Т. 10. №12 2024
https://doi.org/10.33619/2414-2948/109

УДК 004.946 https://doi.org/10.33619/2414-2948/109/21

ПРОЦЕСС СОЗДАНИЯ В СРЕДЕ РАЗРАБОТКИ UNITY ОБУЧАЮЩЕЙ АНГЛИЙСКОМУ ЯЗЫКУ ИГРЫ

©Сафина Г. Ф., *ORCID: 0000-0002-7326-0896, SPIN-код: 4562-2453, канд.физ.-мат.наук, Уфимский университет науки и технолологий, Нефтекамский филиал, г. Нефтекамск, Россия,safinagf@mail.ru*
©Кириллова Е. А., *Уфимский университет науки и технолологий, Нефтекамский филиал, г. Нефтекамск, Россия, elize.none@yandex.ru*

CREATING AN ENGLISH LEARNING GAME IN THE UNITY DEVELOPMENT ENVIRONMENT

©Safina G., *ORCID: 0000-0002-7326-0896, SPIN-code: 4562-2453, Ph.D., Ufa University of Science and Technology, Neftekamsk Branch, Neftekamsk, Russia,safinagf@mail.ru*
©Kirillova E., *Ufa University of Science and Technology, Neftekamsk Branch, Neftekamsk, Russia, elize.none@yandex.ru*

Аннотация. Приведены пошаговые компоненты и результаты процесса разработки обучающей игры с помощью кроссплатформенной среды Unity – редактора (движка) для разработки компьютерных игр. Игра призвана оказать помощь обучающимся в первичном знакомстве и освоении английского языка посредством взаимодействия с миром и его изучения. Кодовая часть игры реализована с помощью библиотек и функционала языка программирования C#, для визуализации проекта (игры) привлечены возможности 3d-моделирования.

Abstract. Shows the step-by-step components and results of the process of developing a training game using the Unity cross-platform environment—an editor (engine) for developing computer games. The game is designed to help students in primary acquaintance and mastering the English language through interaction with the world and its study. The code part of the game is implemented using libraries and functionality of the C# programming language; 3D modeling capabilities are involved to visualize the project (game).

Ключевые слова: движок Unity, разработка игр, английский язык, язык программирования C#, компьютерные игры, 3d-моделирование, игровое обучение.

Keywords: Unity engine, game development, English, C # programming language, computer games, 3d - modeling, game learning.

Видеоигры могут приносить много пользы, быть не только развлечением. Они помогают развивать различные когнитивные навыки. Например, многие игры требуют стратегического мышления и быстрого реагирования, что тренирует мозг и позволяет улучшить реакцию. Играя в игры на иностранном языке, вы усваиваете новые слова и фразы в контексте, что может помочь развить навыки чтения, понимания текста на другом языке [1, 2].

Рассмотрим процесс создания компьютерной игры, идея которой заключается помощи игроку в изучении английского языка посредством построения связи визуально знакомого

объекта с его названием на иностранном языке. Для реализации проекта используется среда разработки компьютерных игр Unity, позволяющая создавать приложения для разных платформ, операционных систем, устройств и обладающая богатым магазином готовых наборов ресурсов. Первоначально в разработке создаем новый 3d-проект (модель) под названием “Learn English” [2, 3]. Процесс создания представлен на Рисунке 1.

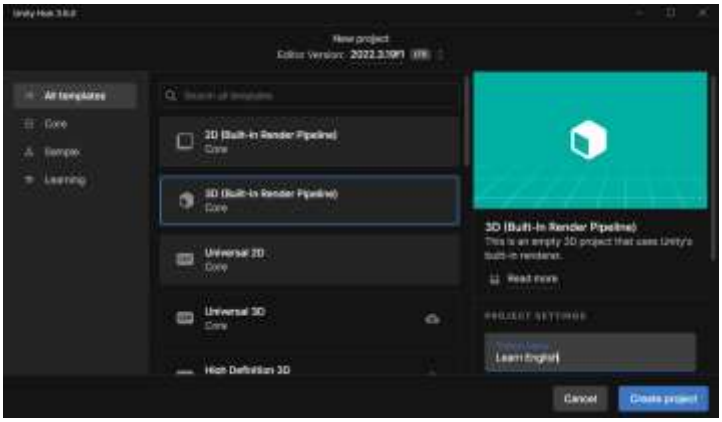


Рисунок 1. Создание нового проекта

Далее нужно реализовать игрока, локацию и тестовые предметы. Для создания игрока помещаем куб, который будет являться телом, внутрь камеры, названной Main Camera (Рисунок 2). Ей добавляем свойство RigidBody, позволяющее стать физически твердым объектом, благодаря которому мы не сможем проходить сквозь другие твердые объекты. Телу игрока добавляем свойство Box Collider, который помогает установить границы предмета в пространстве [3, 4].

Рисунок 2. Иерархия компонентов игрока

Для перемещения игрока создаем и закрепляем следующий скрипт на Main Camera:

```
public class move : MonoBehaviour
{
public float speed = 1.5f;
public Transform head;
public float sensitivity = 5f; // чувствительность мыши
public float headMinY = -40f; // ограничение угла для головы
public float headMaxY = 40f; // ограничение угла для головы
   private Vector3 direction; // хранит вектор направления движения
private float h, v;
private int layerMask;
private Rigidbody body;
private float rotationY;
void Start()
{
        body = GetComponent<Rigidbody>();
        body.freezeRotation = true;
```

Т. 10. №12 2024
https://doi.org/10.33619/2414-2948/109

```
            Cursor.visible = false; //скрываем курсор
            Cursor.lockState = CursorLockMode.Locked;
        }
    void FixedUpdate()
    {
            body.AddForce(direction * speed, ForceMode.VelocityChange);
            // Ограничение скорости, иначе объект будет постоянно ускоряться
            if (Mathf.Abs(body.velocity.x) > speed)// проверка на превышение скорости и ее ограничение по x
            {
                body.velocity = new Vector3(Mathf.Sign(body.velocity.x) * speed, body.velocity.y, body.velocity.z);
            }
            if (Mathf.Abs(body.velocity.z) > speed)// проверка на превышение скорости и ее ограничение по z
            {
                body.velocity = new Vector3(body.velocity.x, body.velocity.y, Mathf.Sign(body.velocity.z) * speed);
            }
        }
        void Update()
        {
            h = Input.GetAxis("Horizontal");
            v = Input.GetAxis("Vertical");
            // управление головой (камерой)
            float rotationX = head.localEulerAngles.y + Input.GetAxis("Mouse X") * sensitivity;
            rotationY += Input.GetAxis("Mouse Y") * sensitivity;
            rotationY = Mathf.Clamp(rotationY, headMinY, headMaxY);
            head.localEulerAngles = new Vector3(-rotationY, rotationX, 0); // применение вращения к голове
            // вектор направления движения
            direction = new Vector3(h, 0, v);
            direction = head.TransformDirection(direction);
            direction = new Vector3(direction.x, 0, direction.z);
        }
```

Добавим возможность совершать прыжки с помощью следующего скрипта, который также помещаем на Main Camera:

```
    public class JumpScript : MonoBehaviour
    {
        public float speed = 10f; //  скорость прыжка
        public Rigidbody rb;
        private bool isJumping; //флаг, указывающий на то, происходит ли сейчас прыжок
        void Start()
        {
            rb = GetComponent<Rigidbody>();
            isJumping = false; // сброс флага прыжка
        }
```

```
    void Update()
    {
        if (Input.GetKeyDown(KeyCode.Space) && isJumping == false) //проверка нажатия пробела
        {
            rb.velocity = new Vector3(0, 5, 0);
            isJumping = true;// установка флага прыжка в значение true
        }
    }
    private void OnCollisionEnter(Collision collision)
    {
        isJumping = false;// сброс флага прыжка
    }
```

Следующим шагом создаем несколько параллелепипедов – пол и ограждения локации, а также тестовые предметы, в данном случае это куб, шар и капсула (Рисунок 3). Добавляем им свойство Box Collider, чтобы нельзя было пройти сквозь них [3].

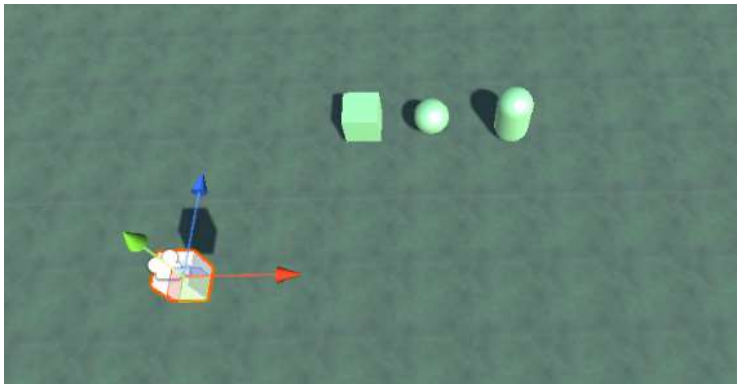


Рисунок 3. Тестовая локация

Далее разрабатываем интерфейс обучающей игры. Для этого создаем Canvas и текстовый объект, который называем OnHoverText (Рисунок 4).

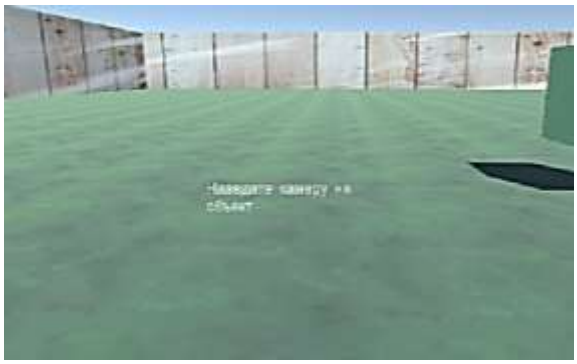
Рисунок 4. Иерархия объекта Canvas

Следующие два скрипта реализуют функциональность наведения курсора на объект:

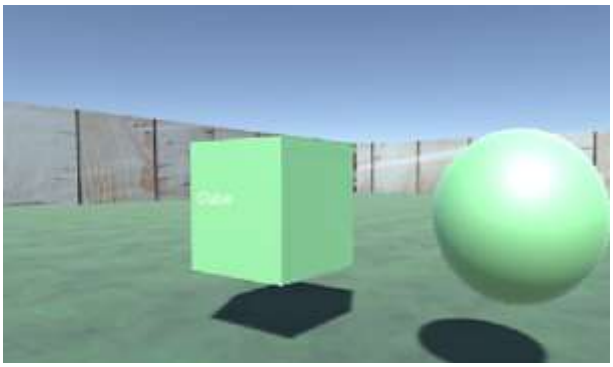
```
    // скрипт для Canvas
    public class UIManager : MonoBehaviour
    {
        private static TextMeshProUGUI OnHoverText;
        void Start()
        {
            OnHoverText = GameObject.Find("Canvas/OnHoverText").GetComponent<TextMeshProUGUI>();
        }
        public static void SetOnHoverText(string objName)
```

```
    {
        OnHoverText.text = objName;
    }
    public static void OffOnHoverText()
    {
        OnHoverText.text = "";
    }
}
// скрипт для объектов
public class Hoverable : MonoBehaviour
{
    private void OnMouseEnter()
    {
        UIManager.SetOnHoverText(gameObject.name);
    }
    private void OnMouseExit()
    {
        UIManager.OffOnHoverText();
    }
```

Когда курсор мыши находится в области объекта, он вызывает метод SetOnHoverText в классе UIManager для отображения названия объекта. Когда курсор мыши покидает область объекта, вызывается метод OffOnHoverText, который удаляет текст. Приступаем к тестированию. Запускаем игру и подходим к объектам, их названия успешно выводятся на экран. Далее добавим больше элементов локации и снова протестируем. Некоторые результаты тестирования представлены на Рисунке 5 (а, б, в, г).

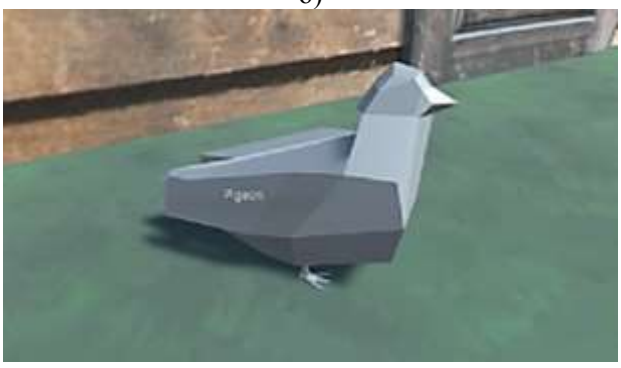


а)



б)

в)



г)

Рисунок 5. Тесты распознавания объектов

Таким образом, с помощью редактора Unity и языка программирования C# была успешно разработана компьютерная игра, помогающая в обучении английскому языку. Проект оснащен яркой графикой и интуитивно понятным интерфейсом, что делает его привлекательным, способствует изучению английского языка.

Источники:

(1). Unity. https://docs.unity3d.com/Manual/index.html
(2). Modern problems of science and education. https://lyl.su/Dl0e
(3). Scientific Benefits of Playing Videogames. https://lyl.su/6eOy
(4). FPS скрипт управлениия от первого лица. https://lyl.su/M5KJ

Список литературы:

1. Корнилов А. В. Unity полное руководство. СПб.: Наука и техника, 2021. 496 с.
2. Ларкович С. Н. Unity на практике. Создаем 3D-игры и 3D-миры. М.: Наука и техника, 2022. 384 с.
3. Мэннинг Д. Unity и для разработчика. СПб: Питер, 2018. 352 с.
4. Гейг М. Разработка игр на Unity 2018 за 24 часа. М.: Эксмо, Бомбора, 2020. 461 с.

References:

1. Kornilov, A. V. (2021). Unity polnoe rukovodstvo. St. Petersberg. (in Russian).
2. Larkovich, S. N. (2022). Unity na praktike. Sozdaem 3D-igry i 3D-miry. Moscow. (in Russian).
3. Menning, D. (2018). Unity i dlya razrabotchika. St. Petersberg. (in Russian).
4. Geig, M. (2020). Razrabotka igr na Unity 2018 za 24 chasa. Moscow. (in Russian).

Работа поступила
в редакцию 08.11.2024 г.

Принята к публикации
27.11.2024 г.

Ссылка для цитирования:

Сафина Г. Ф., Кириллова Е. А. Процесс создания в среде разработки Unity обучающей английскому языку игры // Бюллетень науки и практики. 2024. Т. 10. №12. С. 155-160. https://doi.org/10.33619/2414-2948/109/21

Cite as (APA):

Safina, G., & Kirillova, E. (2024). Creating an English Learning Game in the Unity Development Environment. *Bulletin of Science and Practice*, *10*(12), 155-160. (in Russian). https://doi.org/10.33619/2414-2948/109/21