|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Информационная безопасность

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОМУ ПРОЕКТУ***

***НА ТЕМУ:***

***Разработка компьютерной 3D игры***

Студент ИУ8-33 **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** Овсепян А. Н.

(Группа) (Подпись, дата) (И.О. Фамилия)

Руководитель курсового проекта **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** Вершинин В. В.

(Подпись, дата) (И.О. Фамилия)

*2020 г.*

Оглавление

[**Введение** 3](#_Toc26805181)

[**Цель работы** 3](#_Toc26805182)

[**Задачи** 3](#_Toc26805183)

[**Ход работы** 4](#_Toc26805184)

[Теоретическая часть 4](#_Toc26805185)

[Практическая часть 8](#_Toc26805186)

[**Результаты** 29](#_Toc26805187)

[**Литература** 30](#_Toc26805188)

# **Введение**

Предметная область курсового проекта включает в себя теоретический материал из области игровой индустрии. Особенности, преимущества и недостатки платформы Unity3D для разработки видеоигр.

Tower Defense — жанр игр, который, несмотря на свою простоту, смог завоевать немалую популярность у современных геймеров. Суть игры сводится к уничтожению волн наступающих противников, которые с каждым новым раундом становятся сильнее и многочисленнее. Устранение врагов происходит с помощью различных пушек, башен, ловушек и прочих аналогичных средств, которые игрок расставляет по уровню на свое усмотрение.

**Цель работы**

Целью данной работы является разработка компьютерной 3D игры в жанре «TD» на платформе Unity3D.

**Задачи**

В данной работе можно выделить следующие поставленные задачи, необходимые для достижения поставленной цели:

1) Выполнить анализ требований и разработать внешние спецификации;

2) Выполнить проектирование игрового приложения;

3) Реализовать игровое приложение;

4) Протестировать разработанное игровое приложение;

**Ход работы**

Теоретическая часть

[Tower](https://ru.wiktionary.org/wiki/tower#tower_(%D1%81%D1%83%D1%89%D0%B5%D1%81%D1%82%D0%B2%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5)) [Defense](https://ru.wiktionary.org/wiki/defense" \l "%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9" \o "wikt:defense) (с [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) — «Башенная защита»), сокращенно TD — название жанра [компьютерных стратегических игр](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D0%B0%D1%82%D0%B5%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%B8%D0%B3%D1%80%D0%B0). Задача игрока в играх подобного жанра — расправиться с наступающими врагами, называемыми в некоторых играх «крипы» (от [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) [creep](https://ru.wiktionary.org/wiki/creep" \l "creep_(%D1%81%D1%83%D1%89%D0%B5%D1%81%D1%82%D0%B2%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5)" \o "wikt:creep) «ползучая тварь»), до того, как они пересекут карту, с помощью строительства башен, атакующих их, когда те проходят вблизи. Противники и башни обычно различаются по характеристикам и цене. Когда враги побеждены, игрок зарабатывает деньги или очки, которые используются для покупки или модернизации башен.

Подбор вида башен и их расположение — неотъемлемая стратегия игры. Обычно ползучие твари пробегают через подобие [лабиринта](https://ru.wikipedia.org/wiki/%D0%9B%D0%B0%D0%B1%D0%B8%D1%80%D0%B8%D0%BD%D1%82), что дает игроку возможность стратегического размещения башен, но также существуют известные версии игры, названные линейными TD, где используются прямые пути вместо лабиринтов. В некоторых версиях игрок может сам выстраивать лабиринт из башен и блоков.

Рисунок 1.1 - Пример игры в жанре TD

Теперь стоит поговорить о платформе, на которой реализуется описанное выше приложение. Unity3D – это среда для разработки игр для различных платформ, в которой объединены различные программные средства, используемые при создании ПО – текстовый редактор, компилятор, отладчик и т.д. Unity3D является простым и комфортным средством для разработки, а мультиплатформенность позволяет разработчикам распространить свое приложение на большое количество игровых платформ и операционных систем.

Как у любого продукта, у Unity есть свои положительные аспекты и отрицательные. К преимуществам можно отнести:

1. Возможность разработки игр, не требуя для этого каких-либо особых знаний. Здесь используется компонентно-ориентированный подход, в рамках которого разработчик создает объекты и к ним добавляет различные компоненты.
2. Наличие большой библиотеки различных ресурсов и программных модулей, с помощью которых можно значительно ускорить игровой процесс.
3. Поддержка большого количества платформ, технологий, API. Unity3D поддерживает DirectX и OpenGL, работает со всеми современными эффектами отрисовки, включая новейшую технологию трассировки лучей в реальном времени.
4. Физика твердых тел, Ragdoll и тканей, система Level of Detail, коллизии между объектами, сложные анимации – все это можно реализовать силами игрового ПО.
5. Unity3D доступен бесплатно, что открывает перед независимыми разработчиками дверь в игровую индустрию.

К негативным сторонам можно отнести следующие пункты:

1. Если команда захочет разработать что-нибудь сложнее простого платформера, то ей придется искать хорошего программиста на C#, который напишет интерпретируемые файлы и компоненты, внедрит их в игру и заставит работать.
2. Создание масштабных, сложных сцен с множеством компонентов может негативно повлиять на производительность игры, в результате чего разработчикам придется потратить дополнительное время и ресурсы на оптимизацию.
3. Приложения, созданные на Unity, довольно «тяжеловесны»: даже самая простая пиксельная игра может занимать несколько сотен мегабайт на ПК.

Практическая часть

Перед объяснением процесса реализации курсового проекта, следует разобраться в основных моментах разработки на Unity. Для начала, давайте разберемся, как создаются и настраиваются объекты на сцене? Для начала нам нужно создать пустой объект на сцене (gameObject) как на рисунке 2.1

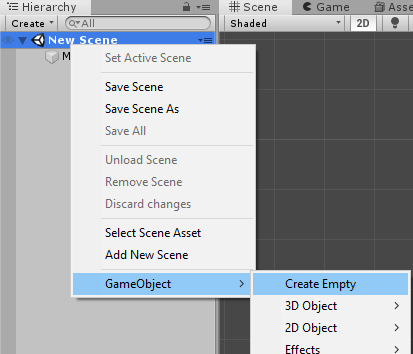


Рисунок 2.1 - создание нового объекта на сцене

Дальше настройка объекта происходит в окне Inspector. Теперь посмотрим реализацию объекта «Cannon» (Одна из башен), рассмотрим его особенности на сцене. На рисунках 2.2.1 и 2.2.2 можно посмотреть пример окна инспектора для «Cannon».

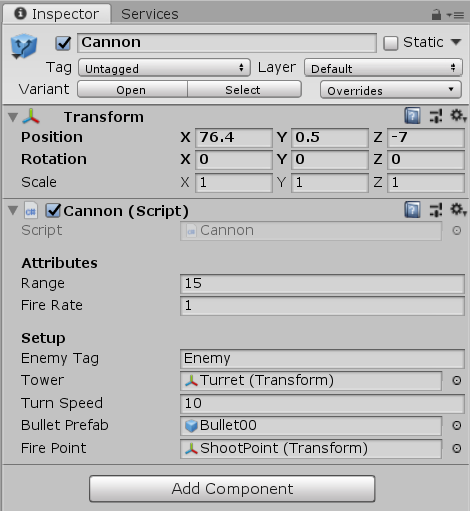


Рисунок 2.2 - окно Inspector

Именно в инспекторе, с помощью кнопки «Add Component», добавляются компоненты, такие как: физические свойства, интерпретируемые файлы и скрипты. А теперь посмотри на само расположение объекта «Cannon» на сцене (Рисунок 2.3).



Рисунок 2.3 - Объект "Cannon" на сцене

Можно заметить, что для нашего объекта есть еще 2 дочерних, первый это “Turret” он служит для поворота ствола башни, в тот момент, когда второй дочерний объект “Plate” остается не подвижен. Объект “ShootPoint” задает координаты, от куда будет выпущена пуля, в случае стрельбы. Аналогично настраиваются и помещаются на сцене и многие другие объекты

Так же добавлено 2 окна интерфейса, отвечающие за показатели игрока “lowerBar” и “upperBar”. Первый отвечает за количество денег и показывает время между волнами. Второй, в свою очередь, хранит в себе информацию об объекте, такую как количество жизней. А также он обеспечивает привязку к панели игрового интерфейса, который реализован в Canvas, помимо этого, Canvas также дает множество возможностей для реализации игрового меню или оформления дисплея. Например, Canvas, который реализует возможность меню паузы и игрового интерфейса изображен на рисунке 2.4.

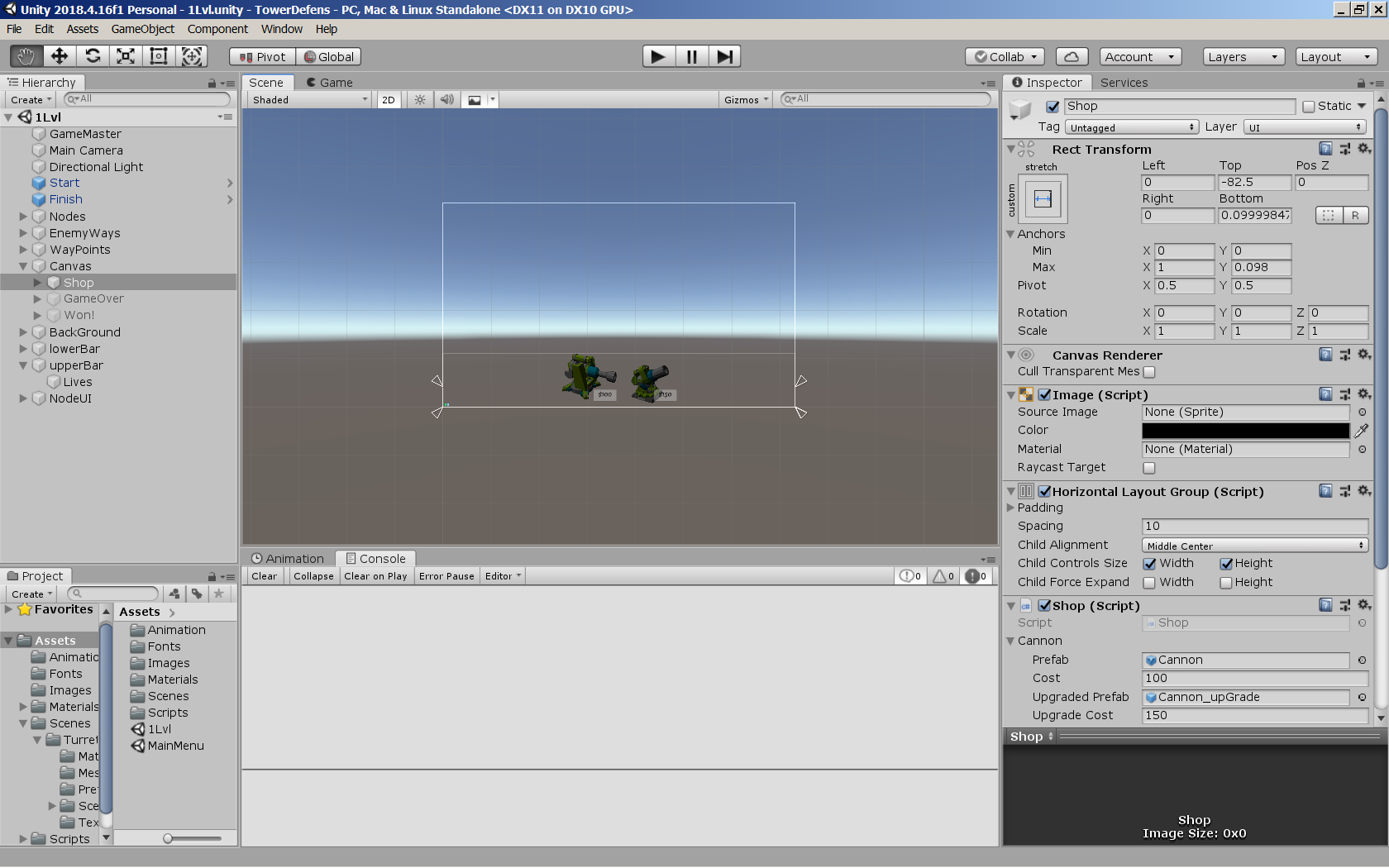


Рисунок 2.4 - Canvas

Также, были созданы анимации на обьекты Canvas. Посмотрим пример создания такой анимации для объекта “GameOver”. Для начала создадим Animator, который отвечает за связь анимаций между собой и Animation Clip (Рисунок 2.5).

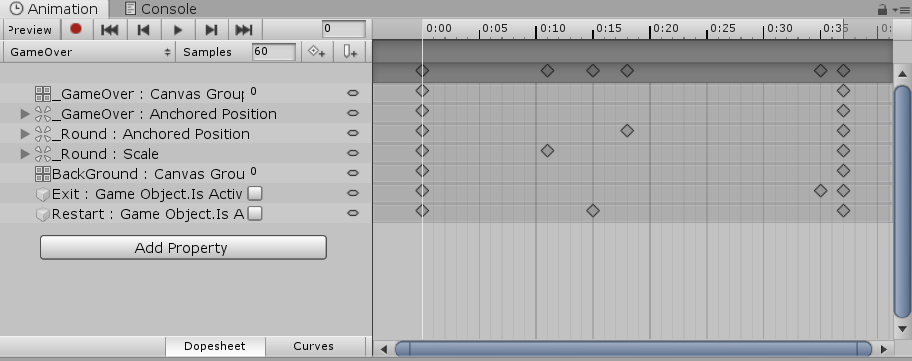


Рисунок 2.5 - Окно для работы с анимацией

Во вкладке “Add Property” можно выбрать любой дочерний объект ”Canvas”. И далее выбрать любой компонент данного обьекта, который мы хотим изменять во временном промежутке.

На рисунке 2.6 можно заметить объект главного меню (MainMenu). Это главное окно которое позволяет при нажатии на “Play” начать игру и при нажати на “Exit” выйти из игры.

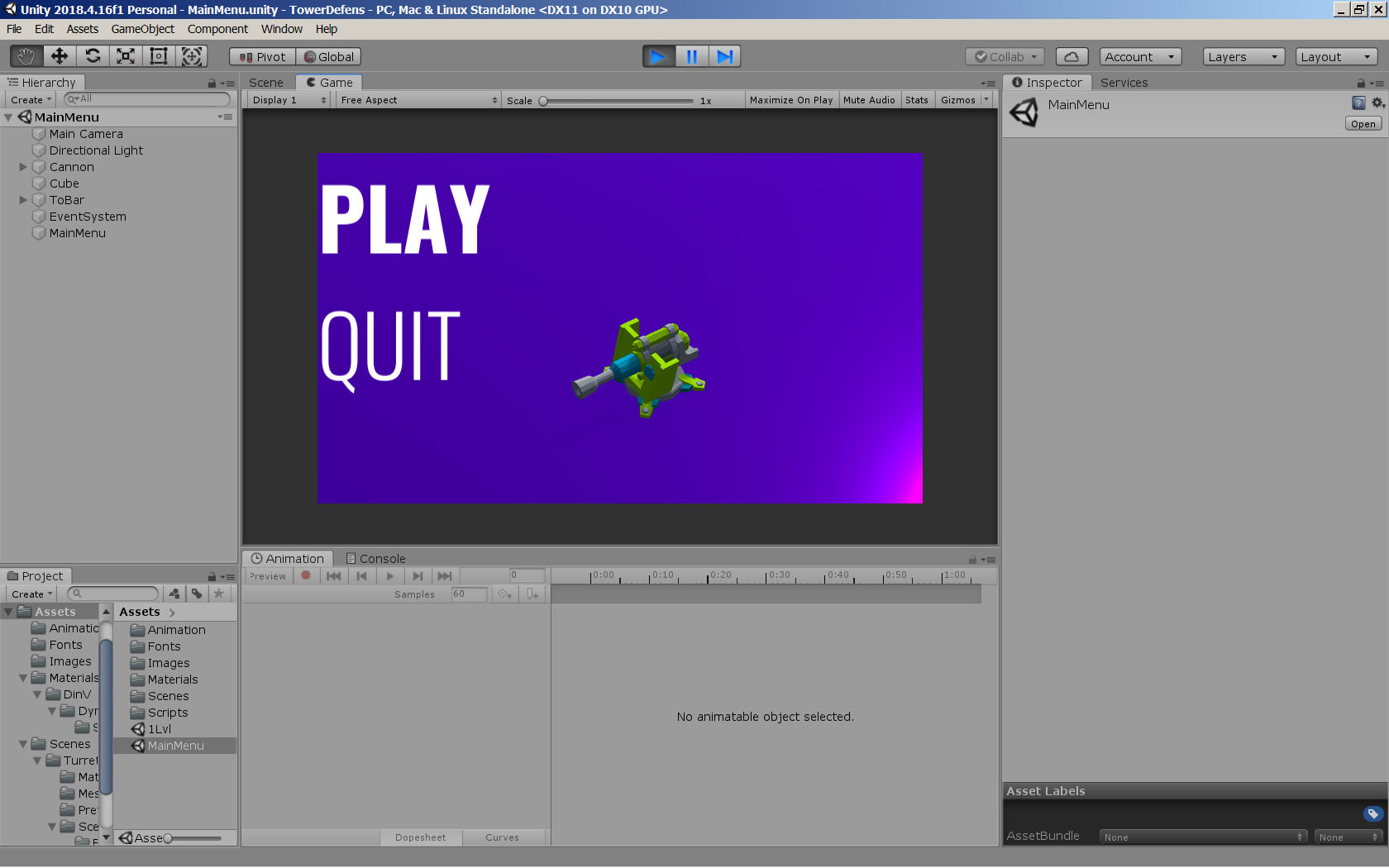


Рисунок 2.6 - Главное меню

Для реализации игровой механики игры было написано 18 интерпретируемых файла на C#, подробнее о них и об их структуре можно посмотреть в документации, сгенерированной с помощью Doxygen, о котором речь пойдет чуть позже.

Следует также сказать о том, как производится сборка приложения в Unity. Для этого надо перейти в панель File - Build Settings и выбрать подходящую платформу, добавить все игровые сцены и настроить параметры по своему усмотрению, после чего нажать кнопку «Build» или «Build and Run» (Рисунок 2.7) и указать директорию, куда соберется игра, потом начнется процесс сборки и на выходе мы получим готовое приложение.

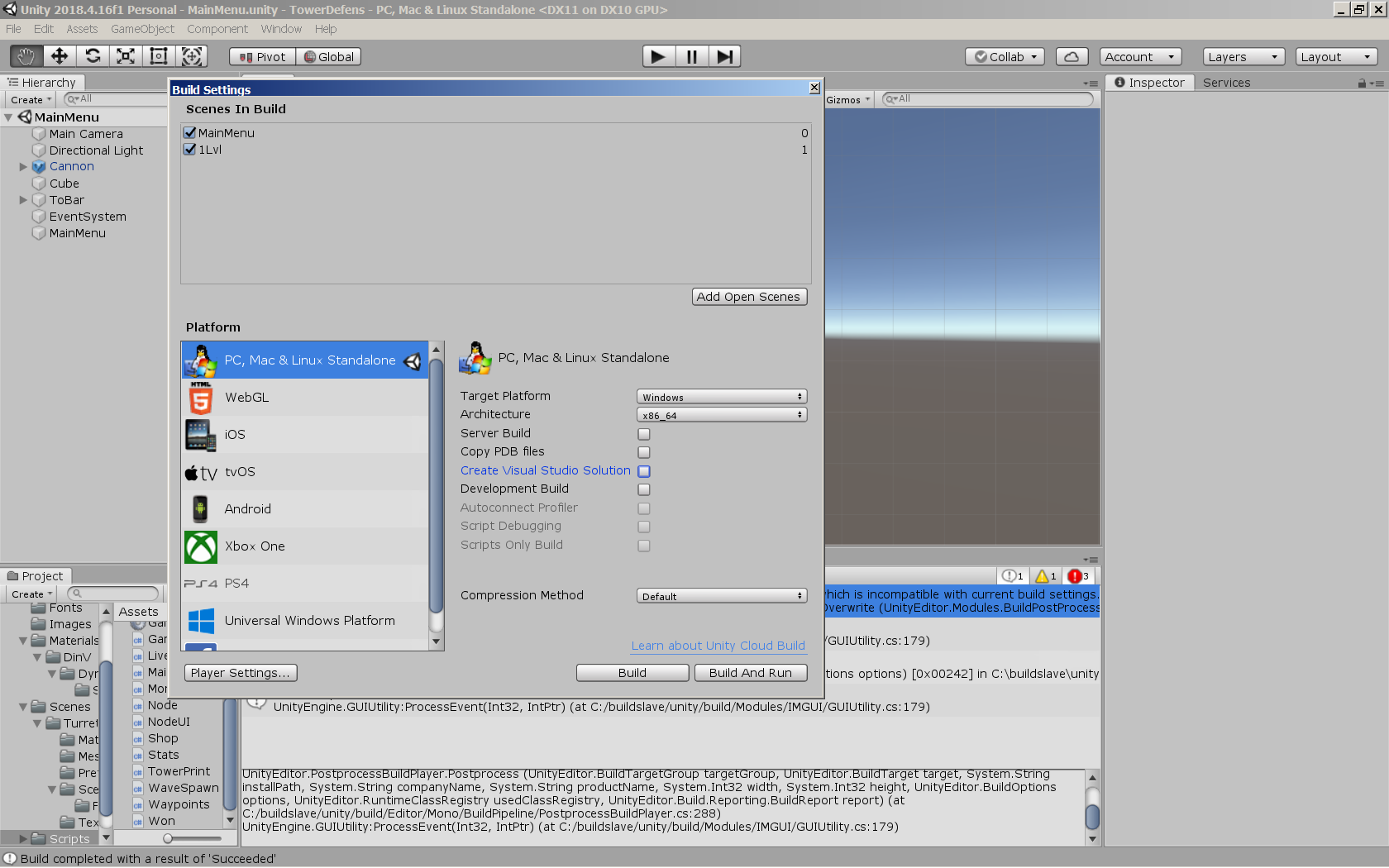


Рисунок 2.7 - Панель Build Settings

**Рисунок 2.8.1** – Панель Build Settings.

Необходимой частью курсовой работы служит применения технологий и методов программирования, соответственно, необходимы нижеперечисленные приемы.

1. Покрытие игры тестами
2. Написание документации с использованием Doxygen
3. Настройка CI/CD для тестирования и непрерывной сборки приложения.

Покрытие игры тестами в Unity происходит с помощью встроенного инструмента «Test Runner» который позволяет выполнять, как и обычные тесты, так и интеграционные. На рисунке 3.1 можно наблюдать панель описанного выше инструмента.

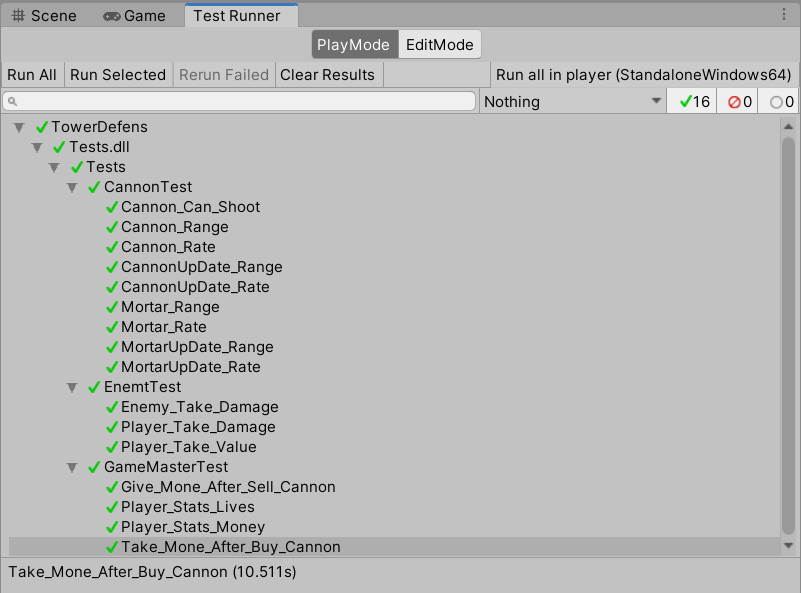


Рисунок 3.1 - утилита Test Runner

Для написания документации, с помощью приложения Doxygen, необходимым условием является комментирование интерпретируемых файлов на C#. Пример такого комментария можно посмотреть на рисунке 3.2.

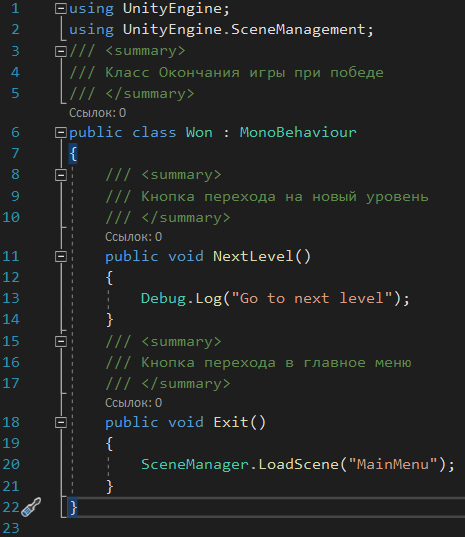


Рисунок 4.2 - пример задокументированного интерпретируемого файла на С#

Пример написанной документации, в которой оформлена главная страница, а также были описаны все поля и методы, можно посмотреть на рисунках 3.3 по 3.5.



Рисунок 3.3 - Список классов в составленной документации



Рисунок 3.4 - Список файлов в составленной документации.

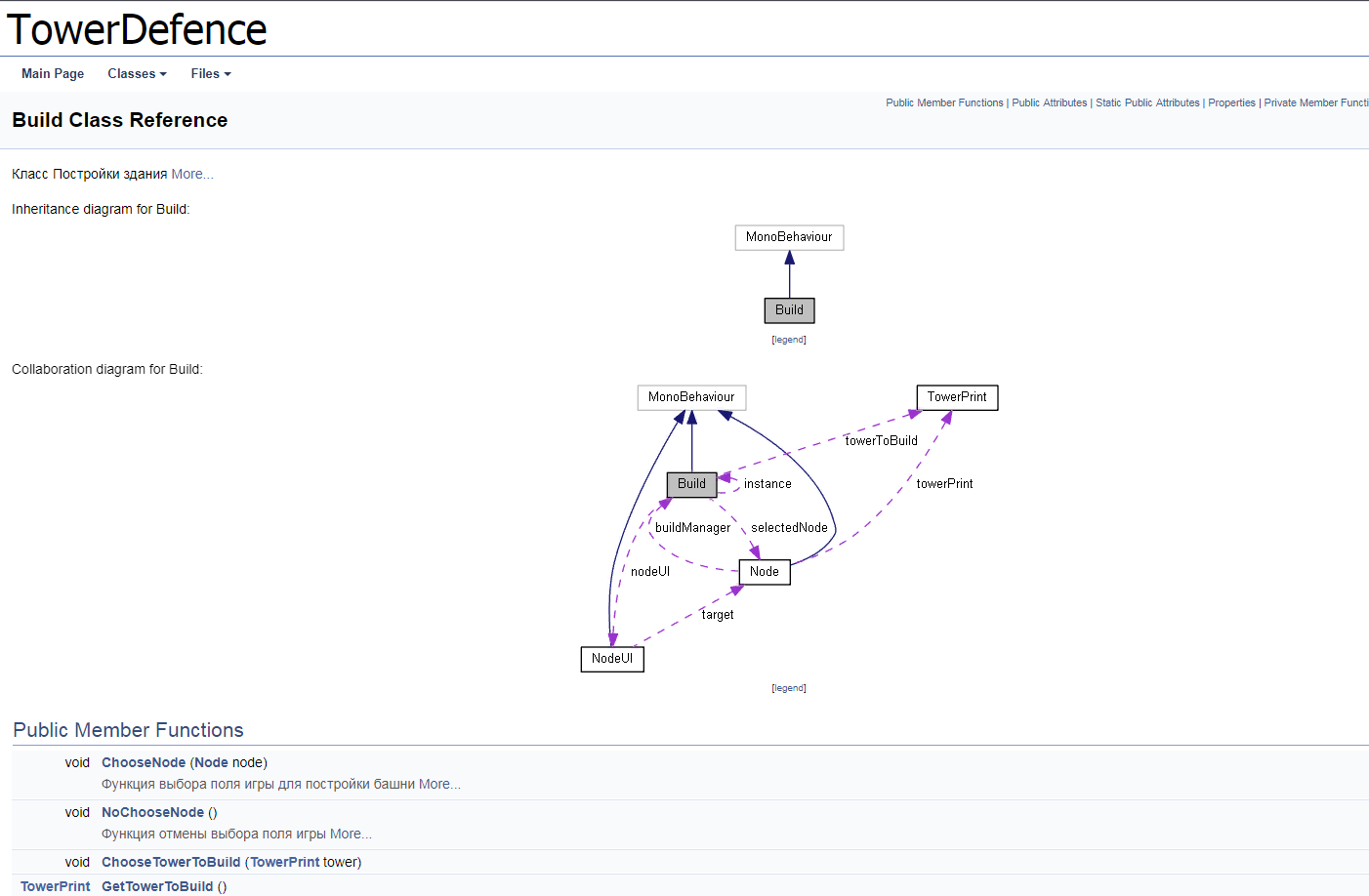


Рисунок 3.5 - Описание класса в составленной документации.

**Результаты**

Подводя итоги, следует обобщить, что было сделано в ходе курсового проекта. Мною была разработана и реализована 3D игра на платформе для разработки видеоигр Unity3D. Написаны и успешно пройдены юнит-тесты и интеграционные тесты для проверки игровых моментов. Была сгенерирована документация с помощью Doxygen, опирающаяся на комментарии в интерпретируемых файлах на C#. Настроена система непрерывной интеграции (CI/CD) на сервисе Unity Cloud Build.

**Литература**

1. Жанр TD URL: <https://ru.wikipedia.org/wiki/Tower_Defense#cite_note-wsj-3> (дата обращения 10.02.2020) – Текст: электронный
2. Движок Unity – особенности, преимущества и недостатки URL: <https://cubiq.ru/dvizhok-unity/> (дата обращения 10.02.2020) – Текст: электронный
3. Функция LookAt. Слежение за объектами URL: <https://3dgame-creator.ru/catalog/uroki/unity5/unity-5-lookat-slezhenie-za-obektami/> (дата обращения 10.02.2020) – Текс: электронный
4. MainMenu URL: <https://habr.com/ru/post/346370/> (дата обращения 10.02.2020) – Текст: электронный
5. Искодные файлы КП URL: <https://developer.cloud.unity3d.com/build/orgs/ara_ovsepyan/projects/towerdefens/>
6. Сгенерированная документация КП URL: https://drive.google.com/open?id=1Hv5FW8LpnGurhD9\_idRwX7q3PCNyvQnY
7. Анимация клипов в Unity3D URL: <https://docs.unity3d.com/ru/530/Manual/AnimationClips.html>