**Тема 9. Программное создание и обработка объектов на сцене.**

1. Создать новый проект и добавить на сцену 3D-объект типа **Plane** – горизонтальную плоскость.
2. Создать для объекта **Plane** программный код, генерирующий по нажатию на клавишу, например W, клавиатуры рождение на сцене нового объекта над плоскостью и падающего на нее с некоторой высоты (добавить для плоскости компоненту **Rigidbody**, убрать в ней гравитацию **Gravity** и установить режим **Is Kinematic**):

* в новом классе перед функцией **Start()** определить переменные типа **float** для определения координат точек краев горизонтальной плоскости и координат генерируемого объекта, например: **minX, maxX, minZ, maxZ**;
* в функции **Start():**

а) определить переменную, например **rend**, типа **MeshRenderer** через компонент игрового объекта (это в данном случае плоскость) **gameObject.GetComponent<MeshRenderer>(),**

б) задать значения для координат краев плоскости через свойства **bounds, min, max** для этого объекта, например **rend.bounds.min.x**, **rend.bounds.min.x** и т. д.,

в) задать значение для высоты (координата по оси **Y**), на которой будет генерироваться объект над плоскостью, например, **gameObject.transform.position.y+5**;

* в функции **Update():**

а) задать последовательно значения для координат по осям **X** и **Z** сгенерированного объекта через функцию случайного числа в пределах краев плоскости, например **Random.Range(minX,maxX),**

б) определить условие для оператора **if,** нажата ли определенная клавиша на клавиатуре, например **Input.GetKeyDown(KeyCode.Space)** – в данном случае выбрана клавиша пробела **Space**,

в) задать для условного оператора действия при выполнении условия:

- генерация нового 3D-объекта для значения переменной типа **GameObject**, например, с именем **cub**, следующей командой **GameObject.CreatePrimitive( PrimitiveType.Cube)**, в данном случае 3D-объекта в виде куба,

- задание позиции генерируемого объекта **cub** с помощью конструктора из класса **Vector3** с заданными ранее координатами по осям **X, Y, Z**;

- добавление объекту **cub** компоненты твердого тела с действием гравитации **cub.AddComponent<Rigidbody>().**

*Запустить приложение и сгенерировать нажатием заданной в п. 2 клавиши несколько десятков кубов, при этом некоторые из них после столкновений будут «сваливаться» с плоскости.*

1. Добавить в функцию **Update()** запускнаклона плоскости вокруг оси **Z**, например при нажатии клавиши **W**, и «сбросить» с нее оставшуюся на плоскости горку кубиков.
2. Создать в этой же сцене объект3D-объект типа **Sphere** ик немудобавить компоненту **Rigidbody**, оставив все параметры по умолчанию.
3. Создайте новый скрипт с названием **Creator**, в котором определить новую публичную переменную **public GameObject prefub**, в которой будет храниться префаб (другими словами шаблон) сферы.
4. В методе **Update()** создайте условный оператор, проверяющий событие нажатия на клавишу, например, пробела (Space), в котором:

* определитеразмер области, в которой будут создавать новые экземпляры префаба сферы по всем трём осям случайным образом:

**Vector3 position = new Vector3 (Random.Range(-5.0f, 5.0f), Random.Range (1f, 10.0f), Random.Range (-5.0f, 5.0f));**

* добавьте в условие метод **Instantiate()**, который позволяет создавать экземпляр нового объекта из префаба, указав ссылку на префаб, место создания экземпляра и угол его поворота (в этом примере угол идентичен углу в префабе):

**Instantiate(prefub, position, Quaternion.identity)**

1. Создайте для объекта **Sphere** новый скрипт с названием, например, **Destroyer**, в котором:

* определите новую публичную переменную **public** **float lifeTime = 10f** для хранения времени жизни нашей сферы (здесь время жизни задано на 10 кадров).
* в методе **Update()** создайте условие, если время жизни объекта больше нуля, то его необходимо уменьшить. Если же время жизни объекта меньше или равно нулю, то необходимо объект удалить со сцены, выполнив метод **Destruction()**:

**if (lifeTime > 0) {  
            lifeTime -= Time.deltaTime;  
            if (lifeTime <= 0)  
            { Destruction(); }        }**

* после метода **Update()** создайте новый метод **Destruction()**, который удаляет объект, на котором находится данный скрипт:

**void Destruction()  {  Destroy(this.gameObject);  }**

1. Перетяните объект **Sphere** из окна **Hierarchy** в окно **Project** в папку **Assets**, таким образом у вас получится префаб (шаблон) сферы. Из окна **Hierarchy** удалите объект **Sphere** (его не должно быть на сцене, он будет рождаться каждый раз на сцене программно из префаба – шаблона при нажатии на заданную клавишу Space).
2. Разместите скрипт **Creator** на объекте **Main Camera** перетягиванием его мышью из панели **Assets** на объект камеры в окне **Hierarchy**. В появившееся для объекта **Main Camera** в окне **Inspector** поле **Prefub** перетяните из окна **Project** папки **Assets** созданный ранее префаб. Запустите проект и нажимая на пробел создайте несколько сфер.
3. Создайте **Physic Material** нажав в окне окна **Project** папки **Assets** и выполните команду **Create/Physic Material** и установите **Bounciness** = 1 (прыгучесть). Перетяните данный материал на созданный папке **Assets** префаб сферы и проверьте, как будут вести себя сгенерированные сферы на плоскости при запуске проигрывания сцены **Play**. Попробуйте изменить другие параметры данного материала и проанализируйте полученные изменения.
4. Сгенерируйте кубики и сферы на плоскости, наклоните плоскость нажав клавишу **W** для сброса с нее всех объектов. *Для этого предварительно добавить в функцию* ***Update()*** *запуск**наклона плоскости вокруг оси* ***Z****, например при нажатии клавиши* ***W****, и «сбросить» с нее оставшуюся на плоскости горку кубиков (****!!!*** *предварительно необходимо для плоскости добавить компоненту* ***Rigidbody****, убрать в ней гравитацию* ***Gravity*** *и установить режим* ***Is Kinematic****).*
5. Сохраните сцену под именем «**Генерация объектов**».