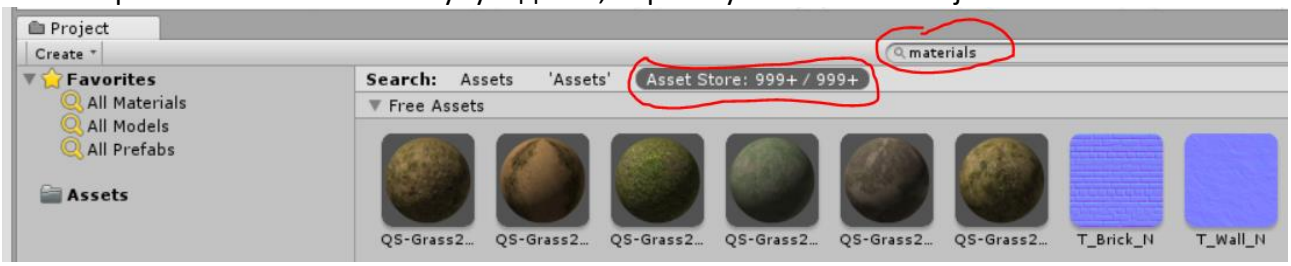
Лабораторная работа № 1

Тема: Знакомство с Unity3d. Rigidbody. Коллайдер.

1. Создайте новый 3d проект руководствуясь документацией https://docs.unity3d.com/ru/540/Manual/GettingStarted.html
2. Разместите на сцене 3d объекты: плоскость и куб.
3. Изучите способы навигации по сцене (обход вокруг объекта, перемещение по сцене): https://docs.unity3d.com/ru/540/Manual/SceneViewNavigation.html
4. Опробуйте на кубе инструменты трансформирования: https://docs.unity3d.com/ru/540/Manual/PositioningGameObjects.html
5. Сохраните сцену, она появиться в окне Project.
6. Добавьте еще один куб и сделайте из него плоскую доску (инструменты масштабирования или параметры Scale в окне Inspector). Созданную доску разместите под наклоном.
7. Создайте **материалы с различными текстурами** (например, камня и дерева). Можно воспользоваться возможностями поиска в окне Project на AssetStory см.рис. Назначьте материалы соответственно кубу и доске, перетянув их из окна Project на объект.



8. Создайте сферу. Назначьте ей материал с каким-либо цветом без текстуры: https://docs.unity3d.com/ru/540/Manual/Materials.html
9. Разместите цветную сферу на верхнем конце наклонной доски.
10. В окне Inspector добавьте сфере компонент **Rigidbody** (твердое тело, подчиняющееся законам физики). При помощи кнопки Play на панели инструментов запустите приложение и посмотрите на поведение сферы.
11. Создайте еще одну сферу. Разместите ее над поверхностью. Сделайте так, чтобы при запуске она упала на поверхность.
12. Для отскакивания сферы от поверхности назначьте ей упругий **физический материал**. https://docs.unity3d.com/ru/540/Manual/class-PhysicMaterial.html
Опробуйте различные режимы комбинации упругости.
13. Создайте **префаб** со сферой (ЛКМ в окне Project). Информация о префабах по ссылке: https://docs.unity3d.com/ru/540/Manual/Prefabs.html
14. «Вытяните» из префаба на сцену множество сфер, назначьте отдельным экземплярам сфер различное значение сопротивления воздуха при падении (поле Drag в компоненте Rigidbody). Запустите сцену.
15. Создайте куб подчиняющееся законам физики.
16. Деактивируйте компонент Collider, который для куба называется BoxCollider (**коллайдер** - это границы твердого тела.). Запустите сцену. Объясните, что происходит.
17. Создайте рядом второй куб, увеличьте размеры его коллайдера (кнопка Edit Collider). Запустите сцену. Объясните, что происходит.

Часть 2: Создание скриптов. Переменные. Обращение к компонентам. Работа с методами интерфейса Input.

1. *Работа с консолью Unity*

Вывод в консоль:
Debug.Log("...") или print("...")

Задание 1: в окне **Project** при помощи контекстного меню создайте новый скрипт. По умолчанию в скрипте определены функции Start и Update. В функции Start напишите вывод в консоль сообщения "Hello World", а в функции Update напишите вывод в консоль любого другого сообщения. Т.к. выполняются только те скрипты, которые присоединены к игровому объекту, то наш скрипт надо привязать к какому-либо объекту. Создайте на сцене пустой объект Create>Empty и перетяните на него написанный скрипт. Перейдите в консоль и запустите режим игры.

2. *Перемещение объекта*

Следующая запись мгновенно перемещает объект в новую точку, т.е. задается новая позиция:

```
gameObject.GetComponent<Transform>().position = new Vector3(2,2,2)
                                        можно записать в кратком виде
transform.position= new Vector3(2,2,2)
```

*Для плавного перемещения необходимо изменять координаты объекта на какое-либо приращение допустим каждый кадр. Для обозначения этого приращения объявим переменную **speed** типа **public**, чтобы иметь возможность изменять ее во время режима игры.*

Задание 2: создайте скрипт следующего вида и присоедините его к кубу на сцене. При запуске вводите различные значения переменной speed на панели **Inspector**, чтобы добиться нужной скорости.

```
public class Cube_position : MonoBehaviour {
    public float speed;

    void Update () {
        float posX = transform.position.x;
        float posY = transform.position.y;
        float posZ = transform.position.z;

        transform.position = new Vector3 (posX + speed, posY, posZ);
    }
}
```

3. *Обращение к другим объектам*

Один из способ - это связывание объектов через переменные

Добавим в скрипт переменную типа GameObject с уровнем доступа public:

```
public class Enemy : MonoBehaviour {
    public GameObject player;

    // Other variables and functions...
}
```

Поле для ввода переменной появится в окне Inspector, и теперь нужно перетащить объект со сцены или из панели Hierarchy в это поле в окне Inspector. Теперь функция GetComponent и доступ к переменным компонента доступны для связанного объекта:

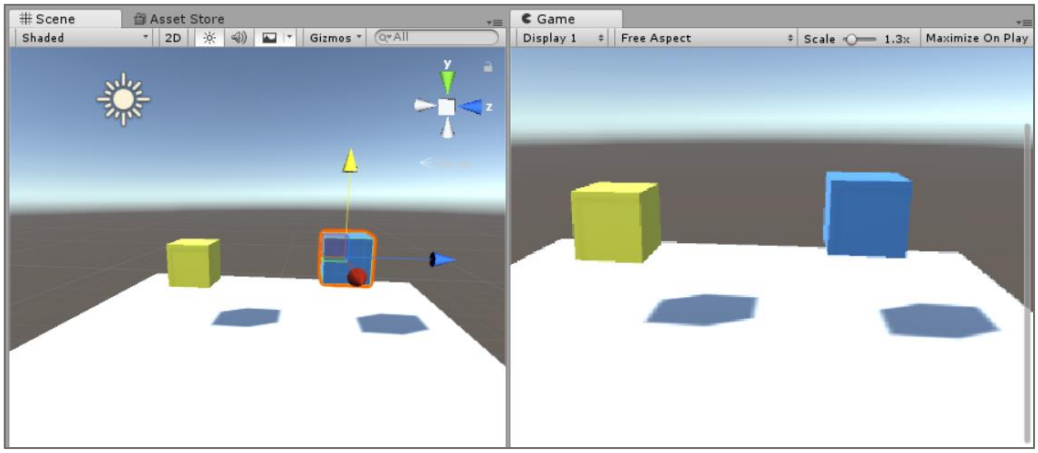
```
public class Enemy : MonoBehaviour {
    public GameObject player;

    void Start() {
        // Start the enemy ten units behind the player character.
        transform.position = player.transform.position - Vector3.forward * 10f;
    }
}
```

Vector3 – структура, которая манипулирует тремя числами.

Задание 3: Опробуйте приведенный ниже скрипт. Для этого разместите на сцене два куба, одному назначьте следующий скрипт, а другой куб задайте в качестве переменной. Параметр Time.deltaTime используется для выполнения программы с одинаковой скоростью на разных устройствах.

```
public class moveOther : MonoBehaviour {
    public GameObject player;

    void Update () {
        transform.position = Vector3.MoveTowards(transform.position,player.transform.position,Time.deltaTime);
    }
}
```

Для выполнения этого примера расположите окна Scene и Game как на рисунке, и в процессе выполнения двигайте куб-цель в окне Scene за оси.



4. *Если у объекта есть компонент **Rigidbody**, то задавать ему движение нужно с помощью приложения силы в заданном направлении **AddForce()**, передав таким образом все расчеты движения физическому движку игры:*

   `GameObject.AddForce(0,0,100)`

   справка: https://docs.unity3d.com/ru/530/ScriptReference/Rigidbody.AddForce.html

Задание 4. Следующий скрипт двигает объект стрелками на клавиатуре. На сцене разместите плоскость и сферу. Назначьте сфере приведенный ниже скрипт.

- Описание класса **Input** и метода **GetAxis** смотрите по ссылке https://docs.unity3d.com/ScriptReference/Input.GetAxis.html
- Информация по **Input Manager** https://docs.unity3d.com/Manual/class-InputManager.html

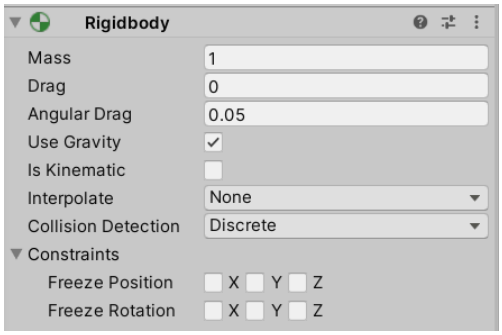
```
public class PlayerController : MonoBehaviour {

    public float speed = 5;
    private Rigidbody rb;

    void Start()
    {
        rb = GetComponent<Rigidbody>();
    }

    void Update()
    {
        float moveHorizontal = Input.GetAxis("Horizontal");
        float moveVertical = Input.GetAxis("Vertical");

        Vector3 movement = new Vector3(moveHorizontal, 0.0f, moveVertical);
        rb.AddForce(movement * speed);
    }
}
```

Добавьте в скрипт прыжок по нажатию на пробел

5. *Камера следует за объектом*

Самый простой способ назначить камере слежение за объектом – это сделать камеру дочерней по отношению к объекту слежки.

Задание 5. Назначьте камере сопровождение нашей сферы, которая управляется клавишами. Для этого в окне Hierarchi переметите камеру в сферу, сделав камеру дочерним объектом и разместите камеру на некотором расстоянии позади и выше сферы. Запустите игру.



Если результат вас не удовлетворил (например, камера крутится вместе со сферой), то можно запретить вращение FreezeRotation вокруг определенных осей в компоненте Rigidbody.

6. *Обращение к компонентам объекта*

*Для обращения к компонентам GameObject необходимо получить ссылку на экземпляр компонента, с которым вы хотите работать. Это делается с помощью функции GetComponent. Пример обращения к компоненту **Rigidbody** для изменения массы объекта через его свойство **mass:***

```
void Start () {
    Rigidbody rb = GetComponent<Rigidbody>();

    // Change the mass of the object's Rigidbody.
    rb.mass = 10f;
}
```

Задание 6. В созданную в предыдущем 4-м задании сцену добавьте возможность изменения цвета сферы по нажатию на произвольную клавишу. Доступ к свойству color осуществляется через свойство material компонента Renderer.

Функции обработки действий с клавиатуры находится в интерфейсе Input (https://docs.unity3d.com/ScriptReference/Input.html).

Задание 7. Добавьте в сцену объект, который будет «догонять» сферу, т.е. будет отслеживать позицию сферы и перемещаться в эту позицию.

Задание 8*

- ✓ Реализуйте следующий скрипт. Назначьте его плоскости (Plane).
- ✓ Объясните код.

- ✓ Добавьте генерацию сфер по нажатию на ЛКМ (Input.GetMouseButtonDown(0)). Сферы должны генерироваться на различной высоте и иметь рандомный цвет.
- ✓ Оптимизируйте скрипт, переместите в ф-ю **Start** те строки кода, которые по логике не должны быть в **Update**.

```
public class CubeGenerator : MonoBehaviour {

    void Update () {
        MeshRenderer render = gameObject.GetComponent<MeshRenderer> ();

        float minX = render.bounds.min.x;
        float maxX = render.bounds.max.x;
        float minZ = render.bounds.min.z;
        float maxZ = render.bounds.max.z;

        float newX = Random.Range (minX, maxX);
        float newZ = Random.Range (minZ, maxZ);
        float newY = gameObject.transform.position.y + 5;

        if (Input.GetKeyDown (KeyCode.Space)) {
            GameObject cubb = GameObject.CreatePrimitive (PrimitiveType.Cube);
            cubb.transform.position = new Vector3 (newX, newY, newZ);

            cubb.AddComponent<Rigidbody> ();

        }

    }
```