

### Лабораторная работа 3.

#### «Создание мини-игры «Сбор букв» в Unity3D»

**Цель:** Изучить основы разработки простых 2D/3D игр с использованием игрового движка Unity, освоить работу с физикой, триггерами, интерфейсом пользователя и созданием игровой механики.

**Задание:** Создать простую игру в Unity, в которой игрок собирает буквы, двигаясь по игровому полю. Задача игрока — собрать как можно больше букв за ограниченное время.

**Номер варианта** - это буквы вашего полного имени и фамилии.

**Среда разработки:** Unity 3D.

**Язык:** C#.

#### Порядок выполнения:

1. Создание сцены:
  - Добавьте на сцену плоскость (Plane), которая будет служить полом.
  - С помощью инструмента TextMeshPro создайте 3D-буквы вашего полного имени и фамилии.
  - Расположите все буквы случайным образом на полу.
2. Напишите скрипт, который обеспечивает медленное вращение каждой буквы вокруг вертикальной оси (ось Y) для улучшения их визуального восприятия.
3. Управление игроком:
  - Создайте виртуальные кнопки: "Вперёд", "Назад" (UI-кнопки в нижней части экрана).
  - Добавьте компонент CharacterController для движения.
  - Реализуйте вращение камеры через свайп по экрану (сенсорное управление направлением обзора).
4. Настройте взаимодействие с буквами:
  - Каждая буква должна иметь триггер-коллайдер.
  - При прикосновении к игроку буква исчезает, очки увеличиваются на 1.
5. Таймер и интерфейс:
  - Добавьте на экран панель с двумя надписями: "Очки" — текущее количество собранных букв и "Время" — обратный отсчет времени (например, 60 секунд).
  - По окончании таймера игра заканчивается.

#### Методические указания к выполнению лабораторной работы №3

Для выполнения лабораторной работы №3 необходимо изучить лекцию 8 и произвести установку необходимого программного обеспечения:

1. Установите Unity Hub: <https://unity.com/>.
2. Установите Unity Editor с поддержкой Android: Откройте Unity Hub, перейдите во вкладку Installs → Add и выберите одну из стабильных версий Unity (например, 2021.3 LTS, 2022.3 LTS или новее). Добавьте следующие компоненты во время установки:
  - ✓ Android Build Support

- ✓ Visual Studio Community (или ваш любимый редактор кода)
- ✓ OpenJDK
- ✓ Android SDK & NDK Tools
- ✓ Documentation

3. Затем необходимо настроить среду разработки:

Шаг 1 — Проверьте настройки Android в Unity.

После открытия Unity Editor перейдите в меню: Edit → Preferences → External Tools и убедитесь, что указан правильный путь к JDK, SDK и NDK (обычно указывается автоматически после установки через Unity Hub). В разделе Configuration убедитесь, что Scripting Backend установлен как Mono2x или IL2CPP (рекомендуется для Android). Также выберите минимальную версию Android (например, API Level 21 — Android 5.0 и выше).

Шаг 2 — Настройте TextMeshPro.

В работе требуется создать 3D-буквы вашего имени и фамилии и расположить их на сцене. Для этого в Unity лучше всего использовать TextMeshPro, так как он обеспечивает высококачественное отображение текста в 3D и UI. Перейдите в меню Window → TextMeshPro → Import TMP Essential Resources. Это добавит шрифты, материалы и примеры использования TextMeshPro.

Шаг 3 — Создайте плоскость (Floor) как пол.

Добавьте объект плоскости: GameObject → 3D Object → Plane, переименуйте объект в "Floor" и настройте его размер с помощью инспектора Transform → Scale :

X: 5, Y: 1, Z: 5 (можно указать другой размер).

Шаг 4 — Создайте 3D-буквы вашего имени и фамилии.

Создайте первый объект 3D-текста: GameObject → 3D Object → TextMeshPro - 3D Text. Объект появится на сцене с надписью "New Text". Переименуйте его в "Letter\_1" (первая буква вашего имени). Затем выберите объект "Letter\_1" в Hierarchy, в инспекторе найдите компонент TextMeshPro - Text и в поле Text замените "New Text" на букву (например, "А"). Настройте параметры: Font Size : ~50 (увеличьте, если буква слишком мелкая), Alignment : Центрируйте текст (иконка "Center" в меню шрифта). Далее продублируйте объект для каждой буквы: выделите объект "Letter\_1" в Hierarchy, нажмите правой кнопкой мыши → Duplicate и переименуйте дубликаты так, чтобы соответствовать порядку букв в вашем имени и фамилии (повторите для каждой буквы). Для каждого дубликата измените текст в инспекторе: например, "Letter\_2" → текст "Б", "Letter\_3" → "В" и т.д.

Шаг 5 — Расположите все буквы случайным образом на полу.

Выберите все объекты с буквами (выделите их в Hierarchy), в инспекторе убедитесь, что у них есть компонент Transform. Измените позицию каждой буквы вручную — в поле Position (X, Y, Z) для каждой буквы задайте случайные значения:

X : от -5 до +5.

Z : от -5 до +5.

Y : оставьте значение по умолчанию (обычно 0, чтобы буквы лежали на полу).

Пример:

Letter\_1 → Position: (2, 0, -3).

Letter\_2 → Position: (-1, 0, 4).

Letter\_3 → Position: (3, 0, 1).

Проверьте визуальное расположение: в окне Scene редактора Unity убедитесь, что буквы не перекрывают друг друга и равномерно распределены на плоскости. Если буква "проваливается" в пол, увеличьте её Y-координату на 0.5–1.

Шаг 6 — Напишите скрипт для вращения букв:

using UnityEngine;

```
public class RotateLetter : MonoBehaviour
{
    public float rotationSpeed = 20f;

    void Update()
    {
        transform.Rotate(Vector3.up * rotationSpeed * Time.deltaTime);
    }
}
```

Перетащите скрипт на все буквы (выделите все объекты с буквами в Hierarchy и перетащите файл скрипта из Project на выделенные объекты).

Шаг 7 — Управление игроком.

Создайте объект игрока: GameObject → 3D Object → Capsule, переименуйте в "Player" и позиционируйте над плоскостью (Transform → Position Y: 1, чтобы не проваливался в пол). Далее добавьте CharacterController: Выберите объект "Player" → Add Component → CharacterController. Настройте параметры:

Radius (радиус): ~0.5 (ширина персонажа).

Height (высота): ~2 (рост персонажа).

Center (центр коллизии): X: 0, Y: 1, Z: 0 (выравнивание по центру).

Затем создайте виртуальные кнопки для движения: GameObject → UI → Button → создайте две кнопки:

- "BtnForward" (текст: "↑", позиция: нижний левый угол экрана).
- "BtnBackward" (текст: "↓", рядом с BtnForward).

Напишите скрипт MobilePlayerMovement.cs:

using UnityEngine;

```
public class MobilePlayerMovement : MonoBehaviour
{
    public float speed = 5f;
    private CharacterController controller;
    private Vector3 movement;

    void Start() => controller = GetComponent<CharacterController>();

    void Update() => controller.Move(movement);

    public void MoveForward() => movement = transform.forward * speed * Time.deltaTime;
    public void StopMoving() => movement = Vector3.zero;
}
```



Затем добавьте скрипт на объект камеры (Main Camera), а в инспекторе привяжите объект "Player" к полю player.

Шаг 8 — Сбор букв.

Добавьте триггер-коллайдер на буквы: выберите объект буквы → Add Component → Box Collider и активируйте Is Trigger в инспекторе.

Создайте скрипт CollectLetter.cs:

```
using UnityEngine;
```

```
public class CollectLetter : MonoBehaviour
{
    private void OnTriggerEnter(Collider other)
    {
        if (other.CompareTag("Player"))
        {
            gameObject.SetActive(false); // Скрывает букву
            ScoreManager.instance.AddPoint(); // Обновляет счет
        }
    }
}
```

Перетащите скрипт на все буквы (выделите все объекты с буквами и добавьте скрипт один раз).

Затем создайте скрипт ScoreManager.cs:

```
using UnityEngine;
```

```
using TMPro;
```

```
public class ScoreManager : MonoBehaviour
{
    public static ScoreManager instance;
    public TextMeshProUGUI scoreText;
    private int score = 0;

    void Awake() => instance = this;

    public void AddPoint()
    {
        score++;
        scoreText.text = "Очки: " + score;
    }
}
```

Шаг 9 — Создайте UI-панель для отображения очков.

- Выберите GameObject → UI → Canvas → Panel.
- Добавьте текст: GameObject → UI → TextMeshPro - Text.
- В инспекторе задайте текст: "Очки: 0" .
- Создайте пустой объект ScoreManager → добавьте скрипт ScoreManager → привяжите scoreText из UI.

Шаг 10 — Таймер и интерфейс.

Добавьте текст для таймера: в том же Canvas создайте ещё один TextMeshPro - Text с надписью "Время: 60". Затем создайте скрипт Timer.cs:

```
using UnityEngine;
using TMPro;
```

```
public class Timer : MonoBehaviour
{
    public float timeLeft = 60f;
    public TextMeshProUGUI timerText;
    public GameObject gameOverPanel;

    void Update()
    {
        if (timeLeft > 0)
        {
            timeLeft -= Time.deltaTime;
            timerText.text = "Время: " + Mathf.Round(timeLeft);
        }
        else
        {
            timerText.text = "Время вышло!";
            gameOverPanel.SetActive(true);
        }
    }
}
```

Далее создайте панель Game Over: GameObject → UI → Panel → добавьте текст "Игра окончена!". Привяжите панель к полю gameOverPanel в скрипте Timer.

Шаг 10 — Тестирование на устройстве.

- Перейдите в File → Build Settings → Android → Switch Platform.
- Подключите Android-устройство по USB.
- Нажмите Build and Run для установки игры на устройство.
- Протестируйте управление кнопками и вращение камеры, сбор букв и обновление счета, а также работу таймера и завершение игры.

### **Требования к отчету**

Результаты выполненной лабораторной работы должны быть оформлены в формате текстового редактора Word, размером шрифта 14 пунктов и включать:

1. Титульный лист.
2. Текст задания, соответствующий Вашему варианту.
3. Описание архитектуры проекта (структура папок Assets/Scripts, Prefabs, Scenes, Materials, используемые скрипты).
4. Скриншоты сцены с буквами и игроком, а также скриншоты интерфейса на экране мобильного устройства.
5. Программный код с комментариями.
6. Выводы о том, что удалось реализовать, возможные доработки.

7. Ссылки на источники внешней информации, которые были использованы при выполнении заданий.

Объем пояснительной записки должен включать в себя не менее 5 страниц поясняющего текста (не считая исходного кода программ).

#### **Список дополнительных источников**

1. Мэннинг, Дж., Баттфилд-Аддисон, П. Unity для разработчиков: создание 2D и 3D игр / Пер. с англ. — СПб.: Питер, 2018. — 464 с. ISBN 978-5-4461-0541-0 (<https://disk.yandex.ru/i/iXhKcC3w0D7IRQ>)