МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Забайкальский государственный университет»

(ФГБОУ ВО «ЗабГУ»)

Факультет: Энергетический

Кафедра: Информатики, вычислительной техники и прикладной математики

**КУРСОВАЯ РАБОТА**

По дисциплине: Интерактивные графические системы

На тему: «Разработка интерактивной графической системы»

Выполнил студент группы ИВТ–18, Долгов Александр Артемович

Руководитель работы: старший преподаватель кафедры ИВТ и ПМ, Долгих Роман Сергеевич

Чита

2021

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Забайкальский государственный университет»

(ФГБОУ ВО «ЗабГУ»)

Факультет: Энергетический

Кафедра: Информатики, вычислительной техники и прикладной математики

**ЗАДАНИЕ**

на курсовую работу

По дисциплине: Интерактивные графические системы

Студенту: Долгову Александру Артемовичу

Специальности (направления подготовки): 09.03.01 Информатика и вычислительная техника

1 Тема курсовой работы: «Разработка интерактивной графической системы»

2 Срок подачи студентом законченной работы: 22.12.2021

3 Исходные данные к работе:

1. Описание предметной области.

2. «Общие требования к построению и оформлению учебной текстовой документации» (МИ 01-02-2018)».

Дата выдачи задания: 15.09.2021

Руководитель курсовой работы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Долгих Р.С./

(подпись, расшифровка подписи)

Задание принял к исполнению

«15» сентября 2021 г.

Подпись студента\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Долгов А.А./

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Забайкальский государственный университет»

(ФГБОУ ВО «ЗабГУ»)

Факультет: Энергетический

Кафедра: Информатики, вычислительной техники и прикладной математики

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовой работе

По дисциплине: Интерактивные графические системы

На тему: «Разработка интерактивной графической системы»

Выполнил студент группы ИВТ–18, Долгов Александр Артемович

Руководитель работы: старший преподаватель кафедры ИВТ и ПМ, Долгих Роман Сергеевич

Чита

2021

СОДЕРЖАНИЕ

[Введение 6](#_Toc90744108)

[1 Описание предметной области 7](#_Toc90744109)

[2 Описание средств реализации 8](#_Toc90744110)

[2.1 Среда разработки Unity 8](#_Toc90744111)

[2.2 Язык программирования С# 9](#_Toc90744112)

[3 Программная реализация 10](#_Toc90744113)

[Заключение 12](#_Toc90744114)

[Список использованных источников 13](#_Toc90744115)

[Приложения 14](#_Toc90744116)

РЕФЕРАТ

Пояснительная записка 21с., 7 рис., 0 табл., 5 источников, 2 прил.

Ключевые слова: ИГРА, UNITY, С#, 3D RUNNER, РАННЕР, БЕСКОНЕЧНЫЙ.

В данной работе рассматривается процесс создания компьютерной игры в стиле бесконечного 3D «раннера» на игровом движке Unity. Для написания скриптов был использован язык программирования C#.

В работе определены методы разработки и описан процесс их применения при создании данной игры.

ВВЕДЕНИЕ

Цель данной работы – разработать интерактивную графическую систему, которая будет представлять собой компьютерную 3D-игру на графическом движке Unity.

Задачи:

– изучить работу игрового движка;

– использовать возможности языка C# при работе с игровым движком;

– реализовать игру – бесконечный 3D «раннер».

В данной работе реализована простая механика передвижения персонажа по бесконечной платформе. Задачей игрока будет обход препятствий во время движения, которые могут возникать в случайных местах на его пути.

1 Описание предметной области

Endless (Infinity) runner – это игры, где персонаж постоянно бежит/летит/ползет вперед по теоретически бесконечному игровому миру. Этот вид игр можно отнести к таким жанрам, как «Игры для всех», «Аркада», «Гипер-казуальные игры» – простые игры с элементарной механикой, в которых происходит мгновенный переход к геймплею в начале игры.

Такого жанра, как бесконечный 3D RUNNER, официально нет в классификации различных магазинов игр (Steam, AppStore, Google Play), но подобные игры стали довольно популярными из-за своей простоты и незамысловатости.

Как правило, всевозможные «раннеры» относят к играм на реакцию. Игроку предстоит управлять персонажем, уворачиваться от препятствий, попутно собирая бонусы и монеты. Принцип жанра — чем дальше, тем сложнее. Следовательно, игра не надоест, пока в ней есть чего достигать и куда бежать. Управление, как правило, интуитивное с малым набором элементов управления [1].

2 Описание средств реализации

**2.1 Среда разработки Unity**

Unity (unity в переводе с англ. «единство») — межплатформенная среда разработки компьютерных игр, разработанная американской компанией Unity Technologies. Unity позволяет создавать приложения, работающие на более чем 25 различных платформах, включающих персональные компьютеры, игровые консоли, мобильные устройства, интернет-приложения и другие.

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, визуализации математических моделей, которые охватывают множество платформ и жанров. При этом Unity используется как крупными разработчиками, так и независимыми студиями.

Unity сразу идёт в комплекте со средой разработки. Это одновременно и редактор кода, и работа с графикой, и логика поведения предметов в игровой сцене, и всё остальное.

Особенность этого редактора в том, что всё это можно делать и настраивать прямо во время запуска или тестирования проекта. Например, если вам не понравилось, как машина в игре реагирует на лужи, то, можно не выходя из сцены поправить нужный параметр и сразу увидеть изменения.

В Unity уже есть готовая физика и шаблоны для создания своих правил — можно сразу использовать их в игре. Движок уже знает, что, если на горку поставить коробку, она сползёт, перевернётся и остановится, а мяч в этой же ситуации должен просто покатиться дальше.

Компоненты в Unity определяют поведение объектов на сцене. Компонентами является всё, что прикрепляется к игровому объекту.

Rigidbody – это основной компонент, подключающий физическое поведение для объектов, которое не придется прописывать вручную [4].

Благодаря данному компоненту станет возможным столкновение игрока с препятствиями.

**2.2 Язык программирования С#**

Язык программирования C# используется движком Unity для написания скриптов.

Данный язык является одним из самых мощных, быстро развивающихся и востребованных языков в IT-отрасли. В настоящий момент на нем пишутся самые различные приложения: от небольших десктопных программок до крупных веб-порталов и веб-сервисов, обслуживающих ежедневно миллионы пользователей.

C# является объектно-ориентированным и в этом плане много перенял у Java и С++. Например, C# поддерживает полиморфизм, наследование, перегрузку операторов, статическую типизацию. Объектно-ориентированный подход позволяет решить задачи по построению крупных, но в тоже время гибких, масштабируемых и расширяемых приложений. И C# продолжает активно развиваться, и с каждой новой версией появляется все больше интересных функциональностей [3].

Написание скриптов производилось в интегрированной среде разработки Microsoft Visual Studio.

3 Программная реализация

В данной игре был реализован следующий функционал:

– простая механика управления кубом, которой управляет игрок, с возможностью перемещения влево и вправо с реалистичной анимацией;

– генерация препятствий в случайных местах, которые двигаются навстречу игроку с различной скоростью;

– при столкновении игрока с препятствием игра заканчивается;

– реализована возможность выбора сложности для игрока.

На главном меню находятся основные графические элементы. Сверху указывается лучший счёт игрока за всё время (best score), а также счёт в текущей игре (current score), который обновляется каждый раз после поражения.

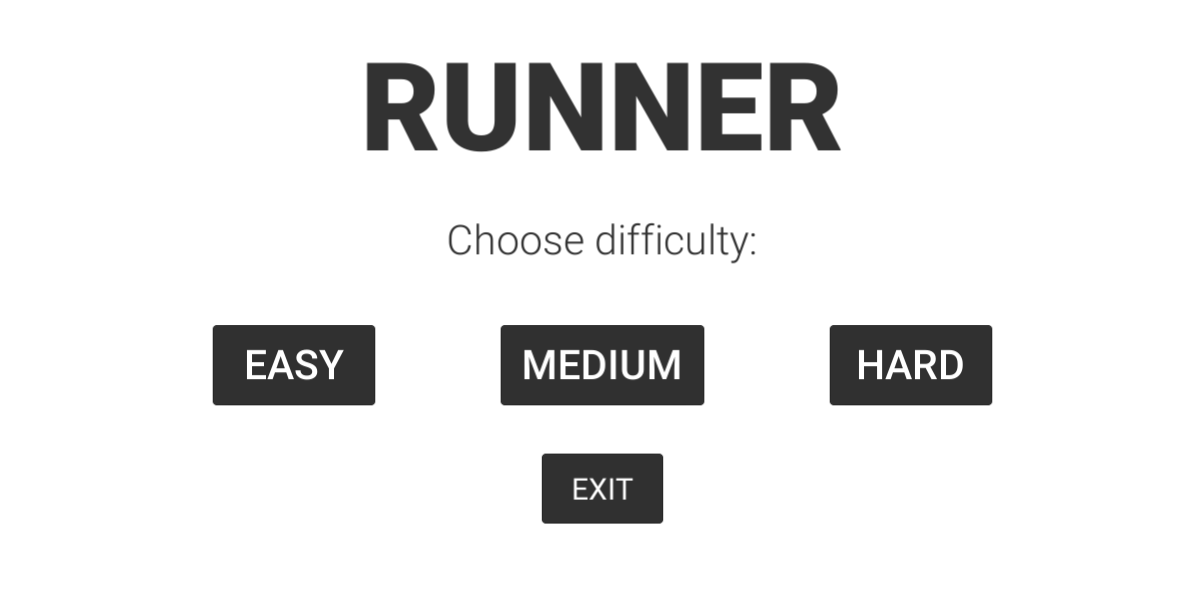


Рисунок 1 – Главное меню

В центре находятся четыре кнопки: easy, medium, hard – запуск новой игры и выбор сложности, exit – выход из игры.

В соответствии с рисунком 2 игрок управляет кубом, который движется по платформе. Навстречу ей направляются препятствия – черные кубы, при столкновении с которыми игра будет заканчиваться. Также постоянно увеличивается расстояние, которое прошел игрок не умерев.

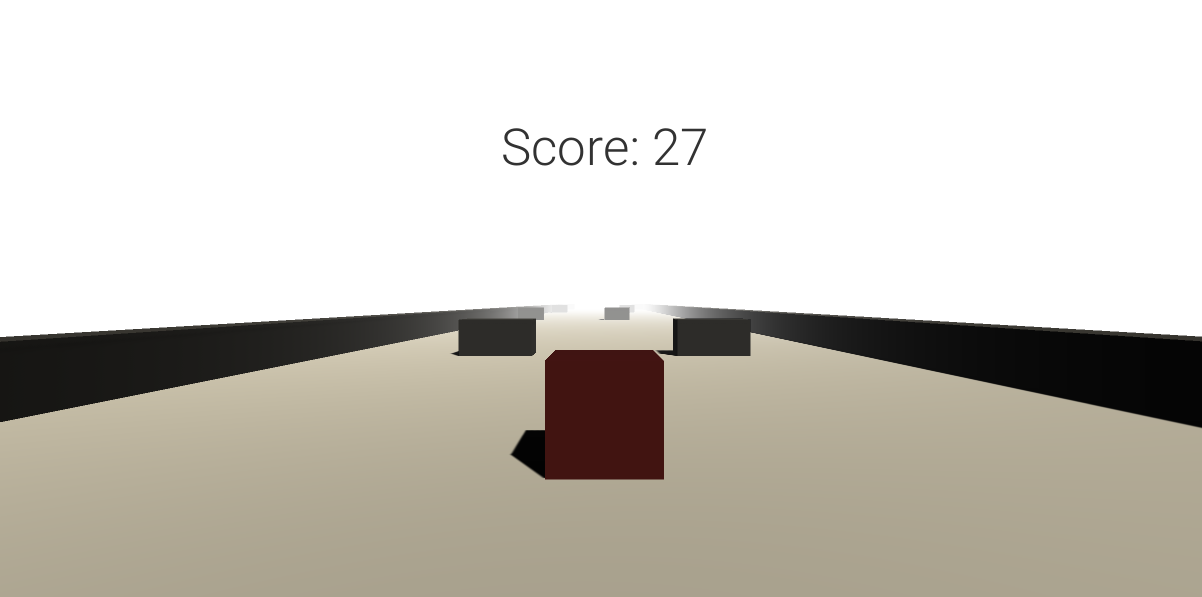


Рисунок 2 – Геймплей игры

ЗАКЛЮЧЕНИЕ

В результате проделанной работы была разработана игра в стиле бесконечного 3D «раннера».

Была изучена работа игрового движка, его механизмы и свойства, а также взаимодействие с 3D моделями. Скрипт игры был написан на объектно-ориентированном языке высокого уровня C#.

Игра функционирует под управлением ОС семейства Windows, но также в дальнейшем может быть портирована на мобильные устройства (Android).

Был реализован весь вышеописанный функционал игры.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ЧТО ЗА ЖАНР? ENDLESS (INFINITY) RUNNING GAME [Электронный ресурс] – URL: https://stopgame.ru/blogs/topic/85445 (Дата обращения: 18.12.2021)
2. Википедия – Свободная энциклопедия [Электронный ресурс] – URL: https://ru.wikipedia.org/wiki/Unity\_(игровой\_движок) (Дата обращения: 22.11.2021)
3. C# и .Net | Введение [Электронный ресурс] – URL: https://metanit.com/sharp/tutorial/1.1.php (Дата обращения: 18.12.2021)
4. UNITY - ВСЁ ПРО RIGIDBODY. [Электронный ресурс] – URL: https://media-xyz.com/ru/articles/1180-unity-vsio-pro-rigidbody (Дата обращения: 18.12.2021)
5. Кто такая Unity и почему она всем нравится [Электронный ресурс] – URL: https://thecode.media/unity/ (Дата обращения: 18.12.2021)

Приложение А. Программный код класса Fade

using UnityEngine;

public class Fade : MonoBehaviour

{

public float fadeSpeed = 1.25f;

private int fadeDirection = -1;

private Color colorGUI = Color.white;

private void OnGUI()

{

// Рассчитать новую альфу

colorGUI.a = Mathf.Clamp01(colorGUI.a + fadeDirection \* fadeSpeed \* Time.deltaTime);

// Помещение GUI (наложение затухания) на верхний слой и прорисовка его.

GUI.depth = -1000;

GUI.color = colorGUI;

GUI.DrawTexture(new Rect(0, 0, Screen.width, Screen.height), Texture2D.whiteTexture);

}

public float BeginFade(int direction)

{

fadeDirection = direction;

// Вернуть, сколько времени потребуется, чтобы исчезнуть

return (1 / fadeSpeed);

}

}

Приложение Б. Программный код класса FollowTransform

using UnityEngine;

public class FollowTransform : MonoBehaviour

{

public Transform transformObject;

public int axis;

private void Update()

{

Vector3 newPosition;

newPosition = transform.position;

newPosition[axis] = transformObject.position[axis];

transform.position = newPosition;

}

}

Приложение В. Программный код класса MainMenu

using UnityEngine;

using UnityEngine.SceneManagement;

using System.Collections;

public class MainMenu : MonoBehaviour

{

public Fade fade;

IEnumerator SceneTransition(int scene)

{

yield return new WaitForSeconds(fade.BeginFade(1));

SceneManager.LoadScene(scene, LoadSceneMode.Single);

}

public void ExitGame()

{

Application.Quit();

}

public void StartGame(int difficulty)

{

GameManager.difficulty = difficulty;

StartCoroutine(SceneTransition(1));

}

private void Start()

{

fade.BeginFade(-1);

}

}

Приложение Г. Программный код класса ObstacleMovement

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class ObstacleMovement : MonoBehaviour

{

public float force = 500f;

// Скорость столкновения (X - угловой множитель)

public Vector3 collisionVelocity = new Vector3(25f, 5f, 10f);

private void OnCollisionEnter(Collision collision)

{

switch (collision.gameObject.tag)

{

case "Player":

// Бросьте препятствие в воздух для драматического эффекта

Rigidbody obstacle = GetComponent<Rigidbody>();

obstacle.velocity = new Vector3(obstacle.velocity.x,

collisionVelocity.y,

collisionVelocity.z);

obstacle.angularVelocity = obstacle.angularVelocity \* collisionVelocity.x;

FindObjectOfType<GameManager>().InitiateDeath();

break;

case "ObstacleWall":

Destroy(gameObject);

break;

}

}

private void Update()

{

// Переместить препятствие по оси Z (к игроку)

GetComponent<Rigidbody>().AddForce(0f, 0f, -force);

}

}

Приложение Д. Программный код класса PlayerMovement

using UnityEngine;

public class PlayerMovement : MonoBehaviour

{

public float force = 200f;

private void FixedUpdate()

{

Rigidbody player;

player = GetComponent<Rigidbody>();

// Переместить игрока по оси Z (в сторону)

if (Input.GetKey(KeyCode.D) || Input.GetKey(KeyCode.RightArrow))

{

player.AddForce(force \* Time.deltaTime, 0f, 0f, ForceMode.VelocityChange);

}

if (Input.GetKey(KeyCode.A) || Input.GetKey(KeyCode.LeftArrow))

{

player.AddForce(-force \* Time.deltaTime, 0f, 0f, ForceMode.VelocityChange);

}

}

}