Министерство образования и науки 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 (ИУ) Кафедра «Информационная безопасность» (ИУ8)**

**Отчет**

**по курсовой работе студента**

на тему Разработка 2D-игры

ФИО студента: Еремицкая Дарья Андреевна Группа: ИУ8-32 Курс: 2

Специальность: *10.05.03 01 «Информационная безопасность автоматизированных систем»*

Научный руководитель: Колесников А.В.

Работа выполнена Д. А. Еремицкая

дата подпись студента И. О. Фамилия студента

Допуск к защите А.В. Колесников

дата подпись научного руководителя И. О. Фамилия научного руководителя

Дата защиты

Результаты защиты

Москва 2018

# РЕФЕРАТ

Отчёт 17 с., 4 источника.

# РАЗРАБОТКА 2D ИГРЫ

Объектом исследования является среда разработки Unity.

Цель работы — разработка 2D игры, приобретение навыков работы с Unity.

В процессе работы проводилось изучение спецификации Unity,  
анализ существующих команд.

В результате исследования были получены как представления о структуре Unity, так и новые навыки программирования.

Задачей проекта является написание игры в программе Unity 2D. Название игры: «gonna go».

Описание: Данная игра создавалась c нуля. Смысл игры очень прост, в управлении она не требуется особых усилий. Управление в игре осуществляется с помощью мышки.

Данная игра представляет собой RPG (Ролевые игры) - Класс ролевых игр, которые могут быть как развлекательного, так и учебного характера. Действие игры происходит в придуманном мире, который может быть либо основан на сюжете художественного произведения, либо полностью придуман авторами игры. В этой вымышленной реальности игроку чаще всего (но не обязательно) ставится какая-то цель, которую он должен достичь в ходе игры.

СОДЕРЖАНИЕ

[**РЕФЕРАТ 4**](#_Toc532541414)

[**РАЗРАБОТКА 2D ИГРЫ 4**](#_Toc532541415)

[**ОПРЕДЕЛЕНИЯ 6**](#_Toc532541416)

[**ВВЕДЕНИЕ 9**](#_Toc532541417)

[**ОСНОВНАЯ ЧАСТЬ 10**](#_Toc532541418)

[**Этапы выполнения работы 10**](#_Toc532541419)

[**Архитектура 2D игры 10**](#_Toc532541420)

[**Функционал пакета инструментов Unity 11**](#_Toc532541421)

[Функции событий 11](#_Toc532541422)

[Обычные Update события 11](#_Toc532541423)

[События инициализации 11](#_Toc532541424)

[Создание и уничтожение игровых объектов (GameObjects) 12](#_Toc532541425)

[**Достоинства и недостатки Unity 12**](#_Toc532541426)

[**К недостаткам Unity можно отнести следующее: 13**](#_Toc532541427)

[**Функционал Tiled 13**](#_Toc532541428)

[**О SuperTiled2Unity 14**](#_Toc532541429)

[**Публикация сборок 14**](#_Toc532541430)

[**ЗАКЛЮЧЕНИЕ 16**](#_Toc532541431)

[**АНАЛИЗ ВЫПОЛНЕННОЙ РАБОТЫ 17**](#_Toc532541432)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 18**](#_Toc532541433)

# ОПРЕДЕЛЕНИЯ

В настоящем отчете применяют следующие термины с соответствующими определениями:

**Скриптинг** - необходимая составляющая всех игр. Даже самые простые игры нуждаются в скриптах для реакции на действия игрока и организации событий геймплея. Кроме того, скрипты могут быть использованы для создания графических эффектов, управления физическим поведением объектов или реализации пользовательской ИИ системы для персонажей игры.

**Навигационная система** - позволяет вам создавать персонажей, которые могут разумно перемещаться по игровому миру, используя навигационные сетки, которые создаются автоматически из вашей геометрии сцены. Динамические препятствия позволяют вам изменять навигацию персонажей во время выполнения, а ссылки вне сетки позволяют вам создавать определенные действия, такие как открытие дверей или спрыгивание с уступа. В этом разделе подробно описываются системы навигации и поиска путей Unity.

**Аудиосистема Unity** включает полный трехмерный пространственный звук, микширование и мастеринг в реальном времени, иерархию микшеров, снимки, предопределенные эффекты и многое другое.

**Ресурсы** (Assets) – строительные/составные блоки всех проектов Unity. Игровой движок ссылается на файлы изображений, 3D моделей, звуков и т.д., которые будут использоваться, при создании игры, в качестве ресурсов (assets). Вот почему в любой папке проекта Unity, есть подкаталог, где хранятся все файлы ресурсов, с именем Assets.

В Unity, вы должны думать о **сценах** (scenes), как об игровых уровнях или областях игрового контента (например, меню). Создавая игру с множеством сцен, вы будете иметь возможность распределять время загрузки и тестировать различные части игры в отдельности.

Когда ресурс (asset) используется в сцене игры, он становиться **Игровым Объектом** (Game Object) – в терминологии Unity – особенно в скриптинге – используется термин "GameObject". Все объекты (GameObjects), изначально, имеют хотя бы один компонент(component) – Transform (Преобразование). Transform говорит движку Unity о положении (position), повороте (rotation) и масштабе (scale) объекта, описанных в X, Y, Z координатах (или в случае с масштабом, размерность). К тому же, к этому компоненту можно обратиться из скрипта, чтобы установить его положение (position), поворот (rotation) или масштаб (scale). Добавляя различные компоненты к объекту, мы можем значительно расширить его функционал, для обеспечения реализации любого желаемого сценария игры.

**Компоненты** (components) имеют различное назначение – они могут влиять на поведение, внешний вид и многие другие функции объектов в игре. ‘Прикрепляя’ ('attaching') компонент к объекту вы можете сразу же воспользоваться возможностями предоставляемые этим компонентом. Unity предоставляет множество компонентов различного назначения. Для обеспечения интерактивности различных игровых элементов, вы будите использовать скрипты, которые рассматриваются Unity как компоненты.

Основной концепцией Unity является использование Игровых Объектов (Game Object), но при этом есть ‘умный’ способ хранить объекты как ресурсы (assets), которые будут использованы в различных частях вашей игры, эти ресурсы могут быть заново 'порождены' или 'клонированы' в любое время. При создании сложных объектов с различными компонентами и настройками, будет эффективней создать первоначальный шаблон, из которого и будут созданы эти объекты, при этом для каждого экземпляра возможны индивидуальные настройки и изменения. Рассмотрим в качестве примера – ящик – вы создали его, настроили различные свойства, написали скрипт разрушения, и теперь хотите использовать его многократно, может быть даже в другой игре. **Префаб** (Prefabs –заготовка) — это один из типов ресурсов, предназначений для многократного использования. Префаб может быть вставлен в любое количество сцен и многократно в одну сцену. Когда префаб добавляется в сцену, создаётся его экземпляр. Все экземпляры являются ссылками на оригинальный префаб и фактически его клонами. Независимо от того, как много экземпляров в проекте, при изменении префаба изменяются соответственно и все его экземпляры.

# ВВЕДЕНИЕ

Игровая индустрия на данный момент является одной из самых быстро развивающихся. Зародившаяся в 70х годах она выросла в огромную индустрию развлечений с оборотами в миллиарды долларов. Разработкой игр сейчас занимаются как крупные компании (Blizzard Entertainment, Namco, Activision, Bethesda Softworks, Rockstar Games), так и небольшие фирмы, сообщества и отдельные индиразработчики. Прогресс не стоит на месте, аппаратная часть развивается, у разработчиков появляется больше свободы для творчества и самореализации.

Эта сфера постоянно растет и появляются новые возможности для создания различных видов игр. Наиболее актуальный движок для разработки – Unity.

Курсовая работа посвящена изучению базовых навыков и их практическому применению в среде разработки Unity, развитию творческого мышления и способности представлять всю цельную картину развития событий.

Основной целью курсовой работы является разработка 2D-игры.

Задачи курсовой работы: изучить спецификацию движка Unity, проанализировать существующие команды, разработать 2D-игру, которая относится к типу ролевых

# ОСНОВНАЯ ЧАСТЬ

## **Этапы выполнения работы**

1. Поиск спрайтов.
2. Анимация персонажей.
3. Создание карт (в среде разработки Tiled).
4. Добавление ограничений(коллизия).
5. Передвижение между сценами.
6. Создание врагов.
7. Создание менеджера здоровья для игрока.
8. Показ урона игроку.
9. Программирование NPC.
10. Диалоговые окна.
11. Добавление звуковых эффектов.

## **Архитектура 2D игры**

1. Ассеты
   1. Анимация
   2. Рисунки
   3. Карты
   4. Материалы
   5. Сцены
   6. Скрипты
   7. SuperTiled2Unity
2. Объекты
3. Пакеты
4. Настройки проекта
5. Сборки проекта

## **Функционал пакета инструментов Unity**

### **Функции событий**

Скрипт в Unity не похож на традиционную идею программы, где код работает постоянно в цикле, пока не завершит свою задачу. Вместо этого, Unity периодически передаёт управление скрипту при вызове определённых объявленных в нём функций. Как только функция завершает исполнение, управление возвращается обратно к Unity. Эти функции известны как функции событий, т.к. их активирует Unity в ответ на события, которые происходят в процессе игры. Unity использует схему именования, чтобы определить, какую функцию вызвать для определённого события. Например, вы уже видели функцию Update (вызывается перед сменой кадра) и функцию Start (вызывается прямо перед первым кадром объекта). В Unity доступно гораздо большее количество функций событий.

### **Обычные Update события**

Игра — это практически тип анимации, в которой кадры генерируются на ходу. Ключевой концепт в программировании игр заключается в изменении позиции, состояния и поведения объектов в игре прямо перед отрисовкой кадра. Такой код в Unity обычно размещают в функции Update. Update вызывается перед отрисовкой кадра и перед расчётом анимаций.

### **События инициализации**

Зачастую полезно иметь возможность вызвать код инициализации перед любыми обновлениями, происходящими во время игры. Функция Start вызывается до обновления первого кадра или физики объекта. Функция Awake вызывается для каждого объекта в сцене в момент загрузки сцены. Учтите, что хоть для разных объектов функции Start и Awake и вызываются в разном порядке, все Awake будут выполнены до вызова первого Start. Это значит, что код в функции Start может использовать всё, что было сделано в фазе Awake.

### **Создание и уничтожение игровых объектов (GameObjects)**

Некоторые игры имеют постоянное количество объектов на сцене, однако обычно персонажи, сокровища и другие объекты создаются и удаляются во время игры. В Unity, игровой объект (GameObject) может быть **создан,** используя функцию Instantiate, которая делает копию существующего объекта

Также есть функция Destroy, которая уничтожит объект после того, как загрузка кадра будет завершена или опционально после короткой паузы.

## **Достоинства и недостатки Unity**

По сравнению с другими движками, Unity имеет следующие

достоинства:

* Простота использования и производительность;
* Огромное число поддерживаемых платформ: Windows, iOS, Android;
* Возможность запуска проекта в браузере как HTML приложения;
* Хорошая производительность как в небольших играх на слабых мобильных платформах, так и в сложных больших проектах на high end консолях;
* Поддержка как 2D, так и 3D режимов без особых усилий;
* Мощная встроенная анимационная система Mechanim;
* Язык программирования С# или JavaScript;
* Большое сообщество, где можно получить поддержку и ответы на интересующие вопросы по Unity;
* Встроенный визуальный редактор, сильно упрощающий и ускоряющий процесс создания приложения;
* Assets store, магазин для разработчиков, открывающий доступ к огромному числу моделей, текстур, скриптов и дополнений для Unity;
* Возможность использования абсолютно бесплатной среды разработки
* MonoDevelop идущей в комплекте с Unity, либо внешней IDE (например, Microsoft Visual Studio или IntelliJ IDEA);

## **К недостаткам Unity можно отнести следующее:**

* Исходный код Unity закрыт и доступен только в случае покупки Enterprise лицензии. В случае каких-либо проблем, придется ждать обновления и исправлений, что иногда занимает довольно много времени;
* Отсутствие такого привычного понятия как “game loop” или игрового цикла;
* Нет единой точки входа, как в других движках;
* Непривычный и местами сложный в освоении визуальный редактор, который может оказаться проблемой для начинающих разработчиков;

## **Функционал Tiled**

Tiled — кроссплатформенный открытый редактор тайловых карт для игр. Он позволяет создавать карты для 2-мерных игр (с видом сбоку, таких, как платформеров, или видом сверху, к примеру JRPG). Начиная с версии 0.11.0 поддерживаются гексагональные тайлы. Tiled был использован при разработке достаточно большого числа игр, как свободных, так и проприетарных. Сохраняет уровни в основанном на XML формате .TMX, чтение которого обеспечивает ряд библиотек для различных языков программирования и такие игровые движки, как Construct 2.

## **О SuperTiled2Unity**

SuperTiled2Unity — это пакет Unity, который дает возможность импортировать карты редактора Tiled Map непосредственно в проекты Unity.

SuperTiled2Unity использует импортеров сценариев, доступных в Unity 2018 и более поздних версиях, так что процесс импорта полностью содержится в вашем проекте Unity на всех платформах. Обновления производятся автоматически по мере внесения изменений в карту в Tiled и их сохранения. Нет необходимости в отдельном процессе экспорта.

## **Публикация сборок**

В любой момент разработки игры можно посмотреть на то, как она выглядит вне редактора, при сборке в качестве самостоятельного приложения или веб-проигрывателя.

Пункт меню File->Build Settings… позволяет открыть окно Build Settings. В нём выводится редактируемый список сцен для включения в сборку игры.

Когда вы готовы к публикации своей сборки, нужно выбрать нужную платформу в списке Platform и удостовериться, что напротив неё находится логотип Unity; если это не так, тогда нажать кнопку Switch Platform, чтобы сообщить Unity о том, под какую платформу нужно совершить сборку. После этого, нужно нажать кнопку Build. Откроется стандартное диалоговое окно сохранения файла, в котором можно выбрать имя и расположение для игры. После нажатия кнопки Сохранить Unity соберёт само приложение.

**ЗАКЛЮЧЕНИЕ**

Поставленные цели курсовой работы были достигнуты, Unity и его команды были изучены. Данный проект может быть внедрен в пример базового создания игры 2D. Все поставленные задачи были решены.

Курсовая может быть полезна для всех интересующихся разработкой игр и желающих обучиться навыкам работы в Unity.

В процессе курсовой работы была спроектирована и разработана игра с использованием кроссплатформенного движка Unity. Данная игра не является законченным продуктом, так как требует наполнения профессионально выполненным графическим контентом, но при соответствующей доработке может занять свою нишу на рынке игр. Движок Unity показал себя как удобный, простой в освоении и профессионально выполненный инструмент для создания игр.

# АНАЛИЗ ВЫПОЛНЕННОЙ РАБОТЫ

Были освоены базовые знания Unity и создана игра на примитивном уровне, исходя из поставленных задач и порядка выполнения.

В будущем в данной игре можно добавить:

* систему атаки врагов игроком
* анимацию атаки врагов с использованием предметов(оружия)
* цифры урона как при ударе игроком врагов, так и при ударе самого игрока
* квесты, которые можно получить, взаимодействуя с неиграбельными персонажами
* наладить систему взаимодействия с неиграбельными персонажами и систему отображения диалоговых окон
* звуковые эффекты при взаимодействии с другими персонажами и локациями
* денежную систему и магазины
* начальное меню при загрузке игры с настройками
* создать систему, которая при достижении здоровья игрока нулевого значения, переносит игрока в начало уровня

# 

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Официальный сайт руководства Unity  
   <https://docs.unity3d.com/ru/current/Manual/AnimationGlossary.html>

<https://docs.unity3d.com/Manual/index.html>

1. О функционале SuperTiled2  
   <https://unitylist.com/p/g0b/Super-Tiled-2-Unity>
2. О функционале Tiled  
   <https://ru.wikipedia.org/wiki/Tiled>