("Money") - price);  PlayerPrefs.SetInt(GetComponent<Image>().name + "buy", 1);  PlayerPrefs.SetInt("characater", charNum);  foreach (Image i in chars) {  if (GetComponent<Image>().name == i.name)  {  PlayerPrefs.SetInt(GetComponent<Image>().name + "select", 1);  }  else  {  PlayerPrefs.SetInt(i.name + "select", 0);  }  }  }  }  else if (PlayerPrefs.GetInt(GetComponent<Image>().name + "buy") == 1)  {  buyButton.GetComponent<Image>().sprite = selected;  PlayerPrefs.SetInt(GetComponent<Image>().name + "select", 1);  PlayerPrefs.SetInt("characater", charNum);  foreach (Image i in chars)  {  if (GetComponent<Image>().name == i.name)  {  PlayerPrefs.SetInt(GetComponent<Image>().name + "select", 1); }  else  {  PlayerPrefs.SetInt(i.name + "select", 0);  }  }  }  }  } |

Скрипт Audio:

|  |
| --- |
| public class Audio : MonoBehaviour  {  [SerializeField] Slider volumeSlider;  void Start()  {  if (!PlayerPrefs.HasKey("musicVolume"))  {  PlayerPrefs.SetFloat("musicVolume", 0.5f);  }  volumeSlider.value=PlayerPrefs.GetFloat("musicVolume");  }    public void ChangeVolume()  {  AudioListener.volume = volumeSlider.value;  PlayerPrefs.SetFloat("musicVolume", volumeSlider.value);  }  } |

Скрипт BackToMenu:

|  |
| --- |
| public class BackToMenu : MonoBehaviour  {  public void BackToMainMenu()  {  SceneManager.LoadScene("Menu");  }  } |

Скрипт Player:

|  |
| --- |
| public class Player : MonoBehaviour  {  public Sprite def;  public Sprite one;  public Sprite two;  public Sprite three;  public GameObject cat;  void Start()  {  switch (PlayerPrefs.GetInt("characater"))  {  case 1:  cat.GetComponent<SpriteRenderer>().sprite = one;  break;  case 2:  cat.GetComponent<SpriteRenderer>().sprite = two;  break;  case 3:  cat.GetComponent<SpriteRenderer>().sprite = three;  break;  default:  cat.GetComponent<SpriteRenderer>().sprite = def;  break;  }  }  } |

Скрипт LevelsManager1:

|  |
| --- |
| public class LevelsManager1 : MonoBehaviour  {  [SerializeField] private bool unlocked;  public Image lockImage;  public Sprite lockedSprite;  public Sprite unlockedSprite;  public Button levelButton;  public int currentLevel;  private void Start()  {  AudioListener.volume = PlayerPrefs.GetFloat("musicVolume");  }    private void Update()  {  UpdateLevelImage();  UpdateLevelStatus();  }  private void UpdateLevelImage()  {  switch (unlocked)  {  case true:  lockImage.GetComponent<Image>().sprite = unlockedSprite;  levelButton.interactable = true;  if (PlayerPrefs.GetInt("1Class" + currentLevel) == 1)  {  lockImage.gameObject.SetActive(false);  }  break;  case false:  lockImage.GetComponent<Image>().sprite = lockedSprite;  levelButton.interactable = false;  break;  }  }  private void UpdateLevelStatus()  {  int prevLevel = int.Parse(gameObject.name) - 1;  if (PlayerPrefs.GetInt("1Class" + prevLevel.ToString()) == 1)  {  unlocked = true;  }  }  } |

Скрипт LevelsButtons1:

|  |
| --- |
| public class LevelsButtons1 : MonoBehaviour  {  public int currentLevel;  public static bool isItTwelveLevel;  public InputField playersAnswer;  public GameObject correctAnswer;  public GameObject uncorrectAnswer;  public GameObject levelMenu;  private string toLowerAnswer;  private string answer;  public void inputedAnswer()  {  toLowerAnswer = playersAnswer.text.ToLower();  answer = toLowerAnswer.Replace(" ", string.Empty);  if (currentLevel == 1)  {  if (answer == "+-=" | answer == "+=-" | answer == "-+=" | answer == "-=+" | answer == "=+-" | answer == "=-+")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 2)  {  if (answer == "12" | answer == "21")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 3)  {  if (answer == "1+4=5")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 4)  {  if (answer == "квадрат")  {  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 5)  {  if (answer == "-=")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 6)  {  if (answer == "многоугольники" | answer == "многоугольник")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 7)  {  if (answer == "в")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 8)  {  if (answer == "3")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 9)  {  if (answer == "7-2=5")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 10)  {  if (answer == "зелёного" | answer == "зеленого" | answer == "зеленый" | answer == "зелёный")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 11)  {  if (answer != "")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else  {  if (answer != "")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  }  public void ToTheNextLevel()  {  levelMenu.SetActive(false);  correctAnswer.SetActive(true);  switch (currentLevel)  {  case int lvl when lvl > 0:  PlayerPrefs.SetInt("1Class" + currentLevel, 1);  MoneyManager.Instance.earnMoney();  PlayerPrefs.SetInt("CurrentClass", 1);  if (lvl == 12)  isItTwelveLevel = true;  else isItTwelveLevel = false;  break;  default:  PlayerPrefs.SetInt("1Class" + 1, 0);  isItTwelveLevel = false;  break;  }    } |

Скрипт LevelsManager2:

|  |
| --- |
| public class LevelsManager2 : MonoBehaviour  {  [SerializeField] private bool unlocked;  public Image lockImage;  public Sprite lockedSprite;  public Sprite unlockedSprite;  public Button levelButton;  public int currentLevel;  private void Start()  {  AudioListener.volume = PlayerPrefs.GetFloat("musicVolume");  }  private void Update()  {  UpdateLevelImage();  UpdateLevelStatus();  }  private void UpdateLevelImage()  {  switch (unlocked)  {  case true:  lockImage.GetComponent<Image>().sprite = unlockedSprite;  levelButton.interactable = true;  if (PlayerPrefs.GetInt("2Class" + currentLevel) == 1)  {  lockImage.gameObject.SetActive(false);  }  break;  case false:  lockImage.GetComponent<Image>().sprite = lockedSprite;  levelButton.interactable = false;  break;  }  }  private void UpdateLevelStatus()  {  int prevLevel = int.Parse(gameObject.name) - 1;  if (PlayerPrefs.GetInt("2Class" + prevLevel.ToString()) == 1)  {  unlocked = true;  }  } |

Скрипт LevelsButtons2:

|  |
| --- |
| public class LevelsButtons2 : MonoBehaviour  {  public int currentLevel;  public static bool isItTwelveLevel;  public InputField playersAnswer;  public GameObject correctAnswer;  public GameObject uncorrectAnswer;  public GameObject levelMenu;  private string toLowerAnswer;  private string answer;  public void inputedAnswer()  {  toLowerAnswer = playersAnswer.text.ToLower();  answer = toLowerAnswer.Replace(" ", string.Empty);  if (currentLevel == 1)  {  if (answer == "\*/" | answer == "/\*" | answer == "x/" | answer == "/x" | answer == "х/" | answer == "/х" | answer == "\*:" | answer == ":\*" | answer == "x:" | answer == ":x" | answer == "х:" | answer == ":х")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 2)  {  if (answer == "суммасторон")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 3)  {  if (answer == "20")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 4)  {  if (answer == "16")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 5)  {  if (answer == "г")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 6)  {  if (answer == "251")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 7)  {  if (answer == ">")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 8)  {  if (answer == "3")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 9)  {  if (answer == "18/2=9" | answer == "18:2=9")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 10)  {  if (answer == "5")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 11)  {  if (answer != "")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else  {  if (answer != "")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  }  public void ToTheNextLevel()  {  levelMenu.SetActive(false);  correctAnswer.SetActive(true);  switch (currentLevel)  {  case int lvl when lvl > 0:  PlayerPrefs.SetInt("2Class" + currentLevel, 1);  MoneyManager.Instance.earnMoney();  PlayerPrefs.SetInt("CurrentClass", 2);  if (lvl == 12)  isItTwelveLevel = true;  else isItTwelveLevel = false;  break;  default:  PlayerPrefs.SetInt("2Class" + 1, 0);  isItTwelveLevel = false;  break;  }  }  } |

Скрипт LevelsManager3:

|  |
| --- |
| public class LevelsManager3 : MonoBehaviour  {  [SerializeField] private bool unlocked;  public Image lockImage;  public Sprite lockedSprite;  public Sprite unlockedSprite;  public Button levelButton;  public int currentLevel;  private void Start()  {  AudioListener.volume = PlayerPrefs.GetFloat("musicVolume");  }  private void Update()  {  UpdateLevelImage();  UpdateLevelStatus();  }  private void UpdateLevelImage()  {  switch (unlocked)  {  case true:  lockImage.GetComponent<Image>().sprite = unlockedSprite;  levelButton.interactable = true;  if (PlayerPrefs.GetInt("3Class" + currentLevel) == 1)  {  lockImage.gameObject.SetActive(false);  }  break;  case false:  lockImage.GetComponent<Image>().sprite = lockedSprite;  levelButton.interactable = false;  break;  }  }  private void UpdateLevelStatus()  {  int prevLevel = int.Parse(gameObject.name) - 1;  if (PlayerPrefs.GetInt("3Class" + prevLevel.ToString()) == 1)  {  unlocked = true;  }  } |

Скрипт LevelsButtons3:

|  |
| --- |
| public class LevelsButtons3 : MonoBehaviour  {  public int currentLevel;  public static bool isItTwelveLevel;  public InputField playersAnswer;  public GameObject correctAnswer;  public GameObject uncorrectAnswer;  public GameObject levelMenu;  private string toLowerAnswer;  private string answer;  public void inputedAnswer()  {  toLowerAnswer = playersAnswer.text.ToLower();  answer = toLowerAnswer.Replace(" ", string.Empty);  if (currentLevel == 1)  {  if (answer == "корнем")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 2)  {  if (answer == "миллиметр,сантиметр,дециметр,метр")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 3)  {  if (answer == "99")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 4)  {  if (answer == "500")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 5)  {  if (answer == "6")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 6)  {  if (answer == "256" | answer == "265" | answer == "526" | answer == "562" | answer == "625" | answer == "652")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 7)  {  if (answer == "548")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 8)  {  if (answer == "8")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 9)  {  if (answer == "нет" | answer == "неверно")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 10)  {  if (answer == "231")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 11)  {  if (answer != "")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else  {  if (answer != "")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  }  public void ToTheNextLevel()  {  levelMenu.SetActive(false);  correctAnswer.SetActive(true);  switch (currentLevel)  {  case int lvl when lvl > 0:  PlayerPrefs.SetInt("3Class" + currentLevel, 1);  MoneyManager.Instance.earnMoney();  PlayerPrefs.SetInt("CurrentClass", 3);  if (lvl == 12)  isItTwelveLevel = true;  else isItTwelveLevel = false;  break;  default:  PlayerPrefs.SetInt("3Class" + 1, 0);  isItTwelveLevel = false;  break;  }  }  } |

Скрипт LevelsManager4:

|  |
| --- |
| public class LevelsManager4 : MonoBehaviour  {  [SerializeField] private bool unlocked;  public Image lockImage;  public Sprite lockedSprite;  public Sprite unlockedSprite;  public Button levelButton;  public int currentLevel;  private void Start()  {  AudioListener.volume = PlayerPrefs.GetFloat("musicVolume");  }  private void Update()  {  UpdateLevelImage();  UpdateLevelStatus();  }  private void UpdateLevelImage()  {  switch (unlocked)  {  case true:  lockImage.GetComponent<Image>().sprite = unlockedSprite;  levelButton.interactable = true;  if (PlayerPrefs.GetInt("4Class" + currentLevel) == 1)  {  lockImage.gameObject.SetActive(false);  }  break;  case false:  lockImage.GetComponent<Image>().sprite = lockedSprite;  levelButton.interactable = false;  break;  }  }  private void UpdateLevelStatus()  {  int prevLevel = int.Parse(gameObject.name) - 1;  if (PlayerPrefs.GetInt("4Class" + prevLevel.ToString()) == 1)  {  unlocked = true;  }  } |

Скрипт LevelsButtons4:

|  |
| --- |
| public class LevelsButtons4 : MonoBehaviour  {  public int currentLevel;  public static bool isItTwelveLevel;  public InputField playersAnswer;  public GameObject correctAnswer;  public GameObject uncorrectAnswer;  public GameObject levelMenu;  private string toLowerAnswer;  private string answer;  public void inputedAnswer()  {  toLowerAnswer = playersAnswer.text.ToLower();  answer = toLowerAnswer.Replace(" ", string.Empty);  if (currentLevel == 1)  {  if (answer == "скорость")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 2)  {  if (answer == "180")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 3)  {  if (answer == "2176")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 4)  {  if (answer == "4")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 5)  {  if (answer == "2760-392")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 6)  {  if (answer == "3")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 7)  {  if (answer == "б")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 8)  {  if (answer == "1/85/83/52/3")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 9)  {  if (answer == "72")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 10)  {  if (answer == "90")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else if (currentLevel == 11)  {  if (answer != "")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  else  {  if (answer != "")  {  ToTheNextLevel();  }  else  {  uncorrectAnswer.SetActive(true);  }  }  }  public void ToTheNextLevel()  {  levelMenu.SetActive(false);  correctAnswer.SetActive(true);  switch (currentLevel)  {  case int lvl when lvl > 0:  PlayerPrefs.SetInt("4Class" + currentLevel, 1);  MoneyManager.Instance.earnMoney();  PlayerPrefs.SetInt("CurrentClass", 4);  if (lvl == 12)  isItTwelveLevel = true;  else isItTwelveLevel = false;  break;  default:  PlayerPrefs.SetInt("4Class" + 1, 0);  isItTwelveLevel = false;  break;  }  }  } |