**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский политехнический университет»**

Методическое пособие

**«Создание камеры для вращение вокруг детали в Unity»**

Выполнил: Борисенко Д.С.

Москва, 2021

**Методические указания по созданию камеры для вращения вокруг объекта в Unity**

**Шаг 1: Добавляем необходимую деталь(объект) на сцену**

Для этого необходимо в нижней части экрана, в файловом проводнике Unity, нажать **ПКМ** и выбрать **Import New Asset,** после чего вы сможете добавить свой объект с расширением .**obj**.

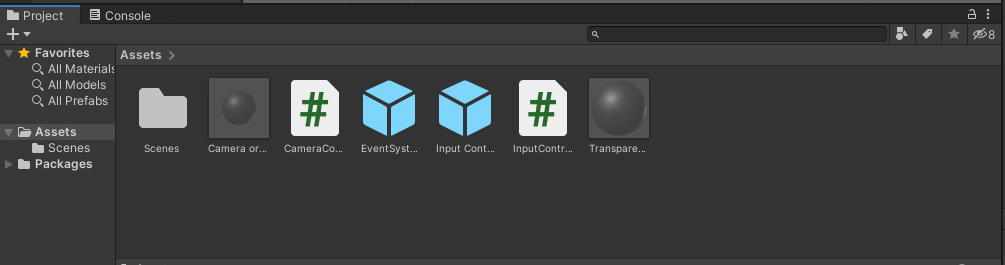


Рис. 1 Файловый проводник Unity

**Шаг 2: Расмещаем деталь в начале координат**

Для этого необходимо выделить деталь в дереве проекта.

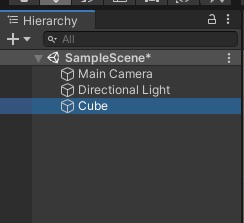


Рис. 2 Дерево проекта

Затем во вкладке **Inspector** найти раздел **Transform**, в котором у свойства **Position** выставить нулевые значения во всех доступных полях

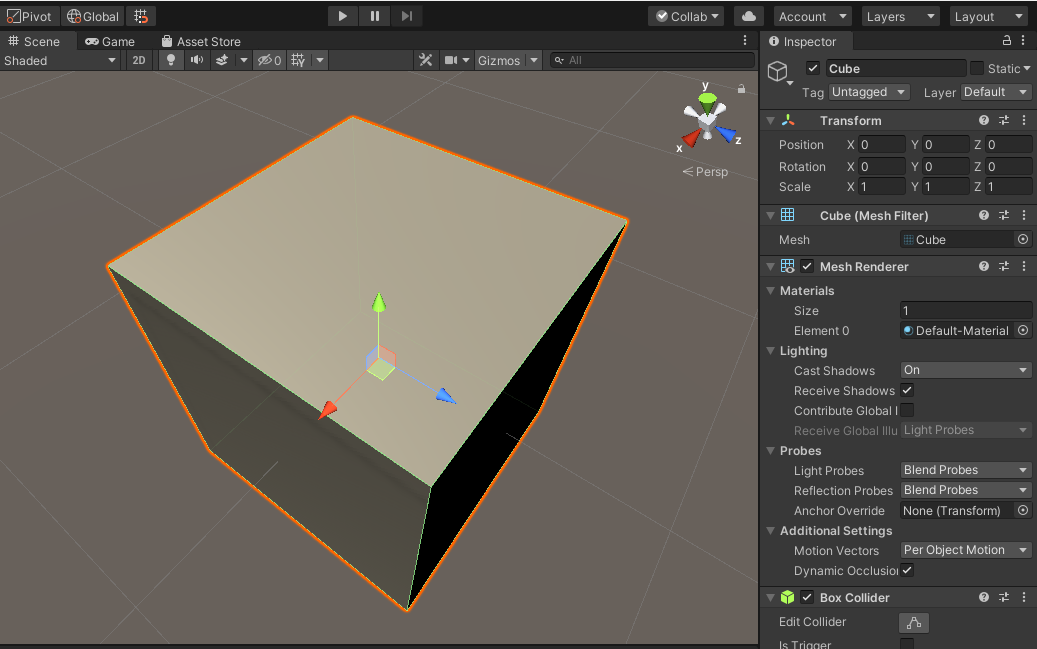


Рис. 3 Вкладка Inspector

**Шаг 3:** **Добавляем орбиту, по которой будет вращаться камера**

Для этого в дереве проекта(рис. 2) нажимаем на “**+**”, выбираем **3D Object – Sphere.**

По желанию, переименовываем данную сферу. Например, **Camera orbit**

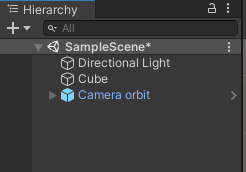


Рис. 4 Новые объект - Camera Orbit

Затем выбираем эту сферу и во вкладке **Inspector**, в разделе **Transform** выставляем у свойства **Scale** такие параметры, что сфера полностью покроет Вашу деталь.

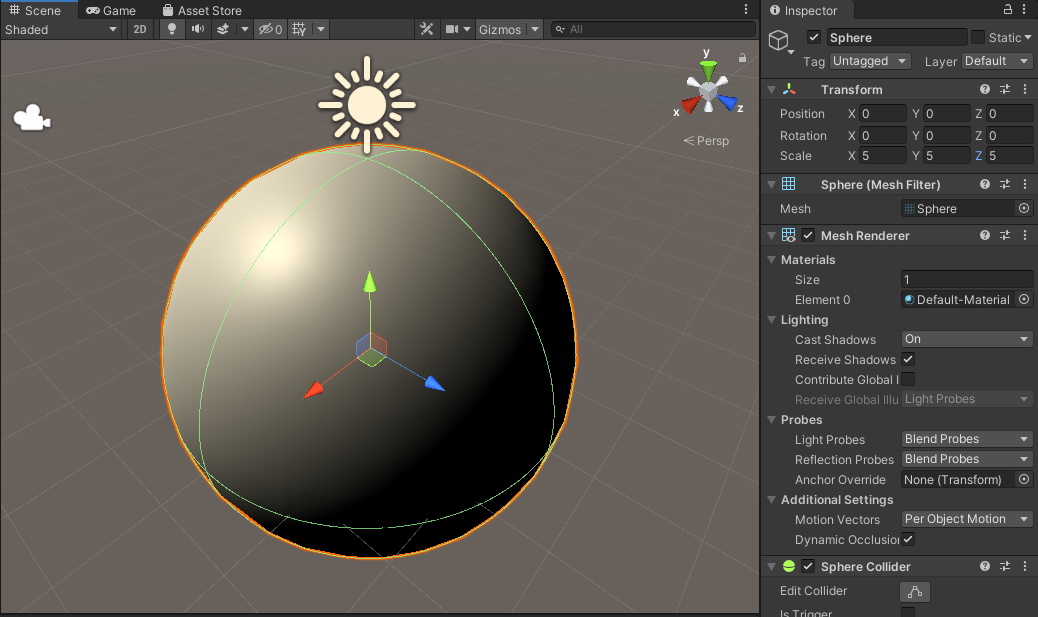


Рис. 5 Изменение параметра Scale

**Шаг 4: Делаем орбиту прозрачной**

Для этого в нижней части экрана, в файловом проводнике **Unity** нажимаем **ПКМ** и выбираем **Create** – **New** **Material.** Затем во вкладке Inspector у данного материала выставляем следующие характеристики

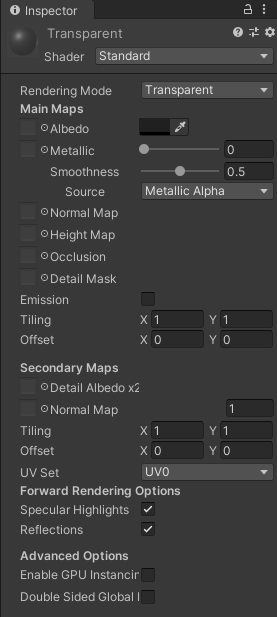


Рис. 6 Необходимые свойства для создания прозрачного материала

**Шаг 5: Добавляем камеру, как потомка в Camera orbit**

Для этого в дереве проекта выделяем объект **Main Camera**, зажимаем **ЛКМ** и перетаскиваем этот объект на **Camera orbit**. Таким образом, камера будет находиться в системе координат орбиты.

После этого настраиваем **Main Camera** таким образом, чтобы камера захватить весь объект целиком. Для упрощения данной работы ориентируйтесь на **Camera Preview** и на угол обзора камеры, выделенный белыми линиями.

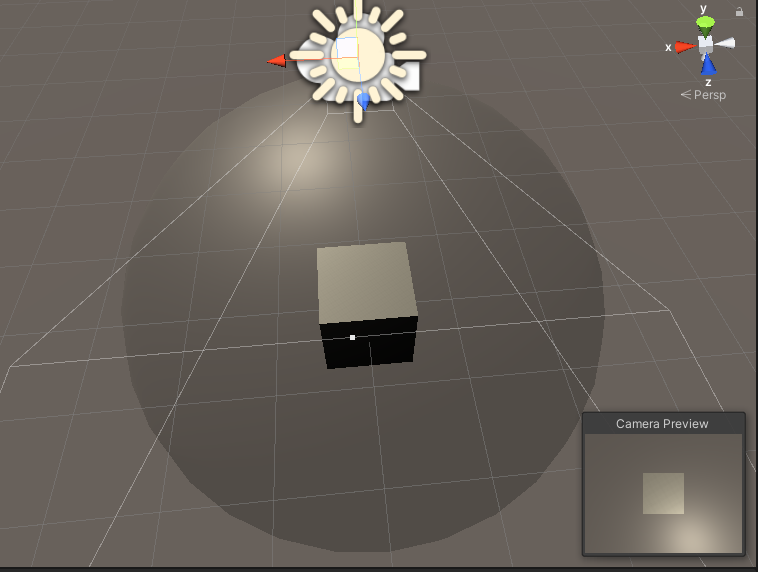


Рис. 7 Отображение камеры в Unity

Переместить камеру можно выбрав ее в дереве проекта и затем с помощью направляющих перетащить камеру в необходимую позицию. Либо же в Inspector выбрать раздел **Transform** и выставить нужные координаты у свойства **Position**

Изменить угол обзора можно во вкладке **Inspector**, изменяя значения у свойства **Field** **of View**

Изменить максимальный и минимальный диапазон, который видит камера, можно поменяв во вкладке **Inspector** значения у свойств **Far** и **Near** соответственно

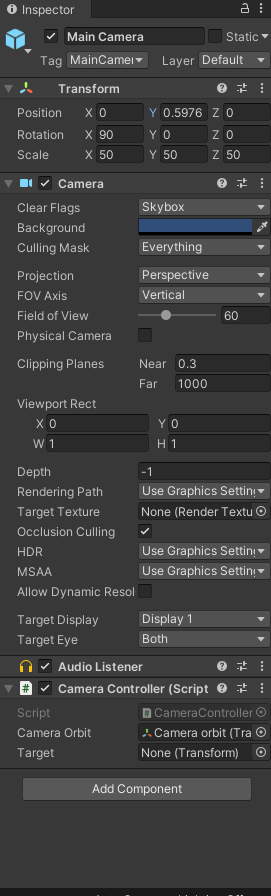


Рис. 8 Свойства камеры во вкладке Inspector

**Шаг 6: Создаем пустой GameObject для контроля перемещений**

Для этого в дереве проекта нажимаем “+” и выбираем **Create Empty**. По желанию переименовываем его. В данном пример назовем его **Input Control**

Данный объект будет отвечать за передачу изменений в камеру

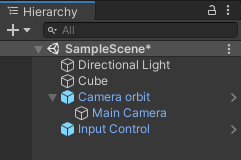


Рис. 9 Добавление нового объекта Input Control

**Шаг 7: Создаем скрипт для работы Input Control**

Данный скрипт даст возможность объекту Input Control реагировать на вращение камеры с помощью мыши

Для создания скрипта необходимо в файловом проводнике нажимаем **ПКМ** и выбираем **Create – C# Script**. Затем даем скрипту название, например **InputControl**.

После того как скрипт был создан необходимо дважды нажать по нему **ЛКМ**. Начнется запуск **Visual Studio**. Затем необходимо добавить следующий код:



Рис. 10 Листинг кода для InputControl.cs

**Шаг 8: Добавляем InputControl.cs в скрипты Input Control**

В данный скрипт **InputControl.cs** и объект **Input Control** никак не связаны. Для того, чтобы это исправить необходимо:  
 a. Выбрать объект **Input Control** в дереве проекта

b. Во вкладке Inspector нажать кнопку **Add Component**

c. В выпадающем списке выбрать **Scripts – Input Control**

d. Выбрать у свойства **Camera Orbit** объект **Camera Orbit**, который является орбитой камеры в проекте

**Шаг 9: Создаем скрипт для работы камеры**

На данном этапе система уже может передавать данные камере для поворота, но поскольку в самом объекте камеры отсутствует какая-либо реакция, то ничего не произойдет. Для того чтобы это исправить необходимо:

a. В файловом проводнике **Unity** нажать **ПКМ** и выбрать **Create – C# Script.** Затем дать скрипту имя, например **CameraController.**

b. Дважды нажать **ЛКМ** по созданному скрипту и перейти в **Visual Studio.**

c. Добавить следующий код

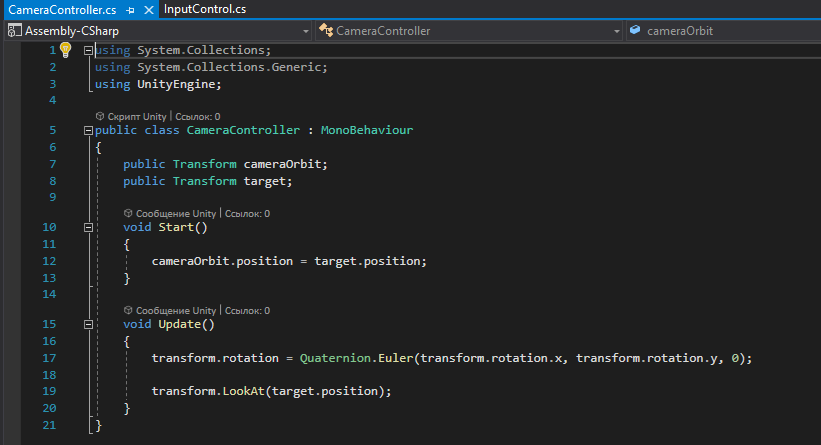


Рис. 11 Листинг кода для CameraContoroller.cs

d. Нажать кнопку **Сохранить всё** и закрыть **Visual Studio**

**Шаг 10: Добавляем CameraController в скрипты камеры**

На данном этапе **CameraController.cs** и **Main Camera** не имеют никакой связи. Следовательно, камера не будет вращаться. Чтобы это исправить необходимо:

a. В дереве проекта выбрать объект Main Camera

b. Во вкладке Inspector нажать кнопку Add Component

c. В выпадающем списке выбрать Scripts – Camera Controller

d. В свойстве Camera Orbit выбрать Camera Orbit

e. В свойстве Target выбрать деталь(объект), вокруг которой будет происходить вращение камеры

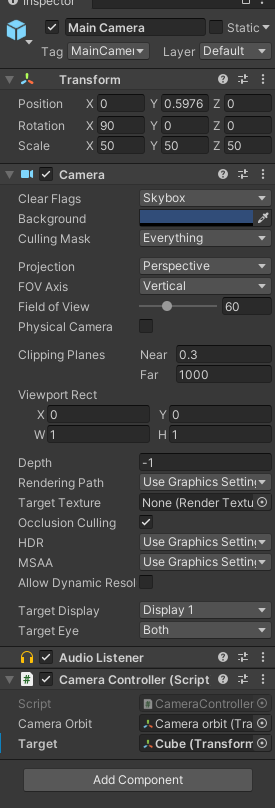


Рис. 12 Итоговые свойства Main Camera