Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовому проекту
на тему

Игра-платформер «Shrek»

Студент гр. 953502
М.С.Криштапович

Руководитель ассистент кафедры информатики
И.А.Удовин

Минск 2020

Оглавление

Введение

Компьютерная игра – компьютерная программа, служащая для организации игрового процесса (геймплея), связи с партнёрами по игре, или сама выступающая в качестве партнёра.

Составляющие компьютерной игры:

- Сеттинг – это среда, в которой происходит действие компьютерной игры; место, время и условия действия.
- Геймплей – компонент игры, отвечающий за интерактивное взаимодействие игры и игрока. Геймплей описывает, как игрок взаимодействует с игровым миром, как игровой мир реагирует на действия игрока и как определяется набор действий, который предлагает игроку игра.
- Музыка – любые мелодии, композиции или саундтреки.

Компьютерные игры могут по праву называться чудом двадцатого века. Частью поп-культуры они стали в конце 1970-ых годов. В современном мире компьютерные игры стали не только развлечением, но и носителем культуры. Так в 2011 году компьютерные игры официально признаны отдельным видом искусства в США, а гейм-разработчики наравне с представителями кинематографа, музыки, литературы могут претендовать на гранты от государства.

Компьютерные игры оказали столь существенное влияние на общество, что в информационных технологиях отмечена устойчивая тенденция к геймификации для неигрового прикладного программного обеспечения[1].

1.Анализ предметной области

1.1 Понятие платформер

Платформер – жанр компьютерных игр, в которых основной чертой игрового процесса является прыгание по платформам, лазанье по лестницам и тд. Игры подобного жанра характеризуются нереалистичностью, рисованной мультяшной графикой. Героями таких игр обычно бывают мифические существа (к примеру драконы, гоблины) или антропоморфные животные. Платформеры появились в начале 1980-х, когда игровые консоли не были достаточно мощными, чтобы отображать трехмерную графику или видео. Они были ограничены статическими игровыми мирами, которые помещались на один экран, а игровой герой был виден в профиль. Через некоторое время после образования жанра у него появилось данное название, отражающее тот факт, что в платформерах геймплей сфокусирован на прыжках по платформам. В некоторых играх данного жанра реализованы различные механики, расширяющие игровой процесс, такие как сбор монет или других предметов, стрельба, наличие предметов с усилениями для персонажа(высокий прыжок, ускорение, неуязвимость), противники(соприкосновение с ним отнимает жизненные силы у героя или вовсе убивает его), наделенные примитивным искусственным интеллектом. Наиболее известные платформеры: Super Mario, Donkey Kong, Rayman, Sonic, Prince of Persia, Ratchet and Clank, Crash Bandicoot, Little Big Planet[1].

1.2 Персонаж «Шрек»

Шрек (англ. *Shrek*) – вымышленный огр, персонаж детской книги Уильями Стейга «Шрек!», а также снятой по её мотивам

популярной серии анимационных фильмив, сопутствующих фильмам компьютерных игр, комиксов, наборов стикеров и т. д.

Кожа зеленого цвета, уши трубочкой, рост 180 см, вес 160 кг, носит бежевую рубашку, коричневую жилетку, коричневые штаны в клетку и ботинки(рис. 1). У Шрека был реальный прототип, Уильям Стейг срисовал его с известного французского рестлера Мориса Тийе, у которого была акромегалия(рис. 2)[1].

Рис. 1 – Шрек

Рис. 2 – Морис Тийе

1.3 Выбор языка и среды разработки

Для написания этой курсовой работы выбран язык C#.

C# – мультипарадигмальный язык программирования. Разработан в 1998—2001 как язык разработки приложений для платформы Microsoft .NET Framework. C# относится к семейству языков с C-подобным синтаксисом, из

них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

Поддерживает такие парадигмы программирования, как объектно-ориентированное программирование, процедурное программирование, событийное программирование, обобщённое программирование.

Интегрированная Среда Разработки (ИСР) − среда, в которой есть все необходимое для проектирования, запуска и тестирования приложений, где все нацелено на облегчение процесса создания программ.

Для разработки программ на языке C# будет использована среда программирования Microsoft Visual Studio 2019, игровой движок Unity 2019.3.13a1 Personal.

Игровой движок – основное ядро игры, базовое программное обеспечение, на основе которого строятся все остальные составляющие игры. Игровой движок содержит в себе:

- Игровую логику
- Физику объектов
- Правила отрисовки объектов

С помощью игрового движка Unity разрабатывается огромное количество игр все самые популярные платформы. Unity является самым популярным игровым движком из всех существующих, в 2019 году количество игр на этом движке превысило 28 миллиардов. Это обуславливается следующими факторами:

- Можно использовать бесплатно. На начальном этапе включает в себя всё, что нужно для создания полноценных игр.

- Кроссплатформенность. На Unity можно создавать игры под любые платформы будь то Windows, Linux, Mac или IOS, Android, Windows Phone, проекты можно запустить даже в браузере.
- Доступный язык C# прост в изучении, поэтому можно создать первую игру за относительно короткое время
- Гибкость. На Unity можно создавать как 2D, так и 3D игры
- Встроенный магазин спрайтов позволяет на тратить много времени на создание текстур, декораций и музыки к игре

На Unity созданы такие игры как Pokemon Go, Beat Saber, HearthStone, Firewatch, Temple Run, Monumanet Valley.

Графический редактор – программа (или пакет программ), позволяющая создавать, просматривать, обрабатывать и редактировать цифровые изображения (рисунки, картинки, фотографии) на компьютере. Для создания спрайтов(визуальных элементов) в стиле векторной графики будет использовать графический редактор Adobe Illustrator 2019[1].

1.4 Постановка задачи

В задачу курсового проекта входит разработать компьютерную игру для операционной системы Windows 10 64-битной архитектуры на языке программирования C#, с возможностью поддержки геймпада DualShock 4, через проводное подключение.

2. Разработка компьютерной игры

2.1 Разработка основного алгоритма программы

Основной алгоритм программы написан на языке C#, который поддерживает объектно-ориентированное программирование.

Объектно-ориентированное программирование (ООП) – методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования.

Идеологически ООП – подход к программированию как к моделированию информационных объектов, решающий на новом уровне основную задачу структурного программирования: структурирование информации с точки зрения управляемости, что существенно улучшает управляемость самим процессом моделирования, что, в свою очередь, особенно важно при реализации крупных проектов.

Управляемость для иерархических систем предполагает минимизацию избыточности данных (аналогичную нормализации) и их целостность, поэтому созданное удобно управляемым – будет и удобно пониматься. Таким образом, через тактическую задачу управляемости решается стратегическая задача – транслировать понимание задачи программистом в наиболее удобную для дальнейшего использования форму.

Основные принципы структурирования в случае ООП связаны с различными аспектами базового понимания предметной задачи, которое требуется для оптимального управления соответствующей моделью:

- абстрагирование для выделения в моделируемом предмете важного для решения конкретной задачи по предмету, в конечном счёте контекстное понимание предмета, формализуемое в виде класса;

- инкапсуляция для быстрой и безопасной организации собственно иерархической управляемости: чтобы было достаточно простой команды «что делать», без одновременного уточнения как именно делать, так как это уже другой уровень управления;
- наследование для быстрой и безопасной организации родственных понятий: чтобы было достаточно на каждом иерархическом шаге учитывать только изменения, не дублируя всё остальное, учтённое на предыдущих шагах;
- полиморфизм для определения точки, в которой единое управление лучше распараллелить или наоборот – собрать воедино.

То есть фактически речь идёт о прогрессирующей организации информации согласно первичным семантическим критериям: «важное/неважное», «ключевое/подробности», «родительское/дочернее», «единое/множественное». Прогрессирование, в частности, на последнем этапе даёт возможность перехода на следующий уровень детализации, что замыкает общий процесс. Обычный человеческий язык в целом отражает идеологию ООП, начиная с инкапсуляции представления о предмете в виде его имени и заканчивая полиморфизмом использования слова в переносном смысле, что в итоге развивает выражение представления через имя предмета до полноценного понятия-класса.

Поэтому программа разделена на классы:

- Camera
- Check
- Pause
- Death
- Finish
- DualShock

- Play
- Shrek
- Transmit
- Traps

2.2 Реализация основных блоков кода

Каждый класс программы находится в отдельном скрипте(файле)[2].

2.2.1 Класс Camera

В данном классе реализована логика перемещения камеры, а именно движение вслед за персонажем и ограничения на выход за пределы игровой сцены.

2.2.2 Класс Check

Этот класс проверяет подключен ли к устройству геймпад и исходя из этого подстраивает начальное меню(Рис. 3(а, б)), меню паузы(Рис. 4(а, б)), меню смерти(Рис. 5(а, б)) и финальное меню(Рис. 6(а, б)) под конкретную ситуацию.

Рис. 3(а)

Рис. 3(б)

Рис. 4(а)

Рис. 4(б)

Рис. 5(а)

Рис. 5(б)

Рис. 6(а)

Рис. 6(б)

```
public class Check : MonoBehaviour
{
    public GameObject upperButton;
    public GameObject lowerButton;
    public GameObject upperImage;
    public GameObject lowerImage;

    void Update()
    {
        try
        {
            if (DualShock4GamepadHID.current.enabled)
            {
                Cursor.visible = false;
                upperButton.SetActive(false);
                lowerButton.SetActive(false);
                upperImage.SetActive(true);
                lowerImage.SetActive(true);
            }
        }
        catch (NullReferenceException)
        {
            Cursor.visible = true;
            upperButton.SetActive(true);
            lowerButton.SetActive(true);
            upperImage.SetActive(false);
            lowerImage.SetActive(false);
        }
    }
}
```

2.2.3 Класс Pause

Этот класс запускает меню паузы, когда нажата клавиша Esc или кнопка Options на геймпаде, наделяет функционалом кнопки Resume и Quit(Рис. 4(а)) и останавливает музыку.

2.2.4 Класс Death

Этот класс запускает меню смерти, когда персонаж соприкасается с определенными объектами(Рис. 13), наделяет функционалом кнопки Restart и Quit(Рис. 5(а)) и останавливает музыку.

```
public class Death : MonoBehaviour
{
    public GameObject deathMenu;
    AudioSource song;


    void Start()
    {
        deathMenu.SetActive(false);
        song = GetComponent<AudioSource>();
    }

    void Update()
    {
        if (deathMenu.activeInHierarchy == true)
        {
            song.Stop();
        }
    }

    public void RestartGame()
    {
        SceneManager.LoadScene(SceneManager.GetActiveScene().name);
        Time.timeScale = 1f;
        song.Play();
        deathMenu.SetActive(false);
    }

    public void QuitGame()
    {
        Application.Quit();
    }
}
```

2.2.5 Класс Finish

Этот класс запускает финальное меню, наделяет функционалом кнопку Quit(Рис. 4(а)) и останавливает музыку.

```
public class Finish : MonoBehaviour
{
    public GameObject finishMenu;
    AudioSource song;

    void Start()
    {
        song = GetComponent<AudioSource>();
        finishMenu.SetActive(false);
    }

    void Update()
    {
        if (finishMenu.activeInHierarchy == true)
        {
            song.Stop();
        }
    }
}
```

2.2.6 Класс DualShock

Этот класс устанавливает цвет свечения панели на геймпаде, позволяет начать игру с начала при совместном нажатии кнопок L2 и R2 или выйти из игры при совместном нажатии на левый и правый стики.

```
public class DualShock : MonoBehaviour
{
    void Update()
    {
        try
        {
            DualShock4GamepadHID.current.SetLightBarColor(Color.green);
            if (DualShock4GamepadHID.current.leftTrigger.isPressed &&
                DualShock4GamepadHID.current.rightTrigger.isPressed)
            {
                SceneManager.LoadScene(SceneManager.GetActiveScene().name);
                Time.timeScale = 1f;

            }
            else if (DualShock4GamepadHID.current.leftStickButton.isPressed &&
                DualShock4GamepadHID.current.rightStickButton.isPressed)
            {
                Application.Quit();
            }
        }
        catch (NullReferenceException) { }
    }
}
```

2.2.7 Класс Play

Этот класс организует загрузку игровой сцены из главного меню при нажатии кнопки Play или сенсорной панели на геймпаде, позволяет выйти из игры при нажатии кнопки Quit или совместного нажатия на левый и правый стики на геймпаде.

```
public class Play : MonoBehaviour
{
    public AudioSource song;

    void FixedUpdate()
    {
        try
        {
            DualShock();
        }
        catch (NullReferenceException) { }
    }

    void DualShock()
    {
        DualShock4GamepadHID.current.SetLightBarColor(Color.green);
        if (DualShock4GamepadHID.current.touchpadButton.isPressed)
        {
            PlayGame();
        }
        else if(DualShock4GamepadHID.current.leftStickButton.isPressed &&
            DualShock4GamepadHID.current.rightStickButton.isPressed)
        {
            QuitGame();
        }
    }

    public void PlayGame()
    {
        song.Stop();
        SceneManager.LoadScene("SampleScene");
    }

    public void QuitGame()
    {
        Application.Quit();
    }
}
```

2.2.8 Класс Shrek

Этот класс описывает передвижение персонажа(бег, прыжок).

2.2.9 Класс Transmit

Этот класс обуславливает сменяемость анимаций персонажа.

2.2.10 Класс Traps

Этот класс запускает и отключает анимации ловушек при приближении персонажа.

```
public class Traps : MonoBehaviour
{
    public GameObject trap;
    public float distance;
    GameObject shrek;
    Animation anim;

    void Start()
    {
        shrek = GameObject.FindGameObjectWithTag("Player");
        anim = trap.GetComponent<Animation>();
    }

    void Update()
    {
        if (Mathf.Abs(shrek.transform.position.x - trap.transform.position.x) <= distance)
        {
            anim.Play();
        }
        else
        {
            anim.Stop();
        }
    }
}
```

2.3 Основные компоненты игровых объектов[2]

- Rigidbody 2D(двумерное твердое тело)

Отвечает за физические свойства объекта.

- Collider 2D

Отвечает за границы объекта.

- AudioSource

Источник звука.

- AudioListener

Приёмник звука.

- Animation

Анимация.

- Animator

Инструмент управления анимациями.

2.4 Описание игры

Главный герой – Шрек, который может управляться игроком, исполняя простые команды(движение влево, движение вправо, прыжок). Команды подаются с клавиатуры при помощи клавиш A и D или стрелок вправо, влево и пробела. В игре также присутствую шипы – объекты, при соприкосновении с которыми герой умирает и возвращается в начало пути, в течение игры также встречаются подвижные ловушки.

Цель игры: дойти до дома Шрека через долину и лес.

2.5 Визуальная составляющая игры

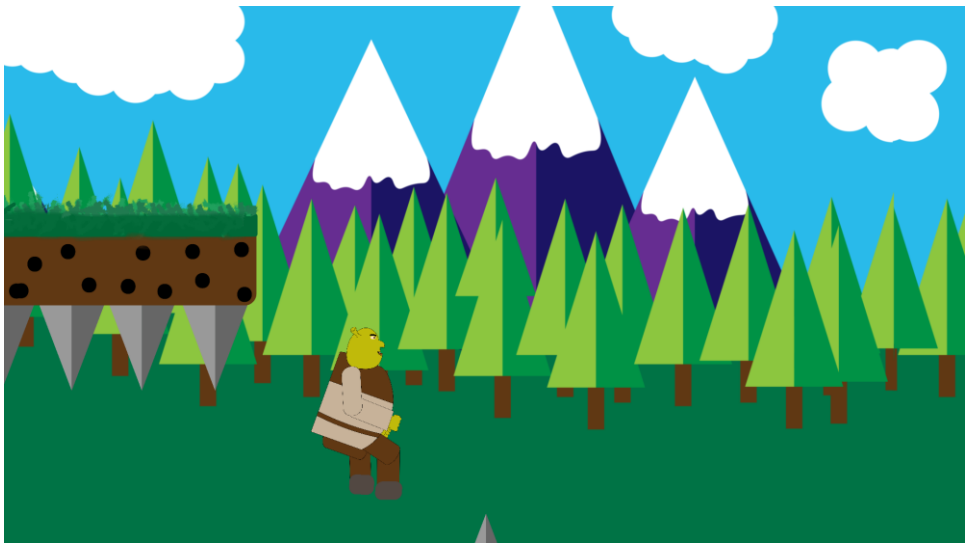

Рис. 7 – игровой процесс

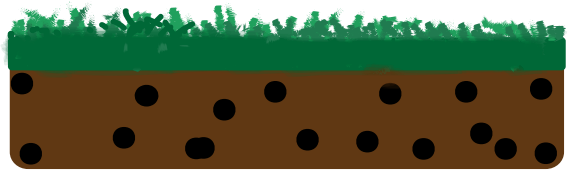

Рис. 8 – платформа 1

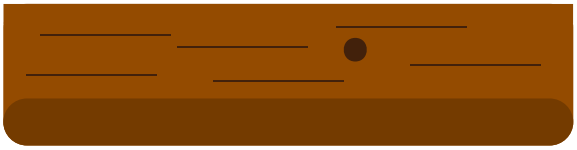

Рис. 9 – платформа 2

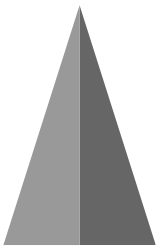

Рис. 10 – Шип

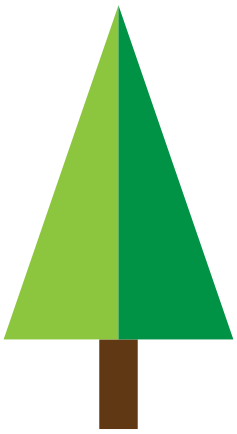

Рис. 11 - Дерево

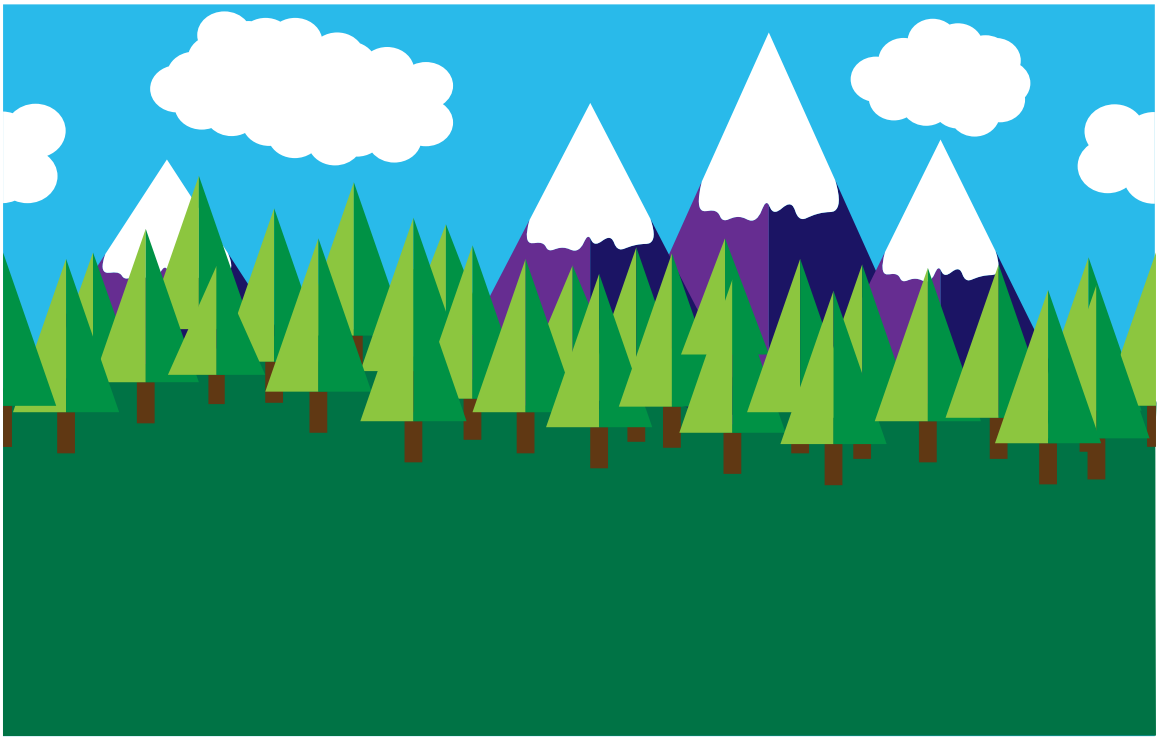

Рис. 12 – Локация Долина

Рис. 13 – Локация Лес

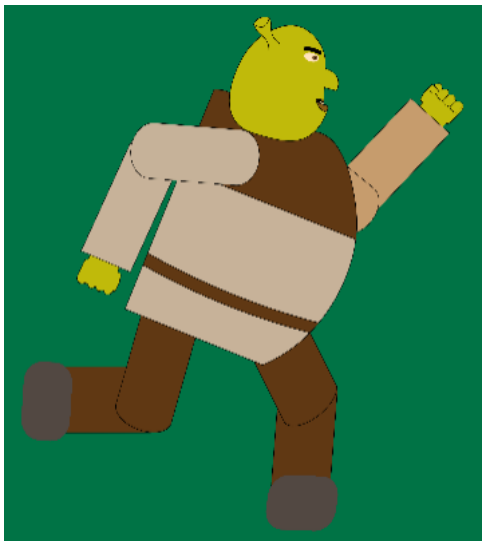

Рис. 14 – Персонаж

Все игровые объекты, включая локации и персонажа, были созданы в Adobe Illustrator 2019, затем конвертированы в формат PNG и добавлены на сцену в редакторе Unity.

Заключение

В результате выполнения курсового проекта с помощью современных средств был разработан 2D платформер для операционной системы Windows 64-битной архитектуры поддерживающий не только стандартный ввод с клавиатуры, но и ввод с геймпада DualShock 4. В ходе его разработки были получены теоретические и практические знания по языку программирования C#, по принципам объектно-ориентированного программирования, а также получены практические навыки работы с игровым движком Unity и графическим редактором Adobe Illustrator.

Список использованных источников

1. Электронный ресурс Wikipedia – https://ru.wikipedia.org
2. Руководство пользователя Adobe Illustrator – https://helpx.adobe.com
3. Документация по Unity – https://docs.unity3d.com
4. Аглоязычный форум Unity Answers – https://answers.unity.com
5. Видеохостинг YouTube – https://www.youtube.com