Инженерная школа информационных технологий и робототехники

Направление 09.03.01 «Информатика и вычислительная техника»

Отделение информационных технологий

Отчёт по теме

**«Разработка игровых приложений на Unity»**

по дисциплине «Творческий проект»

Выполнил:

студент гр. 8В11 \_\_\_\_\_\_\_\_\_\_\_\_\_ Топчеева Д.А.

\_\_\_.\_\_\_.\_\_\_\_\_\_

Проверил:

Ассистент отделения ИТ

оценка (до 30 баллов): \_\_\_\_\_\_\_\_\_\_\_\_\_

к защите допускаю \_\_\_\_\_\_\_\_\_\_\_\_\_ Коровкин В. А.

\_\_\_.\_\_\_.\_\_\_\_\_\_

**Содержание**

[Введение 3](#_Toc106782530)

[Цели и задачи 3](#_Toc106782531)

[1. Целевая аудитория 4](#_Toc106782532)

[2. Использованные платформы 5](#_Toc106782533)

[2.1. Среда разработки Unity 5](#_Toc106782534)

[2.2. Язык программирования C# 7](#_Toc106782535)

[2.3. Графический редактор gimp 8](#_Toc106782536)

[2.4. Git 8](#_Toc106782537)

[3. Разработка игрового приложения 10](#_Toc106782538)

[3.1. Движение персонажа 10](#_Toc106782539)

[3.2. Окружающая среда 12](#_Toc106782540)

[Заключение 15](#_Toc106782541)

[Список источников: 16](#_Toc106782542)

# Введение

В современном мире развитие техники не стоит не месте. В последнее время все чаще можно услышать об увеличениях масштабов разработки программного обеспечения развлекательного характера. К числу таких продуктов относятся видеоигры.

Игры стали охватывать огромную аудиторию по всему миру, появляясь на различных игровых устройствах. Они стали сложнее и масштабнее: улучшилась обработка визуального пространства, обработка физики объектов и искусственный интеллект.

В данном проекте планируется создать игровое приложение на Unity, комфортное для использования.

# Цели и задачи

Создание игрового приложения на Unity

Для выполнения данной цели были поставлены следующие задачи:

1. Изучить среду разработки Unity.
2. Разработка игрового приложения.
   1. Механика движения.
   2. Окружающая среда.
3. Оптимизация и исправление ошибок.

# Целевая аудитория

Сегодня в РФ насчитывается свыше 65 млн игроков, разного пола и возраста.

Важным критерием является платформа, которой чаще всего пользуются игроки. Так самое популярное устройство для игр — [компьютер или ноутбук](https://www.statista.com/forecasts/1002821/video-game-usage-by-device-in-russia). Второе место с небольшим отрывом заняли смартфоны (Один человек может использовать сразу несколько устройств, поэтому сумма всех ответов больше 100%). Это позволяет выбрать наиболее удобную платформу для игроков.

Рисунок 1. Статистика пользования разных устройств для игр.

Видеоигры наиболее популярны и более знакомы молодежной аудитории. Так, представители возрастных групп от 18 до 24 лет (40%) и от 25 до 34 лет (25%) чаще других заявляют, что играют в видеоигры. Значит при создании игрового приложения стоит уделить внимание на интересы этих возрастных групп.

Рисунок 2. Статистика пользователей игр в зависимости от возраста

Таким образом, игровое приложение преимущественно будет направлено на людей возрастом 18-34 лет, и будет выпускаться на компьютер или ноутбук.

# Использованные платформы

# Среда разработки Unity

Unity – игровой движок и среда разработки трехмерных и двухмерных игр, а также различных приложений, поддерживающий свыше 25 платформ.

В качестве языков программирования Unity позволяет выбрать между языками C# и JavaScript.

Графический редактор Unity может выполнять весьма ограниченный ряд операций. В нем нельзя моделировать. Исключение составляет базовая работа с примитивами. Однако, Unity без проблем интегрируется со сторонними 3D-редакторами (3D Maya, 3Ds Max, Softimage, CINEMA 4D, Blender и пр.), что означает отсутствие проблем с чтением различных форматов. А после релиза Unity 4.3 стала возможна работа с 2D-графикой, поддерживая спрайты и 2D-физику, благодаря чему с помощью движка можно также создавать графику для 2D-игр.

Редактор Unity имеет простой Drag&Drop интерфейс, который легко настраивать, состоящий из различных окон, благодаря чему можно производить отладку игры прямо в редакторе.

Проект в Unity делится на сцены (уровни) — отдельные файлы, содержащие свои игровые миры со своим набором объектов, сценариев, и настроек. Сцены могут содержать в себе как, собственно, объекты (модели), так и пустые игровые объекты.

К объектам можно применять коллизии (в Unity т. н. коллайдеры — collider). В редакторе имеется система наследования объектов - дочерние объекты будут повторять все изменения позиции, поворота и масштаба родительского объекта. Скрипты в редакторе прикрепляются к объектам в виде отдельных компонентов.

**Преимущества:**

1. Выгодная лицензионная политика (Unity доступен бесплатно, причем базовая версия содержит ровно тот же функционал, что и профессиональная);
2. Легкость в использовании;
3. Поддержка большого количества платформ;

**Недостатки:**

1. Ограниченный набор инструментов (некоторые придется

разработать самостоятельно);

Перед началом разработки игрового приложения был пройден обучающий курс по Unity. Это облегчило понимание самой платформы, процесса создания игры и механик, использующихся при этом.

# Язык программирования C#

C# - это объектно-ориентированный язык программирования. Язык разрабатывался для платформы Microsoft .NET Framework. С# относится к “С” подобным языкам программирования, синтаксис похож на языки Java и C++. Первая бета версия языка С# была выпущена в 2002 году и усовершенствуется по настоящее время. Для Unity актуальна версия 9.0.

Инструментарий C# позволяет решать широкий круг задач, язык действительно очень мощный и универсальный. На нём часто разрабатывают:

* веб-приложения,
* игры,
* мобильные приложения для Android или iOS,
* программы под Windows.

У C# выделяют много **преимуществ:**

1. Поддержка подавляющего большинства продуктов Microsoft
2. Бесплатность ряда инструментов для небольших компаний и некоторых индивидуальных разработчиков.
3. Типы данных имеют фиксированный размер, что повышает «мобильность» языка и упрощает программирование, так как вы всегда знаете точно, с чем вы имеете дело.
4. Автоматическая «сборка мусора». Общеязыковая среда CLR сама вызовет сборщик мусора и очистит память.

Но есть у C# и некоторые **недостатки**:

1. Приоритетная ориентированность на платформу Windows;
2. Язык бесплатен только для небольших фирм, индивидуальных программистов и учащихся.

# Графический редактор gimp

Для создания объектов окружения был использован графический редактор GIMP, который ещё называют бесплатной и упрощенной версией Adobe Photoshop. GNU Image Manipulation Program или GIMP — свободно распространяемый растровый графический редактор, программа для создания и обработки растровой графики и частичной поддержкой работы с векторной графикой.

Данное приложение отлично подходит для создания пиксельной графики.

**Достоинства:**

* GIMP является свободным ПО;
* GIMP является высококачественным приложением для фоторетуши и позволяет создавать оригинальные изображения и для создания экранной и веб-графики;
* GIMP позволяет автоматизировать выполнение повторяющихся действий;
* GIMP легко расширяем за счёт простой установки дополнений.

**Недостатки:**

* нет поддержки плашечных цветов (и палитры Pantone — по лицензионным соображениям);
* нет полноценной поддержки цветовых моделей, CIELAB и CIE XYZ;
* нет поддержки режима 16 и более разрядов на цветовой канал;

# Git

Так же был использован Git для создания удаленного репозитория. Это помогает отслеживать все изменения, которые были сделаны в проекте.

Git — это бесплатная распределенная система управления версиями с открытым исходным кодом, предназначенная для быстрой и эффективной обработки любых проектов, от небольших до очень крупных.

У каждого разработчика, использующего Git, есть свой локальный репозиторий, позволяющий локально управлять версиями. Затем, сохраненными в локальный репозиторий данными, можно обмениваться с другими пользователями.

**Достоинства:**

* 1. Надежная система сравнения ревизий и проверки корректности данных, основанные на алгоритме хеширования SHA1.
  2. Гибкая система ветвления проектов и слияния веток между собой.
  3. Наличие локального репозитория, содержащего полную информацию обо всех изменениях, позволяет вести полноценный локальный контроль версий и заливать в главный репозиторий только полностью прошедшие проверку изменения.
  4. Высокая производительность и скорость работы.
  5. Возможность делать контрольные точки, в которых данные сохраняются без дельта компрессии, а полностью.
  6. Широкая распространенность, легкая доступность и качественная документация.
  7. Гибкость системы позволяет удобно ее настраивать и даже создавать специализированные контроля системы или пользовательские интерфейсы на базе git.
  8. Универсальный сетевой доступ с использованием протоколов http, ftp, rsync, ssh и др.

**Недостатки:**

* 1. Не отслеживается изменение отдельных файлов, а только всего проекта целиком, что может быть неудобно при работе с большими проектами, содержащими множество несвязных файлов.

# Разработка игрового приложения

# Движение персонажа

Во время разработки была реализована механика движения персонажа, управляемого игроком.

Само движение осуществляется посредством изменения координат за единицу времени на определенное число, зависящее от нажатых кнопок и скорости:

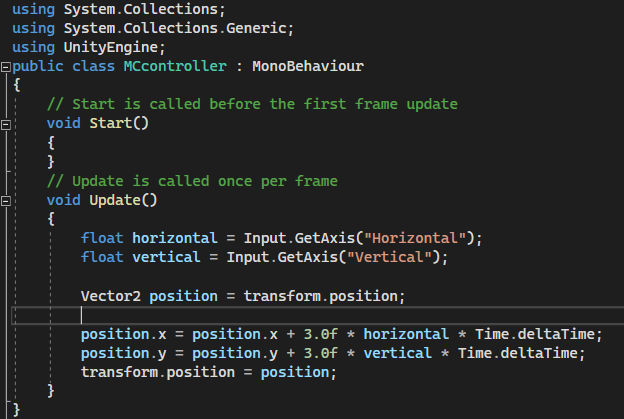
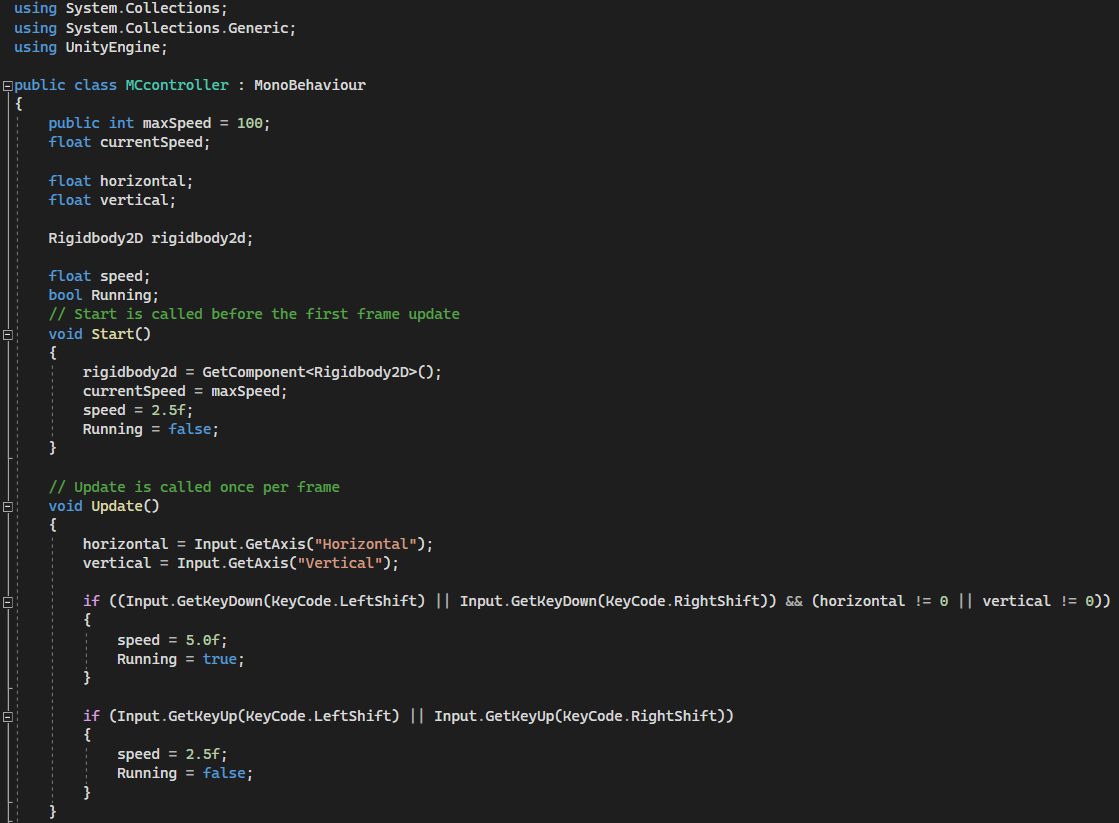


Рисунок 3. Программа отвечающая за движение персонажа

Скорость задается отдельно, так как персонаж может ускоряться, если зажать shift, но при этом ускорение не может длиться вечно, так как это не является реалистичным, поэтому были добавлены ограничения выносливости:



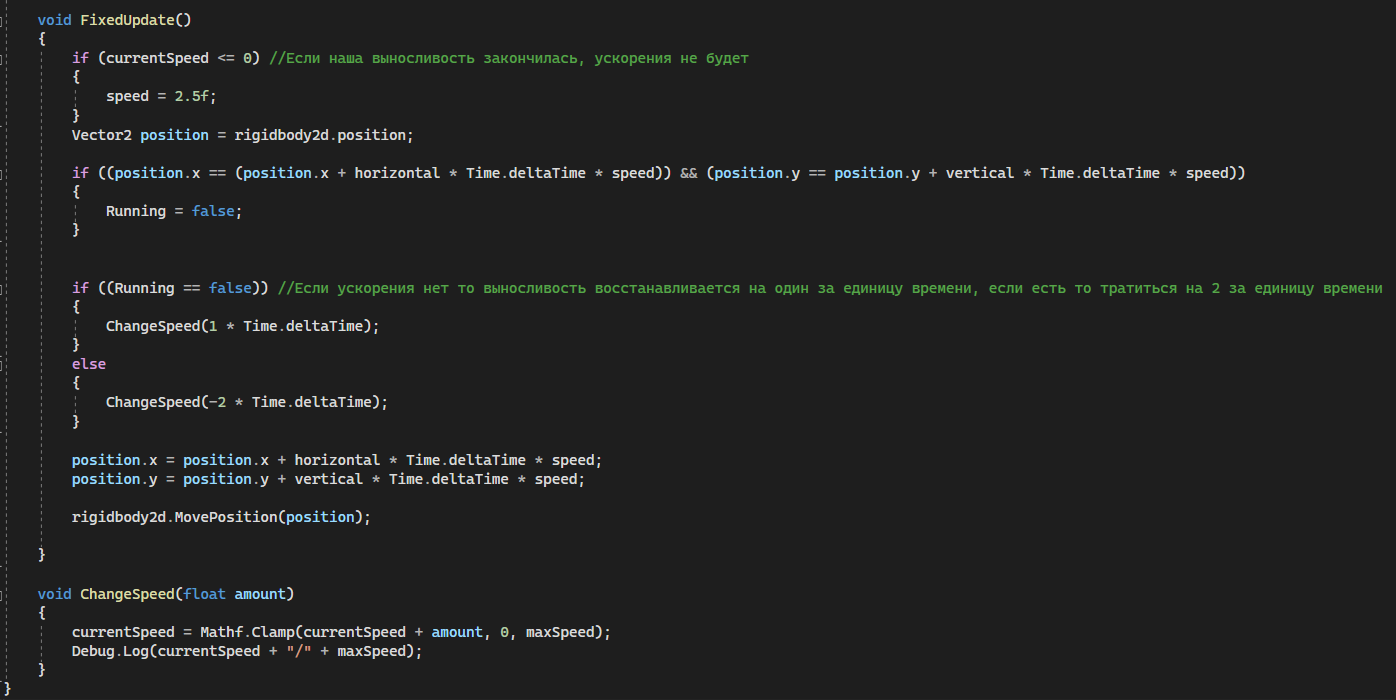


Рисунок 4 и 5. Движение персонажа с ускорением.

На рисунке 6 часть кода отвечает за то, что при нажатии Shift мы получаем ускорение, но в этом месте возникла проблема. Если зажать shift до начала движения, то персонаж получит ускорение и выносливость будет убывать даже когда персонаж стоит. Для решения проблемы в условие было добавлено (&& (horizontal != 0 || vertical != 0), то есть ускорение мы получаем только если мы одновременно движемся и зажимаем shift.

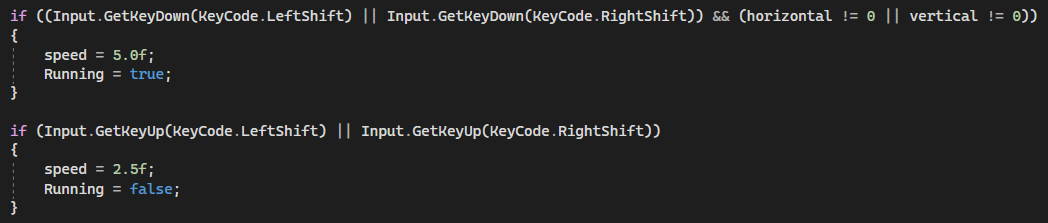


Рисунок 6.

# Окружающая среда

Чтобы игрок не мог проходить сквозь предметы, персонажу было добавлено «твердое тело» (rigidbody). А также всем объектам и самому герою были добавлены коллайдеры. Благодаря этому персонаж не может пройти сквозь предмет, так как их коллайдеры столкнутся.

Так как игра двумерная, мы не можем видеть за объектами (рисунок 7), из-за чего передвижение по локации становится затруднительным, поэтому, чтобы было приятнее исследовать локации, объекты, за которые мы зашли или от которых на не большом расстоянии, будут становиться полупрозрачными (рисунок 8).

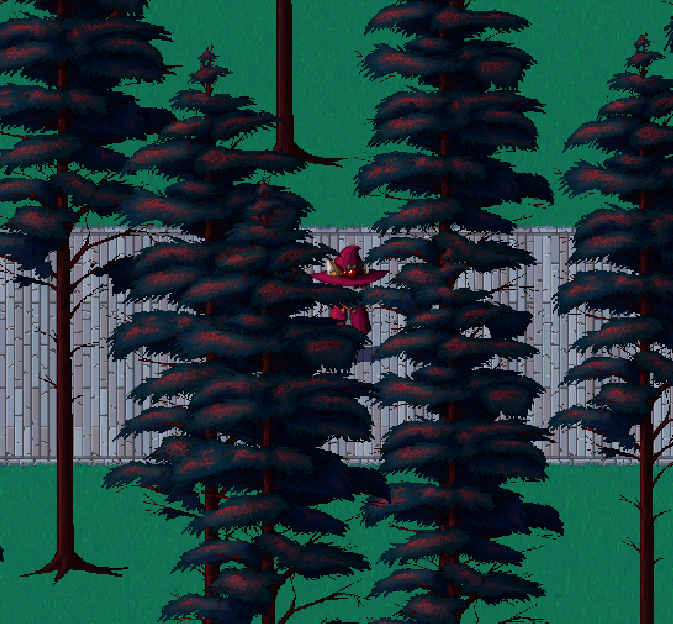


Рисунок 7. За объектами мы видеть не можем

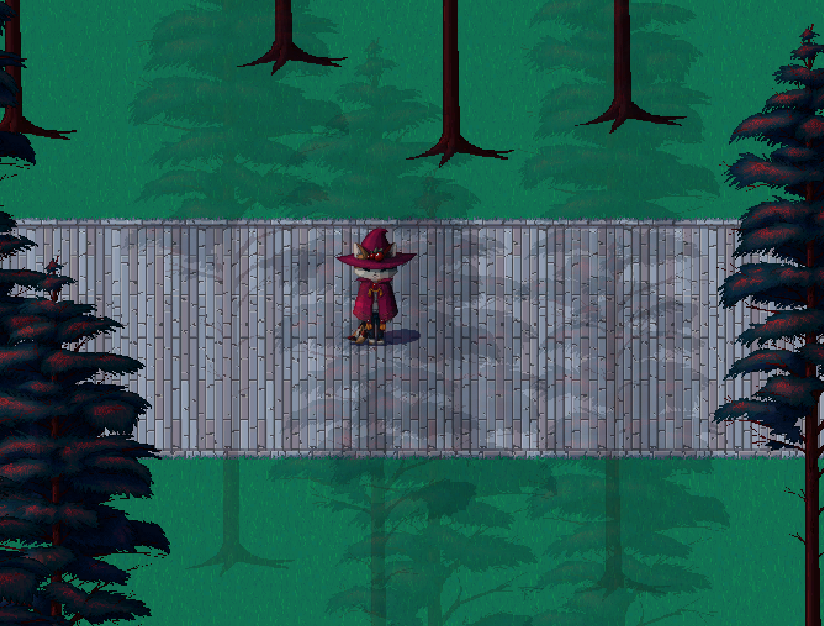


Рисунок 8. Если объекты полупрозрачные

Программа, написанная для этого:

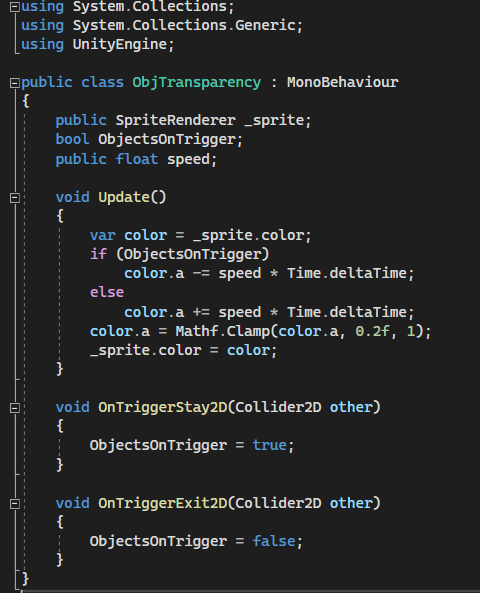


Рисунок 9. Программа, отвечающая за прозрачность объектов.

Здесь возникла проблема, что при выходе из коллайдера-триггера и при входе в него (при нахождении в котором, объект, к которому коллайдер прикреплен, будет оставаться прозрачным) четкость объекта меняется мгновенно, а не постепенно, что не очень приятно для глаз.

Для решения в функции Update было добавлено условие. Если персонаж находится внутри триггера, то предмет постепенно бледнеет (color.a -= speed \* Time.deltaTime), если нет, то восстанавливает свою четкость (color.a += speed \* Time.deltaTime).

# Заключение

За текущий семестр были изучены основные моменты работы с средой разработки Unity. Был пройден курс обучения по данной платформе. Были выполнены задачи, поставленные на этот семестр. Возникшее в процессе проблемы были решены.

На следующий семестр планируются такие задачи:

1. Разработка пользовательского интерфейса.
2. Добавление механик взаимодействия персонажа с миром.
   1. Сбор предметов.
   2. Взаимодействие с враждебными персонажами.
3. Добавление анимации.

Подведя итог, можно сказать, что в следующем семестре планируется больше разработать механику взаимодействия с миром и реализовать front-end составляющую приложения. Таким образом будет продолжена работа над данным проектом в будущем.

# Список источников:

* 1. Видеоигры – и кто в них играет? // wciom.ru [Электронный ресурс] -Режим доступа: <https://wciom.ru/analytical-reviews/analiticheskii-obzor/videoigry-i-kto-v-nikh-igraet> (Дата обращения 21.06.2022)
  2. Обзор рынка игровой индустрии// yandex.ru [Электронный ресурс] -Режим доступа: <https://yandex.ru/adv/solutions/analytics/2019-research-game> (Дата обращения 21.06.2022)
  3. Информация о Unity// cubiq.ru -Юлия [Электронный ресурс] -Режим доступа: <https://cubiq.ru/dvizhok-unity/> (Дата обращения 20.06.2022)
  4. Ruby's Adventure: 2D Beginner // unity.com [Электронный ресурс] -Режим доступа: <https://learn.unity.com/project/ruby-s-2d-rpg?uv=2020.3> (Дата обращения 20.06.2022)
  5. Инструменты разработки: основы для полных новичков // unity.com [Электронный ресурс] -Режим доступа: <https://unity.com/ru/how-to/beginner/game-development-tools#core-mechanics> (Дата обращения 20.06.2022)
  6. Методика разработки игр в Unity// dtf.ru - [Vladimir Afonin](https://dtf.ru/u/149602-vladimir-afonin)

[Электронный ресурс] -Режим доступа: <https://dtf.ru/gamedev/248191-metodika-razrabotki-igr-na-unity> (Дата обращения 20.06.2022)

* 1. Разработка обучающего игрового приложения на платформе Android // studbooks.net [Электронный ресурс] -Режим доступа: <https://studbooks.net/2015844/informatika/analiticheskiy_obzor> (Дата обращения 20.06.2022)
  2. Форум Unity// unity.com [Электронный ресурс] -Режим доступа: <https://forum.unity.com/> (Дата обращения 20.06.2022)
  3. Документация Git// git-scm.com [Электронный ресурс] -Режим доступа: <https://git-scm.com/doc> (Дата обращения 20.06.2022)
  4. Официальный сайт GIMP// gimp.org [Электронный ресурс] -Режим доступа: <https://www.gimp.org/> (Дата обращения 20.06.2022)