**Муниципальное бюджетное общеобразовательное учреждение**

**«Средняя общеобразовательная школа №7 г. Лениногорска»**

**МО«ЛМР» Республики Татарстан**

**Проектная работа**

**на тему: «Создание платформера на движке Unity»**

**Работу выполнили:**

**Ученики 10 «А» класса**

**Кузьмин Д., Васильев Г.**

**Руководитель:**

**учитель информатики**

**Осипова О.В.**

г. Лениногорск, 2019

**Содержание**

[Введение 3](#_Toc8065090)

[Актуальность. 4](#_Toc8065091)

[Задачи: 4](#_Toc8065092)

[Объект: 5](#_Toc8065093)

[Предмет: 5](#_Toc8065094)

[Гипотеза. 5](#_Toc8065095)

[Предполагаемый продукт 5](#_Toc8065096)

[Глава.1 Основная часть 6](#_Toc8065097)

[1.1.История компьютерных игр и платформеров 6](#_Toc8065098)

[1.2.Что такое платформер 6](#_Toc8065099)

[1.3.Роль компьютерных игр в современном мире 7](#_Toc8065100)

[1.4Платформер Unity 7](#_Toc8065101)

[1.5 Объяснение терминов 8](#_Toc8065102)

[1.6 С# программирование 8](#_Toc8065103)

[Глава 2. Практическая часть 9](#_Toc8065104)

[2.1.Изучение С# и Unity: 9](#_Toc8065105)

[2.2.План работы: 9](#_Toc8065106)

[2.3.Выбор методики работы: 10](#_Toc8065123)

[Вывод 11](#_Toc8065124)

[Cписок литератур и источников 12](#_Toc8065126)

[Приложение 1. 13](#_Toc8065127)

[14](#_Toc8065128)

Введение

В наше время игры – это место уединения, где человек может расслабиться, отвлечься от реальной жизни на некоторое время, выпустить агрессию. Самая первая игра была выпущена в 1962 году на основе простой программы. Через 10 лет (1972 г. 24 мая) была представлена самая первая игровая приставка - Magnavox Odyssey. В конце 1992 году вышла самая удачная и знаменитая игровая приставка – Dendy.С этих приставок и началась вся игровая индустрия. Позднее появился такой вид спорта, как киберспорт, в котором участники соревнуются, играя в ту или иную игру. На данный момент решается вопрос о добавлении киберспорта в список олимпийских игр, из чего следует то, что скорее всего появятся секции, в которых более опытные "спортсмены" будут обучать желающих (начинающих) данному виду «спорта». Позже стали появляться VR игры (Virtual Reality), смысл которых, надев специальные очки, полностью перенестись в виртуальный мир. В наше время создаются организации, перестраивающие обычные этажи здания в этажи, под определённые уровни VR игр, но данные организации существуют пока только в США. Суммируя все вышесказанное, можно сделать вывод, что игровая индустрия развивается с немыслимой скоростью. Кто знает, что будет завтра?

Актуальность.

Создание игры на платформе Unity в данный момент времени является актуальной, так как множество подростков играют в игры на телефоне и на компьютере.

**Цель:** Создать простую игру на платформе Unity, а в процессе его создания понять весь процесс создания игр, а именно: создание уровней, персонажей, игровых элементов таких как: объекты, игровое меню, пользовательское меню и т.п., игровых алгоритмов, игровой физики.

Задачи:

1. Изучить литературу по данной теме
2. Научиться обращаться с платформой Unity
3. Научиться писать коды для объектов Unity
4. Создать простой платформер
5. Создать 2 уровня для данного платформера

Объект: Программирование в школьном курсе информатики

Предмет: Игра-платформера на движке Unity, как дополнительное средство изучения программирования в рамках школьного курса информатики

Гипотеза: Любой человек: незнающий Unity, незнающий языков программирования, не умеющий работать с игровыми движками – может сделать свою игру, главное – усердие и желание.

Предполагаемый продукт: Платформер – жанр видео игр, где игрок управляет главным персонажем в 2D сцене.

Глава.1 Основная часть

1.1.История компьютерных игр и платформеров

История компьютерных игр начинается в 1940 и 1950 годах, когда в академической среде разрабатывались простые игры и симуляции. Компьютерные игры длительное время не были популярны, и только в 1970-х и 1980-х годах, когда появились для широкой публики аркадные автоматы, игровые консоли и компьютеры, компьютерные игры становятся частью поп культуры.

В 1889 году Фусадзиро Ямаути основал компанию под названием Marufuku, которая делала игральные карты, но в 1907 году компанию переименовали в Nintendo Koppai, впоследствии крупнейшая компания в мире среди производителей интерактивных развлечений.

Платформеры появились в начале 1980-х, когда приставки не были достаточно мощными, чтобы отображать трехмернуюграфику или видео. Они были ограничены статическими игровыми мирами, которые помещались на один экран, а герой был виден в профиль. Персонаж лазал вверх и вниз по лестницам или прыгал с платформы на платформу, часто сражаясь с противниками и собирая предметы. Первыми играми этого типа были Space panic и Apple panic. За ними последовала игра Donkey Kong, аркадная игра созданная фирмой «Nintendo» и выпущенная в 1981 году

1.2.Что такое платформер

Платформер-это жанр компьютерных игр , в которых основной чертой игрового процесса является прыгание по платформам, лазанье по лестницам, собирание предметов, обычно необходимых для завершения уровня. Коллекционные предметы, оружие собираются обычно простым прикосновением персонажа и для применения не требуют специальных действий со стороны игрока. Реже предметы собираются в «инвентарь» героя и применяются специальной командой (такое поведение более характерно для аркадных головоломок).

Противники (называемые «врагами»), всегда многочисленные и разнородные, обладают примитивным искусственным интеллектом , стремясь максимально приблизиться к игроку, либо не обладают им вовсе, перемещаясь по круговой дистанции или совершая повторяющиеся действия. Соприкосновение с противником обычно отнимает жизненные силы у героя или вовсе убивает его. Иногда противник может быть нейтрализован его из оружия, если им обладает герой. Смерть живых существ обычно изображается упрощённо или символически. Игры подобного жанра характеризуются нереалистичностью, рисованной мультяшной графикой. Героями таких игр обычно бывают мифические существа.

1.3.Роль компьютерных игр в современном мире

Сегодня людям надо занять себя чем-то, что не позволит им скучать. Одним из вариантов таких занятий стали компьютерные игры.

Ведь мало кто использует компьютер исключительно для работы. Хотя бы раз открыв для себя мир игры, только исключения смогут от него отказаться, тем более сегодня, когда компьютерные игры достигли такого уровня, что порой сложно определить, где реальный мир, а где виртуальный. За играми время летит быстрее, почему бы не позволить себе такое удовольствие.

1.4Платформер Unity

Unity- Это место где создают игры. Unity позволяет создавать приложения, для персональных компьютеров, игровых консолей, мобильных устройств, интернет-приложений и другие. Выпуск Unity состоялся в 2005 году и с того времени идёт постоянное развитие.

Основными преимуществами Unity являются наличие визуальной среды разработки, межплатформенной поддержки и модульной системы компонентов. К недостаткам относят появление сложностей при работе с многокомпонентными схемами и затруднения при подключении внешних библиотек.

На Unity написаны тысячи игр, приложений и симуляций, которые охватывают множество платформ и жанров. При этом Unity используется как крупными разработчиками, так и независимые студии.

1.5 Объяснение терминов

Sprite(спрайт) — это графический объект в компьютерной графике. Слово «спрайт» было придумано в 1970-е годы одним сотрудником компании Texas Instruments: их новая микросхема TMS9918 могла аппаратно отображать небольшие картинки поверх неподвижного фона. Изначально под спрайтами понимали небольшие рисунки, которые выводились на экран с применением аппаратного ускорения. На некоторых машинах (MSX 1, NES) программная прорисовка приводила к определённым ограничениям, а аппаратные спрайты этого ограничения не имели. Впоследствии с увеличением мощности центрального процессора от аппаратных спрайтов отказались и понятие «спрайт» распространилось на всех двумерных персонажей. В частности, в видеоиграх Super Mario и Heroes of Might and Magic вся графика спрайтовая, анимированная с помощью перебора изображений из атласа спрайтов

Tutorial(туториал)- это инструкция по выполнению чего-либо

Mobs(мобы)- это движущиеся существа, управляемые искусственным интеллектом

1.6 С# программирование

Изначально, Microsoft собиралась выпустить свою версию языка Java (Microsoft Java или J++), однако им пришлось судиться с правообладателями (Sun Microsystems) из-за некоторых спорных моментов. Поэтому руководство приняло решение о необходимости создания собственного языка, который бы отвечал их требованиям и развитие которого они могли бы контролировать. Так и появился С#.

С# был разработан благодаря усилиям Андерса Хейлсберга - создателя компилятора, который лег в основу Turbo Pascal и языка программирования Delphi. Первая версия языка увидел свет в июне 2000 года (возможно, что майкрософт хотела отметить так новое тысячелетие), а окончательная версия вышла в 2002 году вместе с Visual Studio. Сейчас C# стало одним из самых популярных языков программирования, начавшим даже немного опережать своего предшественника.

Глава 2. Практическая часть

2.1.Изучение С# и Unity:

C# имеет строгую архитектуру со статической типизацией. Относится к семейству C-подобных. Он похож на языки программирования Си++ и Java. Он разнообразен, но при этом прост в изучении.

Unity является номером 1 в создании игр для ПК. В программе есть такие средства, которые используются для разработки ПО. Благодаря простому и понятному оформлению создание игр становится простым и комфортным. Благодаря Unity, создание игр становится универсальным.

Unity позволяет разрабатывать игры без специальных навыков для этого. Можно сразу поставить спрайты, и заставить их взаимодействовать с окружающими объектами. Плюс Unity в том, что можно сразу проверить проделанную работу. Работает с DirectX и OpenGL и со многими другими программнимы обеспечениями.

По этому мы решили выбрать именно С# программирования.

2.2.План работы:

Метод теоретического иследования - изучении и обобщение. Мы думаем, что она лучше всего подойдет для работы под нашу тему.

С# программирование очень проста в использование с Unity

|  |  |  |
| --- | --- | --- |
| № | ЭТАП | СОДЕРЖАНИЯ ЭТАПА |
| 1 | Поиск спрайтов. | Мы были в поисках ресурсов для создание спрайта (Герой, мобы, фон, декорации). В этом действии нам помог интернет. |
| 2 | Создание скриптов | Создать скрипт главному герою и мобам (Ходьба, прыжок, стрельба) |
| 3 | Создание меню | Создать меню для игру (Играть, настройки. выход) и туториал, где объясняется управление героя. |
| 4 | Создание уровней | Создать уровней. Расставить препятствия, мобов, главного героя, платформы. |

## Создание программы.

1.Изучение программы Unity.

2. Написание скриптов.

3.Создание сцен и уровней.

4.Структурирование сцен и уровней.

5.Тестирование.

6.Исправление ошибок.

7. СистемныЕ требований.

8.Запись игры на диск.

Системные требования

Процессор: Intel Core 2 Duo (or equivalent)

Оперативная память: 2048 MB ОЗУ

Видеокарта: DirectX9 Compatible GPU with 1024 MB Video RAM

DirectX: Версии 9.0

Место на диске: 100 MB

Для 64 разрядной windows

2.3.Выбор методики работы:

Мы выбрали метод работы эксперимент, потому что, создавая игру мы будем учиться писать скрипт для игры, заставлять взаимодействовать скрипты и спрайты. Во время появления ошибок, мы будем искать их в программном коде. Исправляя эти ошибки , игра будет становиться лучше.

Вывод

Мы поставили цель создать простую игру на платформе Unity, а в процессе создания понять весь процесс создания игр, а именно: создание уровней, персонажей, игровых элементов таких как: объекты, игровое меню, пользовательское меню и т.п., игровых алгоритмов, игровой физики. Для этой цели мы поставили задачи: изучить литературу по данной теме, научиться обращаться с платформой Unity, научиться писать коды для объектов Unity, создать простой платформер и создать 2 уровня для данной игры. Мы сделали 2D платформер, что и планировали изначально изучив документацию по языку программирования C#.

Cписок литератур и источников

1.Unity и C# <https://www.youtube.com/watch?v=CAPVBTKk3Ww>

2. Алан Торн.   Основы анимации в Unity: 2016

3. Роберт и Мика Мартины.Принципы, паттерны и методики гибкой разработки на языке C#

4. Документация по Unity-https://docs.unity3d.com/ru/530/Manual/

5.История компьютерных игр-https://ru.wikipedia.org/wiki/%D0%98%D1%81%D1%82%D0%BE%D1%80%D0%B8%D1%8F\_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D1%8B%D1%85\_%D0%B8%D0%B3%D1%80

6. Платформер-https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B5%D1%80

7. Принципы, паттерны и методики гибкой разработки на языке C#

8. Спрайт-https://en.wikipedia.org/wiki/Sprite\_(computer\_graphics)

9. Туториал-https://de.wikipedia.org/wiki/Tutorial

Приложение 1.

**1)** Спрайт

gothic-hero-idle.png

gothic-hero-run.png





hell-hound-run.png



Скрипты

Все о героее

1!using UnityEngine;

using System.Collections;

public class Character : Unit

{

[SerializeField]

private int lives = 1;

public int Lives

{

get { return lives; }

set

{

if (value < 1) lives = value;

}

}

private LivesBar livesBar;

[SerializeField]

private float speed = 3.0F;

[SerializeField]

private float jumpForce = 15.0F;

private bool isGrounded = false;

private Bullet bullet;

private CharState State

{

get { return (CharState)animator.GetInteger("State"); }

set { animator.SetInteger("State", (int)value); }

}

new private Rigidbody2D rigidbody;

private Animator animator;

private SpriteRenderer sprite;

private void Awake()

{

livesBar = FindObjectOfType<LivesBar>();

rigidbody = GetComponent<Rigidbody2D>();

animator = GetComponent<Animator>();

sprite = GetComponentInChildren<SpriteRenderer>();

bullet = Resources.Load<Bullet>("Bullet");

}

private void FixedUpdate()

{

CheckGround();

}

private void Update()

{

if (isGrounded) State = CharState.Idle;

if (Input.GetButtonDown("Fire1")) Shoot();

if (Input.GetButton("Horizontal")) Run();

if (isGrounded && Input.GetButtonDown("Jump")) Jump();

}

private void Run()

{

Vector3 direction = transform.right \* Input.GetAxis("Horizontal");

transform.position = Vector3.MoveTowards(transform.position, transform.position + direction, speed \* Time.deltaTime);

sprite.flipX = direction.x < 0.0F;

if (isGrounded) State = CharState.Run;

}

private void Jump()

{

rigidbody.AddForce(transform.up \* jumpForce, ForceMode2D.Impulse);

}

private void Shoot()

{

}

public GameObject respawn;

public override void ReceiveDamage()

{

Lives--;

rigidbody.velocity = Vector3.zero;

rigidbody.AddForce(transform.up \* 8.0F, ForceMode2D.Impulse);

if (lives < 1)

{

void OnTriggerEnter2D(Collider2D other)

{

if (other.tag == "Player")

{

other.transform.position = respawn.transform.position;

}

}

}

Debug.Log(lives);

}

private void CheckGround()

{

Collider2D[] colliders = Physics2D.OverlapCircleAll(transform.position, 0.3F);

isGrounded = colliders.Length > 1;

if (!isGrounded) State = CharState.Jump;

}

private void OnTriggerEnter2D(Collider2D collider)

{

Bullet bullet = collider.gameObject.GetComponent<Bullet>();

if (bullet && bullet.Parent != gameObject)

{

ReceiveDamage();

}

}

}

public enum CharState

{

Idle,

Run,

Jump

Shoot

}

Скрипт взаимодействия с мобами

2! public class DieSpace : MonoBehaviour

{

public GameObject respawn;

void OnTriggerEnter2D(Collider2D other)

{

if (other.tag == "Player")

{

other.transform.position = respawn.transform.position;

}

}

}

Стреляющий бес

3!public class ShootableMonster : Monster

{

[SerializeField]

private float rate = 2.0F;

[SerializeField]

private Color bulletColor = Color.white;

private Bullet bullet;

protected override void Awake()

{

bullet = Resources.Load<Bullet>("Bullet2");

}

protected override void Start()

{

InvokeRepeating("Shoot", rate, rate);

}

private void Shoot()

{

Vector3 position = transform.position; position.y += 0.5F;

Bullet newBullet2 = Instantiate(bullet, position, bullet.transform.rotation) as Bullet;

newBullet2.Parent = gameObject;

newBullet2.Direction = -newBullet2.transform.right;

newBullet2.Color = bulletColor;

}

protected override void OnTriggerEnter2D(Collider2D collider)

{

Unit unit = collider.GetComponent<Unit>();

if (unit && unit is Character)

{

if (Mathf.Abs(unit.transform.position.x - transform.position.x) < 0.3F) ;

else unit.ReceiveDamage();

}

}

}

**Меню**

4! public class Menu : MonoBehaviour

{

public Slider slider;

public GameObject settings;

public Text valueCount;

public GameObject levelChanger;

public GameObject exitPanel;

void Update()

{

valueCount.text = slider.value.ToString();

AudioListener.volume = slider.value;

if (levelChanger.activeSelf == true && Input.GetKeyDown(KeyCode.Escape))

{

levelChanger.SetActive(false);

}

else if (exitPanel.activeSelf == false && Input.GetKeyDown(KeyCode.Escape))

{

exitPanel.SetActive(true);

}

else if (Input.GetKeyDown(KeyCode.Escape))

{

exitPanel.SetActive(false);

}

}

public void OnClickStart()

{

levelChanger.SetActive(true);

}

public void Exit()

{

exitPanel.SetActive(true);

}

public void No()

{

exitPanel.SetActive(false);

}

public void OnclickExit()

{

Application.Quit();

}

public void Settings()

{

settings.SetActive(!settings.activeSelf);

}

public void levelBttns(int level)

{

SceneManager.LoadScene(level);

}

}

**Пули монстра**

5! public class Bullet : MonoBehaviour

{

private GameObject parent;

public GameObject Parent { set { parent = value; } get { return parent; } }

private float speed = 10.0F;

private Vector3 direction;

public Vector3 Direction { set { direction = value; } }

public Color Color

{

set { sprite.color = value; }

}

private SpriteRenderer sprite;

private void Awake()

{

sprite = GetComponentInChildren<SpriteRenderer>();

}

private void Start()

{

Destroy(gameObject, 1.4F);

}

private void Update()

{

transform.position = Vector3.MoveTowards(transform.position, transform.position + direction, speed \* Time.deltaTime);

}

private void OnTriggerEnter2D(Collider2D collider)

{

Unit unit = collider.GetComponent<Unit>();

if (unit && unit.gameObject != parent)

{

Destroy(gameObject);

}

}

}

Камера

6! public class Camera : MonoBehaviour

{

public float damping = 1.5f;

public Vector2 offset = new Vector2(2f, 1f);

public bool faceLeft;

private Transform player;

private int lastX;

void Start()

{

offset = new Vector2(Mathf.Abs(offset.x), offset.y);

FindPlayer(faceLeft);

}

public void FindPlayer(bool playerFaceLeft)

{

player = GameObject.FindGameObjectWithTag("Player").transform;

lastX = Mathf.RoundToInt(player.position.x);

if (playerFaceLeft)

{

transform.position = new Vector3(player.position.x - offset.x, player.position.y + offset.y, transform.position.z);

}

else

{

transform.position = new Vector3(player.position.x + offset.x, player.position.y + offset.y, transform.position.z);

}

}

void Update()

{

if (player)

{

int currentX = Mathf.RoundToInt(player.position.x);

if (currentX > lastX) faceLeft = false; else if (currentX < lastX) faceLeft = true;

lastX = Mathf.RoundToInt(player.position.x);

Vector3 target;

if (faceLeft)

{

target = new Vector3(player.position.x - offset.x, player.position.y + offset.y, transform.position.z);

}

else

{

target = new Vector3(player.position.x + offset.x, player.position.y + offset.y, transform.position.z);

}

Vector3 currentPosition = Vector3.Lerp(transform.position, target, damping \* Time.deltaTime);

transform.position = currentPosition;

}

}

}