**Тема 13. Выстрелы танка снарядами и обработка попадания их в цель**

1. В созданном ранее проекте **Unity3D** с управлением движения танка **Тема 12** создайте новую сцену на основе существующей, дав ей новое имя, например, «**выстрелы**» и добавьте на сцену новый объект – снаряд с именем, например, **core** на основе примитива типа **Sphere** или **Capsule** соответствующих размеров для ствола танка.
2. Добавьте к снаряду **core** «физику» командой **Add Component/Physics/Rigidbody**. При необходимости добавьте ядру массу в инспекторе в свойствах **Rigidbody**, а также поставьте галочку **is Kinematic**, чтобы *снаряд мог воздействовать на объекты на сцене, а на него воздействие могло оказываться только из кода*. Наложите на снаряд соответствующую текстуру или цвет.
3. Создать **prefab** снаряда, действуя аналогично **Теме 9** – перетянуть его мышью со сцены в папку **Assets** и удалить его со сцены. *В дальнейшем для префаба снаряда добавим еще скрипт его полета и взаимодействия с целями на сцене при его попадании*.
4. Добавить для **ствола** танка новый скрипт, в котором, прежде всего, объявить публичную объектную переменную типа **GameObject** для ссылки на префаб cнаряда и в скрипте ствола в **Инспекторе** выбрать объект в списке **Assets** для связи его с объявленной переменной, нажав на «кружочек» справа от имени объектной переменной.
5. В метод **Update()** добавьте условие для проверки нажатия определенной клавиши на клавиатуре, которая не задействована для управления танком (например, пробел Space) для выполнения выстрела.
6. В условии нажатия клавиши необходимо записать код для двух действий:

* определить вектор **Vector3**  точки, в которой нужно разместить экземпляр снаряда относительно точки привязки ствола откуда он должен вылетать – нужно добавить к этой точке смещение вперед на длину ствола в соответствии с вашей моделью танка (иначе он будет вылетать из точки привязки объекта ствола танка – там, где он присоединен к башне);
* сгенерировать экземпляр снаряда из префаба в этой точке методом **Instantiate().**

1. Добавьте новый скрипт на **префаб** снаряда в **Assets** для управления поведением снаряда, выполняющий следующие действия:

* определите переменную типа **float** для задания скорости снаряда;
* в методе **Update()** добавьте постоянное перемещение снаряда вперед с заданной скоростью.

1. Запустите игру и проконтролируйте соответствие полета снарядов из разных точек сцены, различных поворотах танка и его башни, их попадания в объекты-цели с автоматической обработкой столкновений коллайдеров снаряда и целей с их отталкиванием просто как твердых тел.
2. После выстрелов можно заметить, что снаряды постоянно находятся на сцене и не удаляются, даже если отлетели уже на большое расстояние и по-прежнему остаются в памяти, что плохо для производительности компьютера (особенно, если вы будете стрелять «очередями»). Чтобы исправить данный факт, в код снаряда в методе **Start()** добавьте метод **Destroy()**,который будет уничтожать снаряд через заданное количество секунд.
3. Для дополнительной обработки (кроме их простого «расталкивания») столкновений снаряда с объектами на сценедобавить в скрипт префаба снаряда код на основе функции **OnCollisionEnter(),** при этом задать для всех объектов-целей один и тот же маркер-тег, например, **goal**, для чего в **Инспекторе** под именем каждого объекта-цели в строке **Tag** вести выбранное имя тега. *При этом все объекты будут обнаруживаться по имени тега, а не по их именам name, которые у всех объектов обязаны быть разными. Это позволит все такие объекты обрабатывать одним и тем же программным кодом.*
4. Добавьте в код скрипта префаба в функцию **OnCollisionEnter()** скрытие снаряда после столкновения с объектом с использованием для компоненты визуализации объекта **GetComponent<Renderer>** свойства **enabled** со значением **false**.
5. Добавьте в код скрипта префаба снаряда публичную объектную переменную, например, **explosion1** для обращения к имеющемся в папке **Assets** проекта в разделе **Standard Assets/ParticleSystems/Prefabs** различным префабам с эффектами, например, к эффекту взрыва **Explosion**. После задания имени для публичной объектной переменной свяжите ее в **Инспекторе** в скрипте префаба снаряда с этим объектом в списке **Assets**, аналогично тому, как это делалось в п. 4 инструкции.
6. Добавьте также в код скрипта префаба снаряда в функцию **OnCollisionEnter()** в обработку условия попадания снаряда в цель генерацию эффекта взрыва **explosion1** в точке столкновения с использованием метода **Instantiate()**.
7. Запустите игру и проконтролируйте реакцию целей на попадание в них снарядов с эффектом взрыва.