Лабораторная работа №10

**Коллайдер**

Коротко коллайдер-это границы твердого тела.

*Задание 1*

*1. Создайте куб и поверхность. Куб должен падать на поверхность.*

*1.1 Деактивируйте компонент BoxCollider (отключите чекбокс у компонента). Запустите игру.*

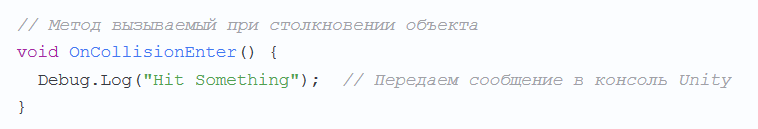
*1.2 Создайте рядом второй куб, увеличьте размеры его коллайдера (кнопка Edit Collider). Запустите игру.*

**Обнаружение столкновений (collisions)**

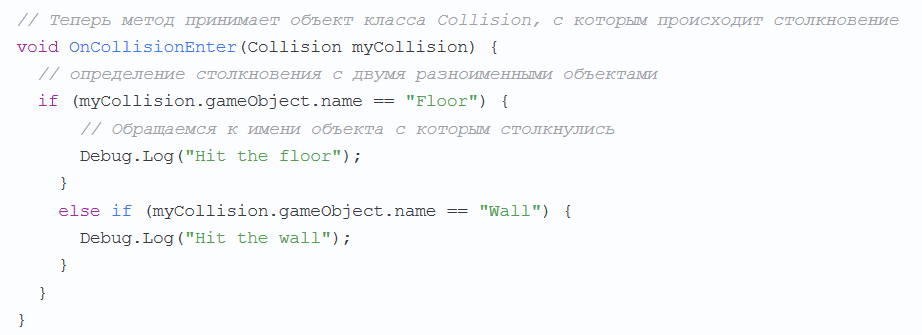
Физический движок сообщит о столкновениях с объектом с помощью вызова функций событий в скрипте этого объекта. Функции **OnCollisionEnter**, **OnCollisionStay** и **OnCollisionExit** будут вызваны по началу, продолжению и завершению контакта.

В функцию передаётся параметр, предоставляющий дополнительную информацию о столкновении (координаты, “личность” входящего объекта и т.д.).

Пример показывает обнаружение столкновений



Пример показывает идентификацию объектов, с которыми произошло столкновение по параметру myCollision



*Задание 2*

*2.1 Создать пол и две стены, в окне Hierarchy дайте этим компонентам соответствующие имена. Расположить над полом куб, слегка повернуть его, что бы при падении он падал на грань или угол, придать ему упругость.*

*2.1. Запустите игру, куб должен ударяться в пол и стены*

*2.2 Реализуйте приведенный ранее пример с выводом сообщений в консоль.*

**Основы движения с помощью силы**

Если на сцене есть объект с компонентом Rigidbody, то задавать ему движение нужно с помощью приложения силы (передав таким образом все расчеты движения физическому движку игры).



*Задание 3*

*3.1 Разместите на сцене несколько кубов один над другим на некотором расстоянии друг от друга (как бы парящими в воздухе). Назначьте им компонент Rigidbody, но сделайте так, чтобы на них не действовала сила гравитации. Запустите игру, кубы не должны падать.*

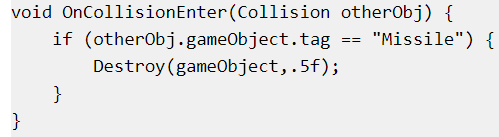
*3.2 Назначьте кубам вращение (через скрипт). Итог – кубы парят в воздухе и вращаются.*

*3.3 Назначьте верхнему кубу скрипт, который при соприкосновении с объектом придаст объекту силу в горизонтальном направлении.*

*3.4. Включите у верхнего куба гравитацию, запустите игру. Кубы должны вылетать из-под верхнего в сторону.*

**Уничтожение объекта**

Функция **Destroy** уничтожит объект после того, как загрузка кадра будет завершена или опционально после короткой паузы:



Второй параметр является не обязательным и указывает время, спустя которое будет уничтожен объект.

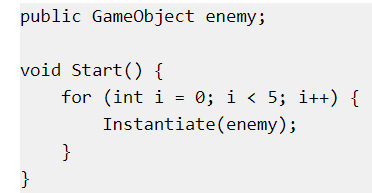
Функция **Destroy** может уничтожать отдельные компоненты без влияния на сам объект. Частая ошибка - писать Destroy(this) это на самом деле уничтожит только вызывающий скриптовый компонент, вместо того, чтобы уничтожить игровой объект, к которому присоединен этот скрипт.

*Задание 4*

*4.1 Создайте сцену с кубами, как в предыдущем задании, только теперь при соприкосновении верхнего куба с нижними, нижние кубы должны исчезнуть.*

**Создание объекта**

Игровой объект (GameObject) может быть создан при использовании функции **Instantiate**, которая делает копию существующего объекта:



Объект с которого берется копия не обязан присутствовать на сцене. Гораздо чаще используется префаб, который был перетащен на открытую переменную (public variable) в панели Project. Также, копируя игровой объект (GameObject), вы копируете все компоненты оригинального объекта.

*Задание 5*

***Придумайте свою игру*** *с применением изученных функций (создание объекта, уничтожение, отслеживание соприкосновений, движение с помощью добавления силы и т.д.)* ***или реализуйте предложенный вариант*** *игры.*

***Игра:***

*Создайте поверхность с бортиками. На неё поместите управляемую клавишами сферу из л.р. 3. Разместите на поверхности голубые и зеленые кубы из префабов. При соприкосновении сферы с кубами, голубые кубы должны взлетать вверх, а зеленые должны исчезать. Разместите один красный куб, при соприкосновении с которым исчезнет сфера.*