

unity

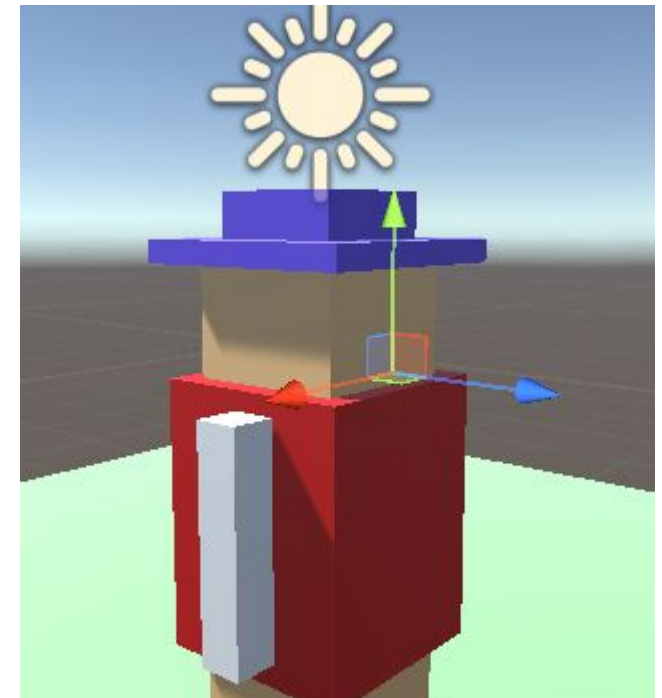
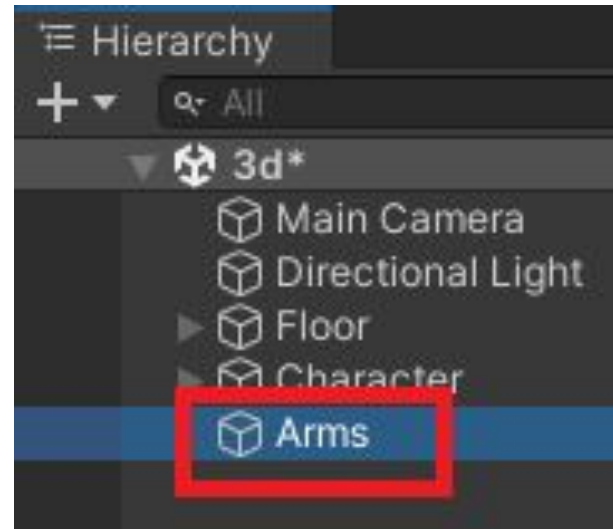
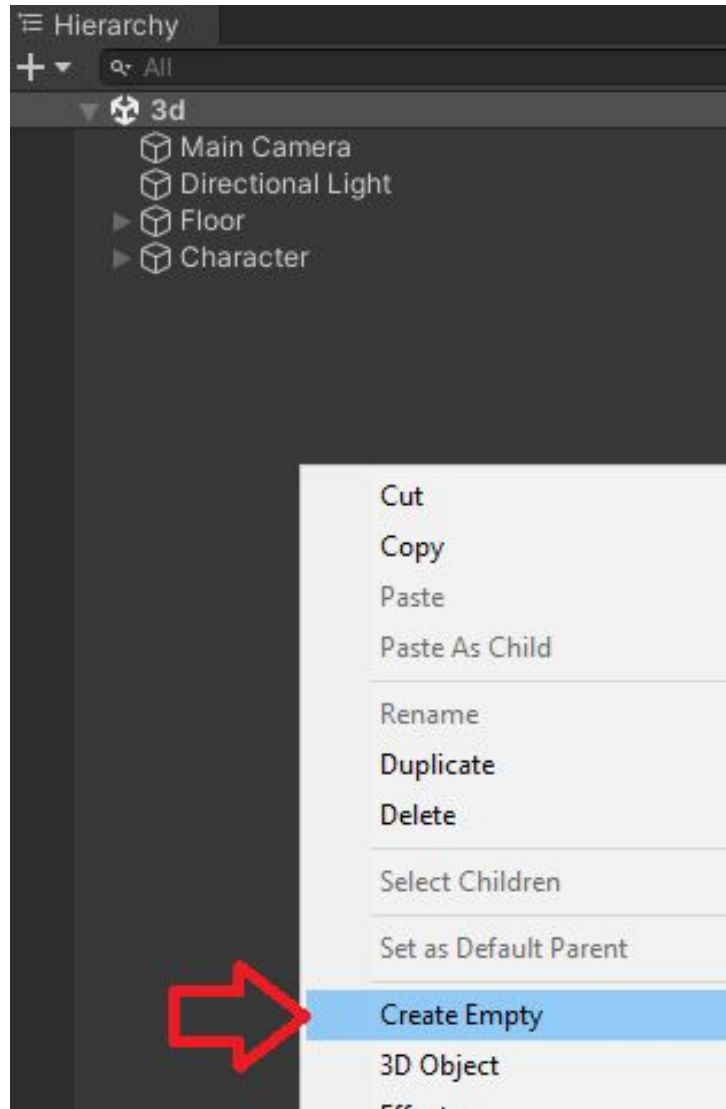
Урок

2

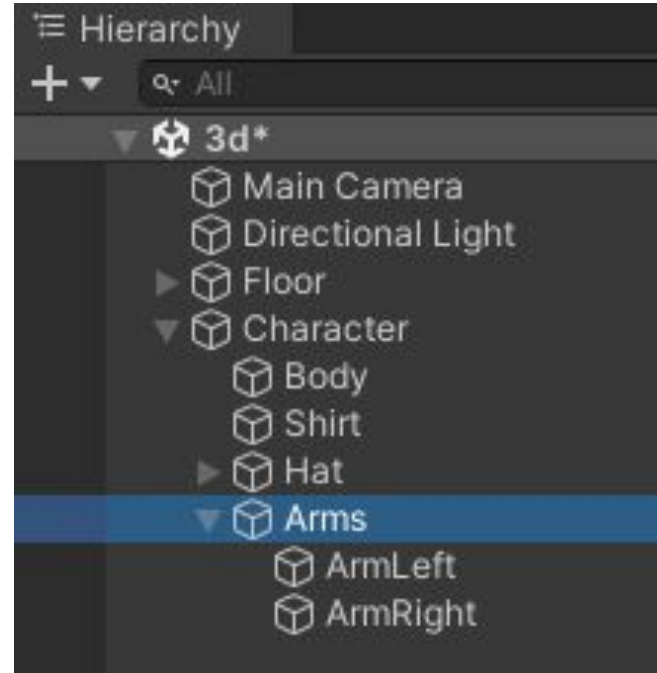
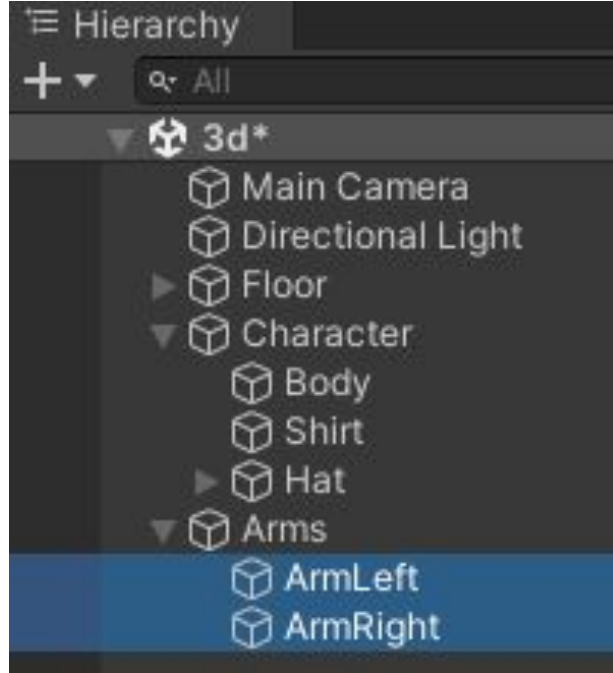
В Иерархии создаем Create Empty с помощью нажатия

**В Иерархии ПКМ
Переименовываем в**

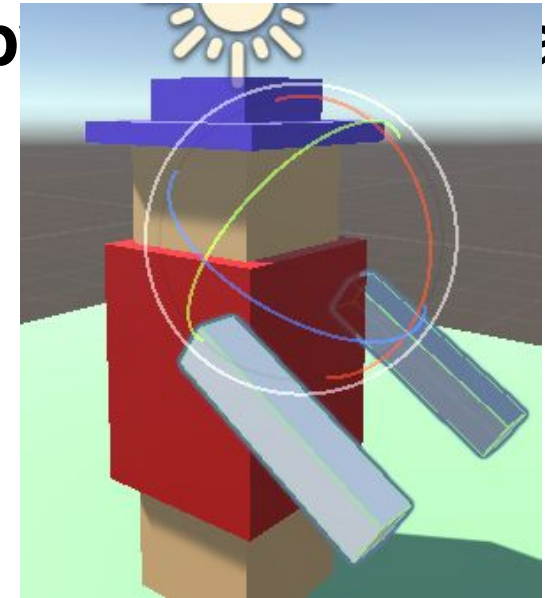
**Перемещаем Arms
как показано на
изображении**



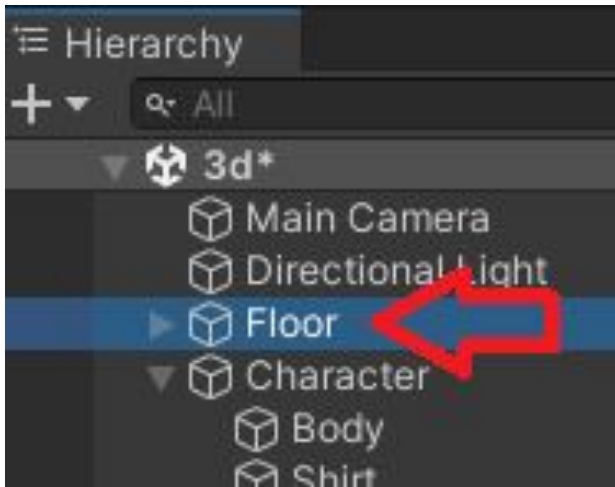
Перемещаем ArmLeft и Перемещаем Arms в Выберите инструмент ArmRight в Arms Character Rotate



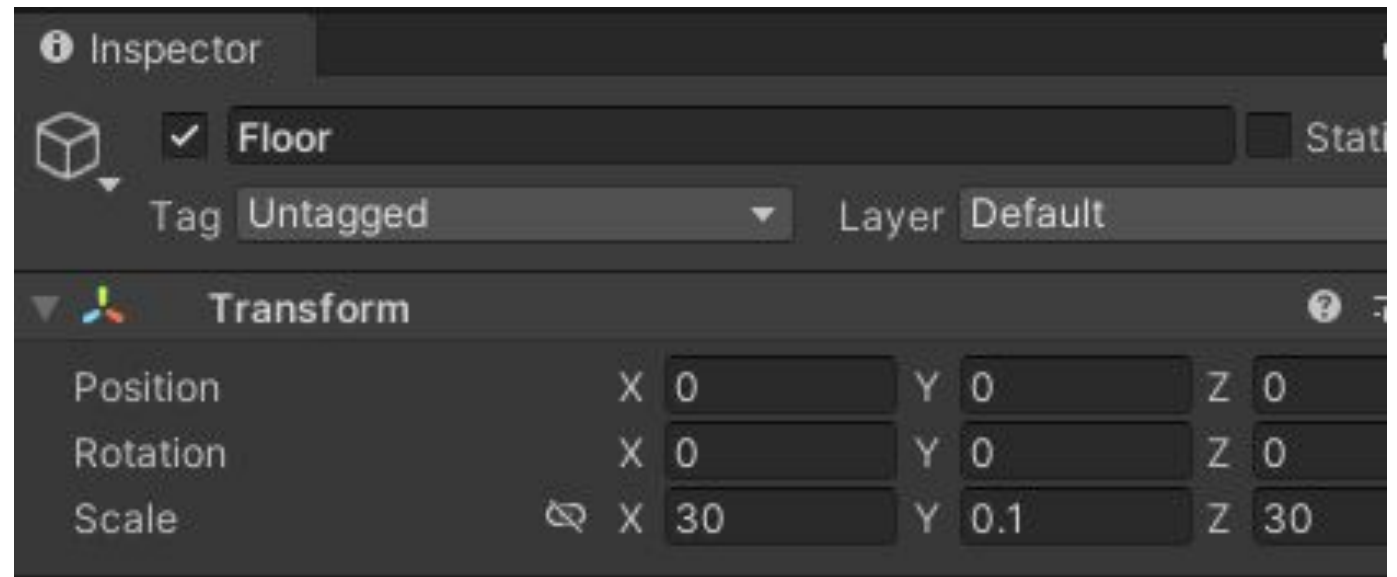
И попробуйте
повращать Arms
(должны двигаться
р а)



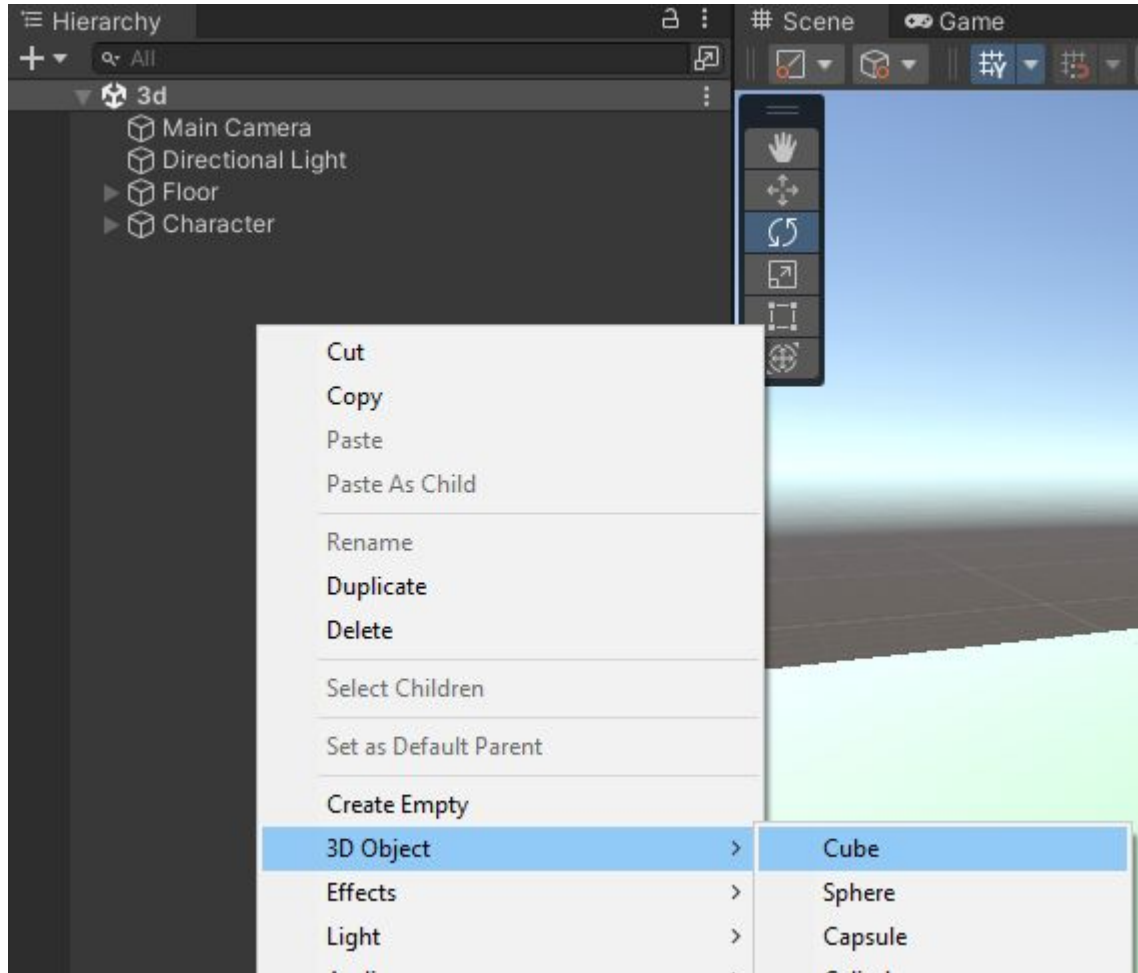
Увеличим площадку Floor



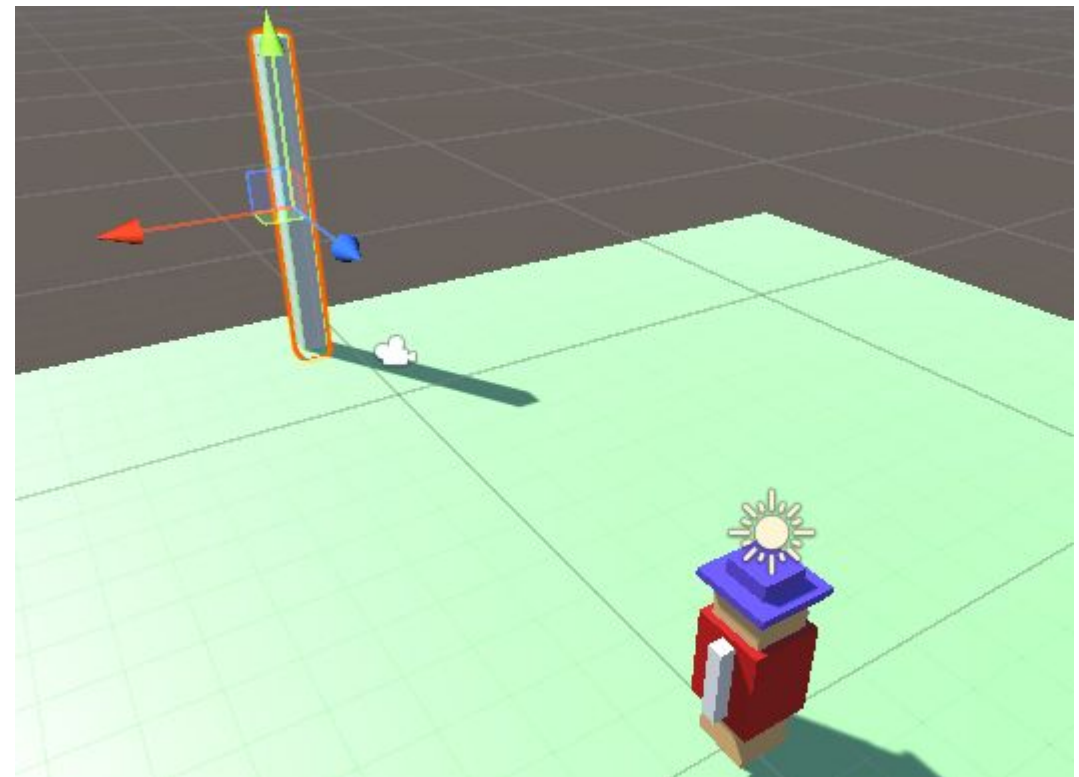
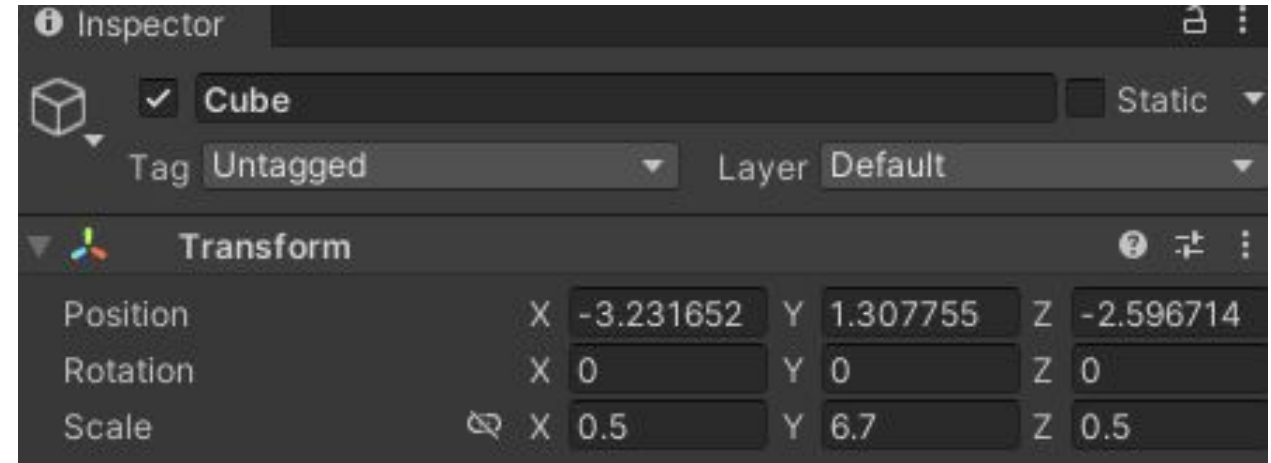
Scale: X = 30 Y = 0.1 Z = 30



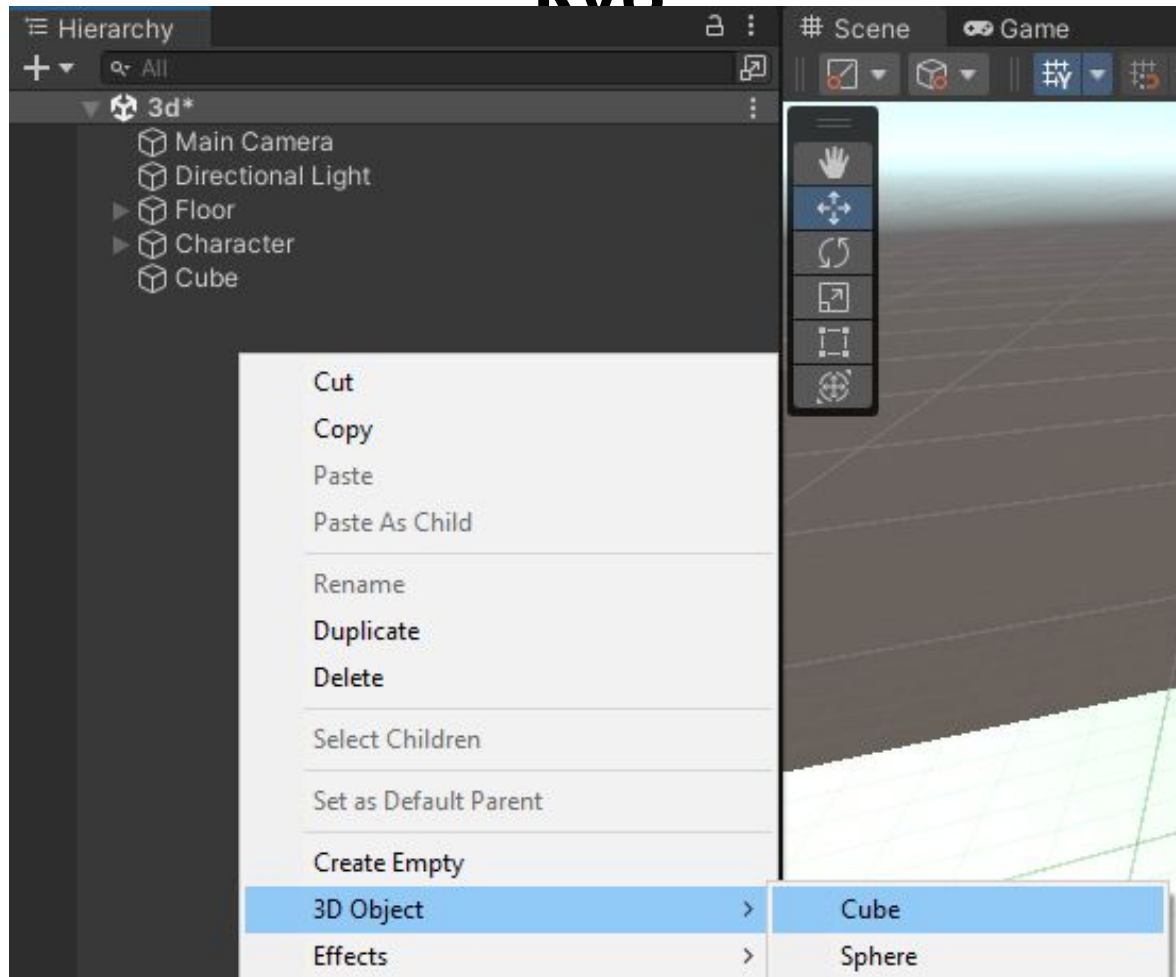
Создадим куб, зададим размер и переместим



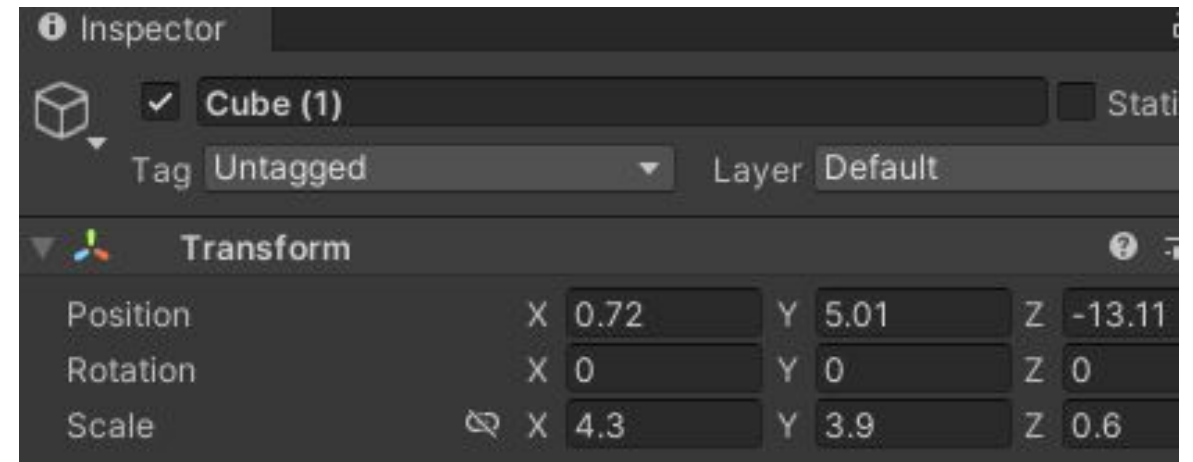
Scale: X = 0.5 Y = 6.7 Z = 0.5



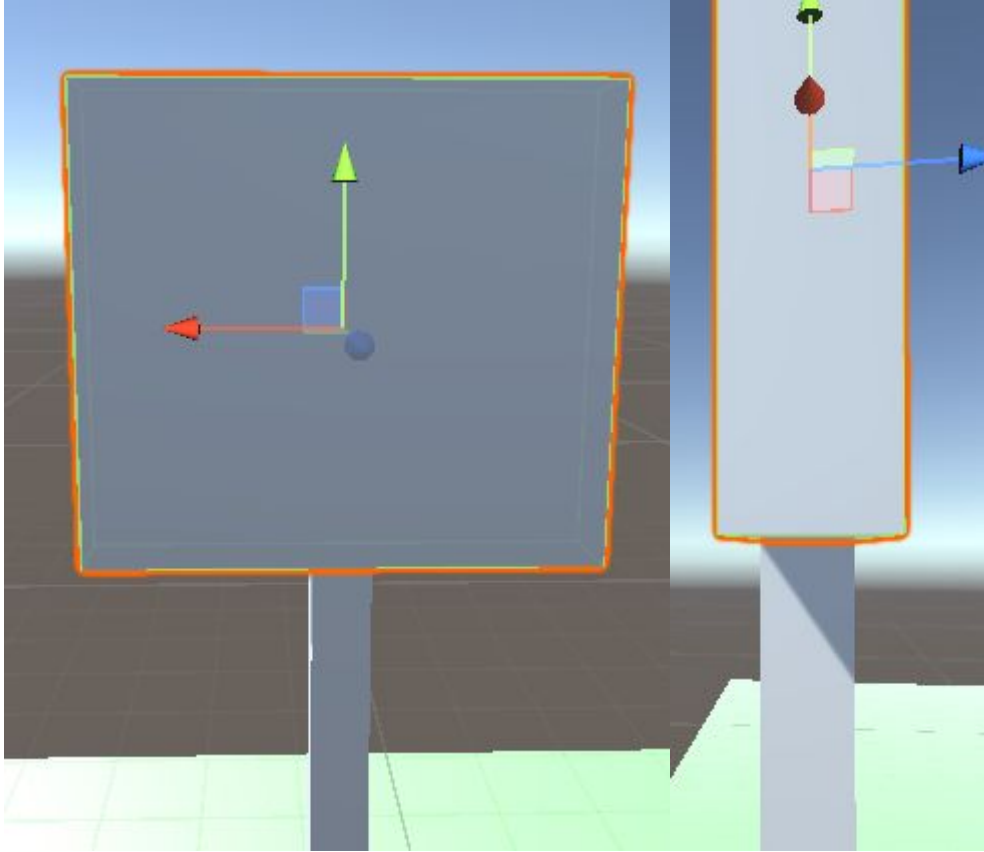
Создадим еще один Куб



Scale: X = 4.3 Y = 3.9 Z = 0.6



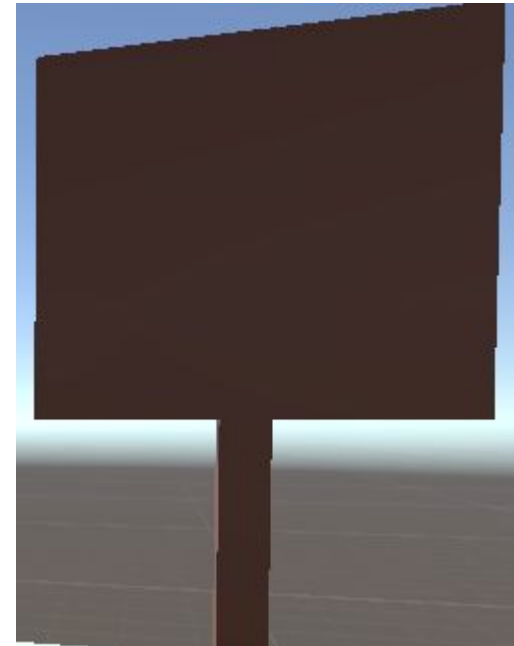
**Перемещаем как показано
на изображении (должно быть всё
симметрично)**



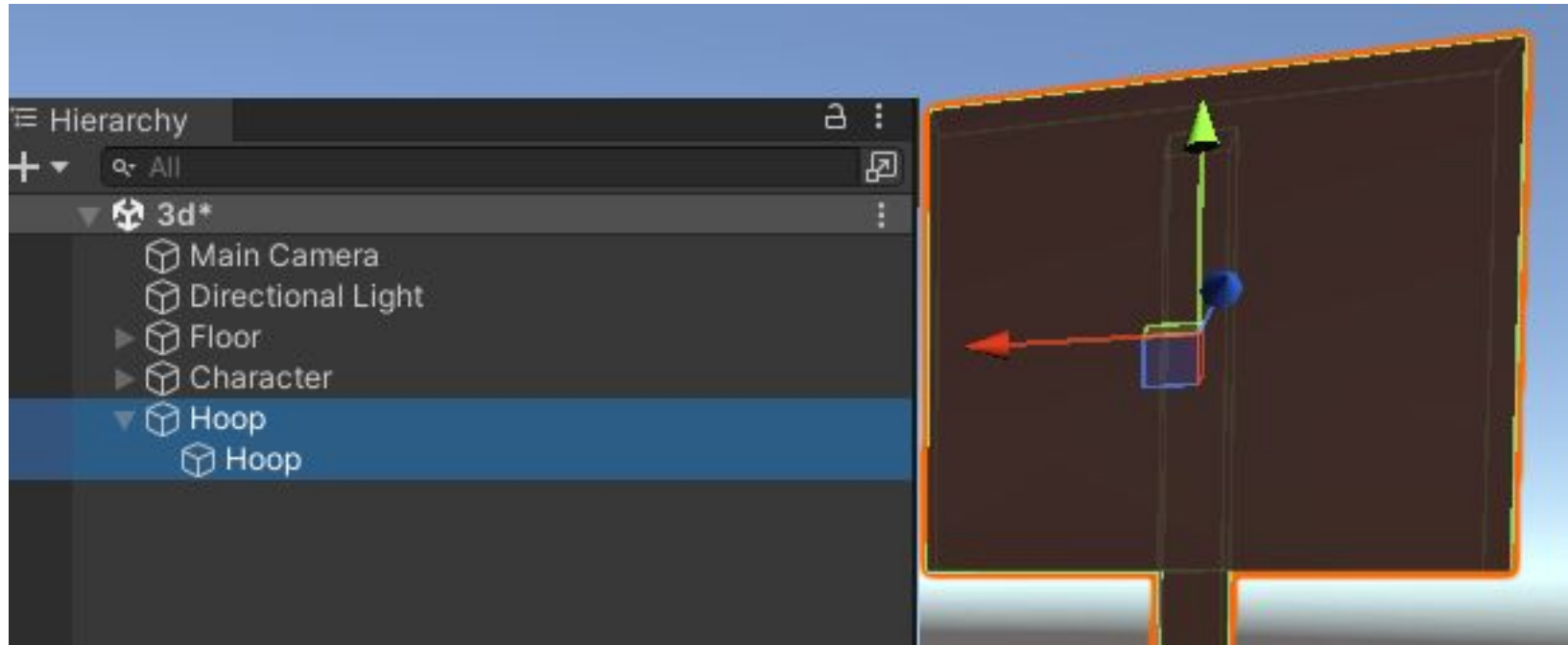
**Создаем материал Ноор
(Цвет ставим темно-коричневый)**



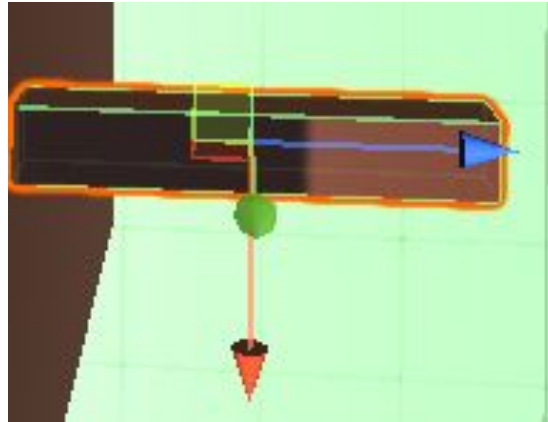
Применяем данный материал



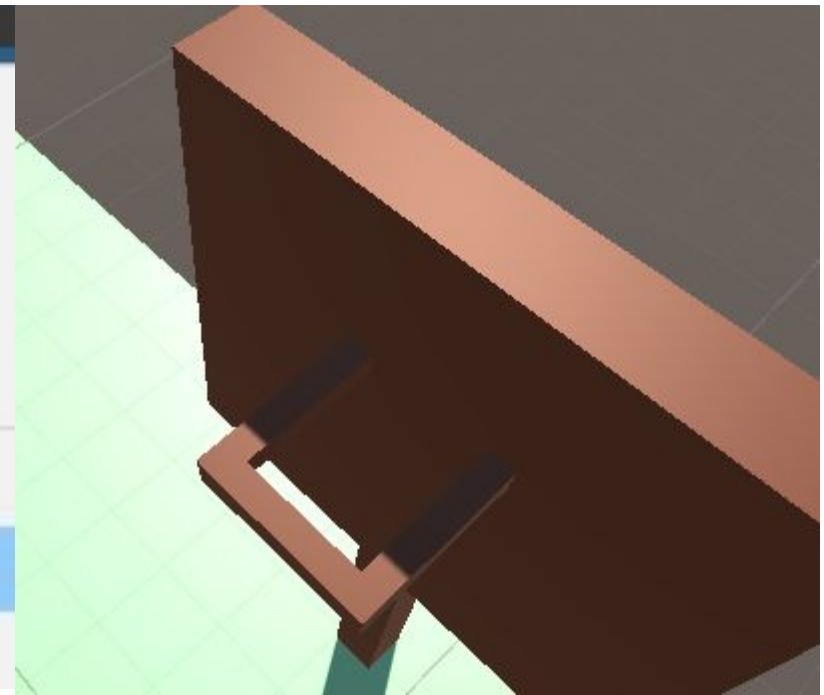
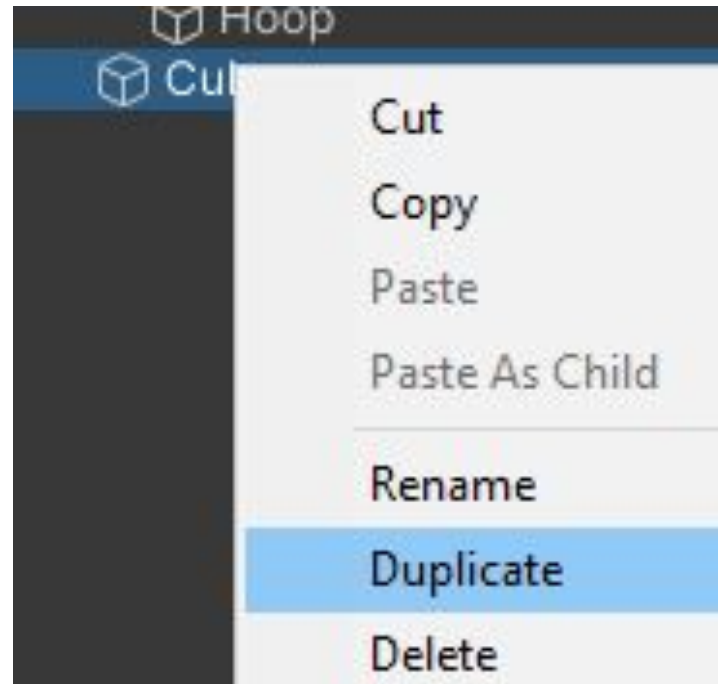
Переименовываем обе детали в Ноор
В Иерархии одну деталь переместим в
другую



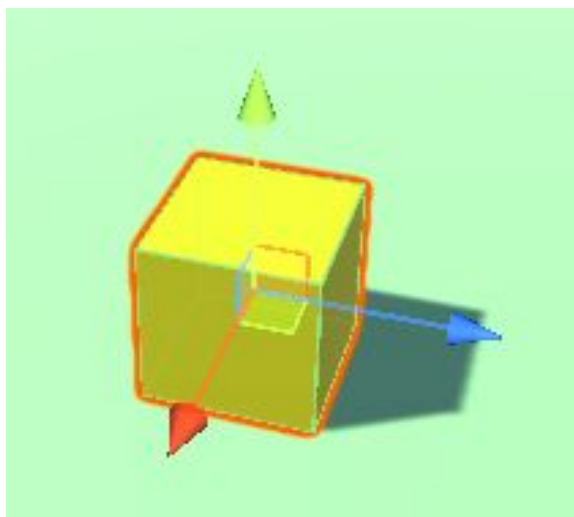
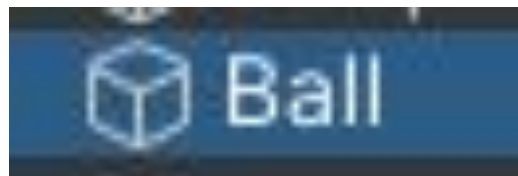
Создадим новый Куб (материал применяем



Дублируем
2 раза
и создаем
квадратное
кольцо для
мяча



Создадим новый Куб, назовем его Ball и создадим материал Ball и зададим любой цвет нашему мячу



Размер

Scale: X = 0.45 Y = 0.45 Z = 0.45

Scale



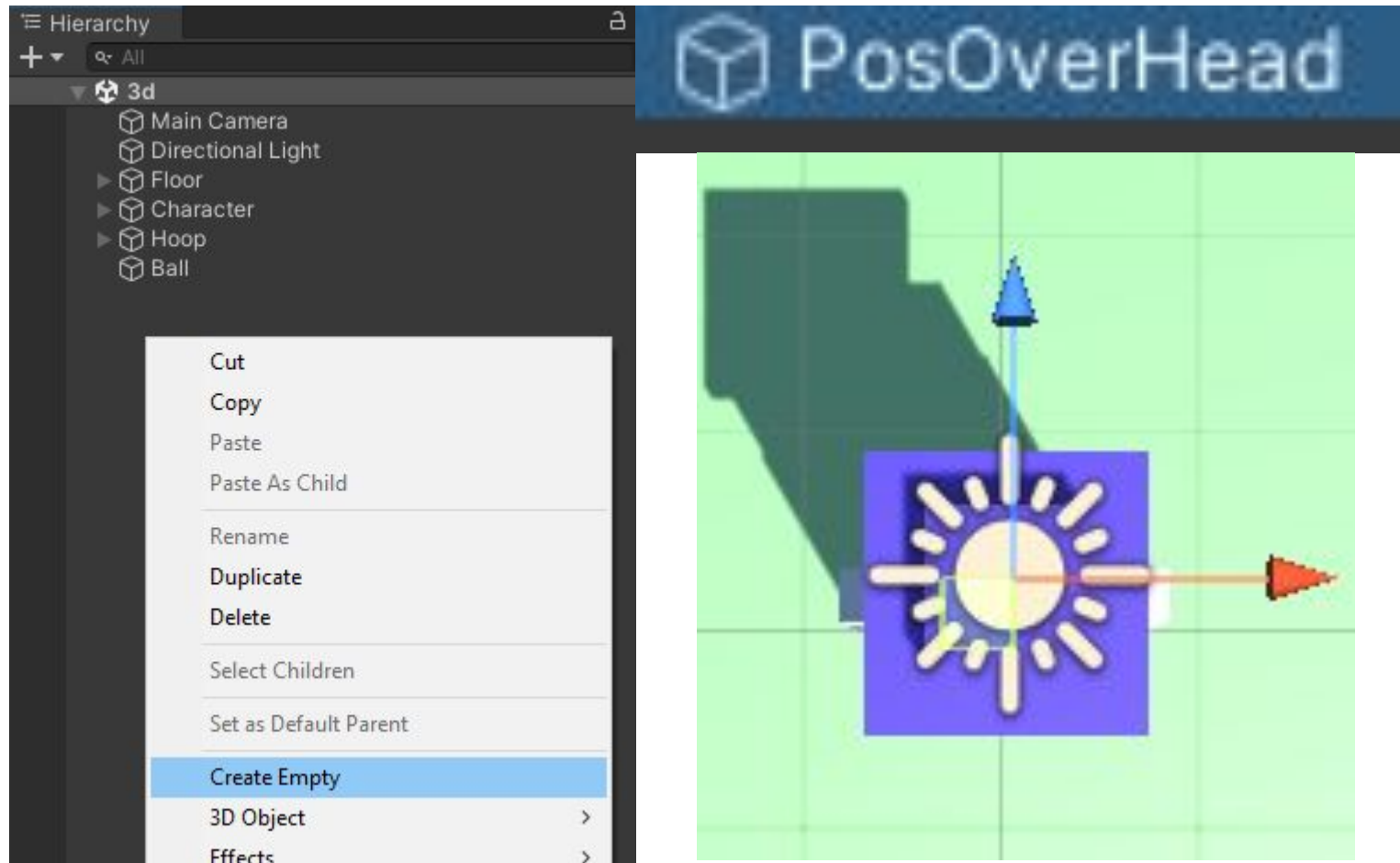
X 0.45

Y 0.45

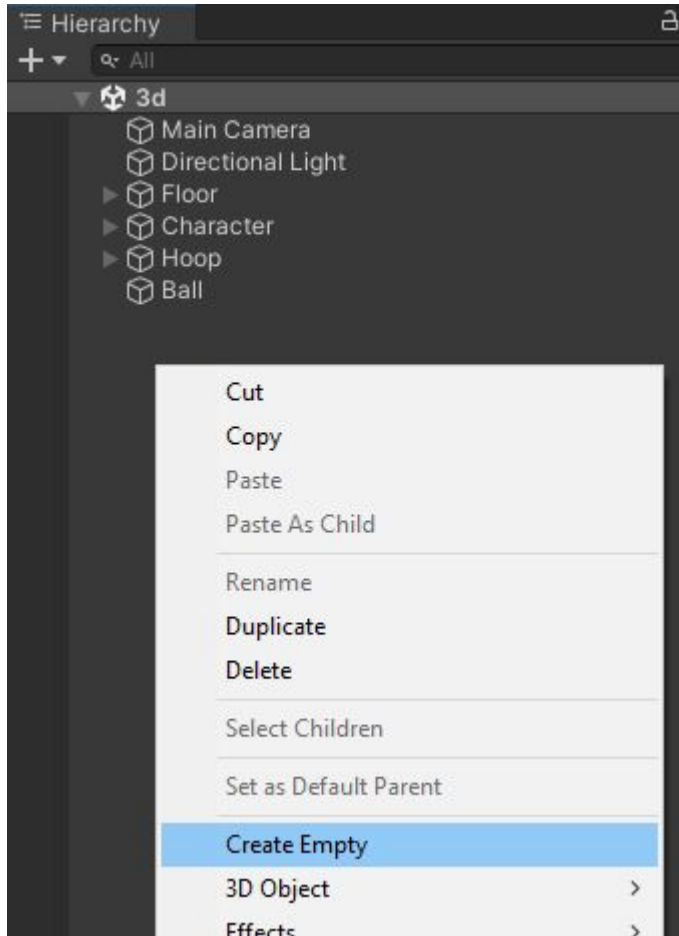
Z 0.45

**В Иерархии создаем Create Empty с помощью
нажатия**

**В Иерархии ПКМ и назовем PosOverHead
Переместим его на голову персонажа**



Создаем еще Create Empty и назовем **PosDribble**
Переместим его в центр мяча

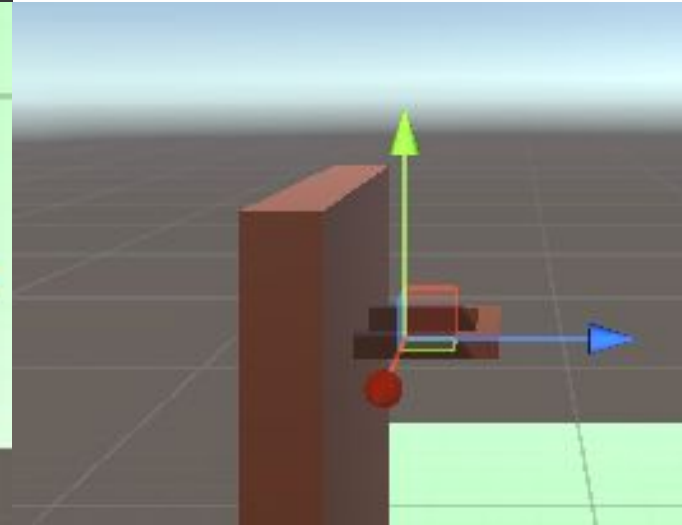
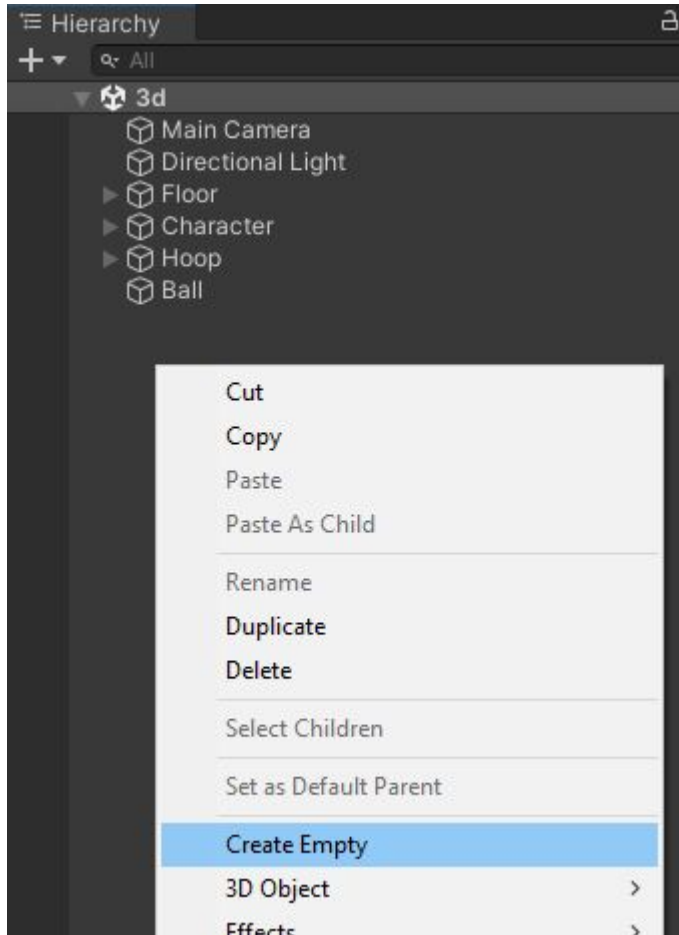


Перемещаем
обе наши
позиции в Arms



Создаем Create Empty и назовем **Target**

Переместим его в центр баскетбольного кольца

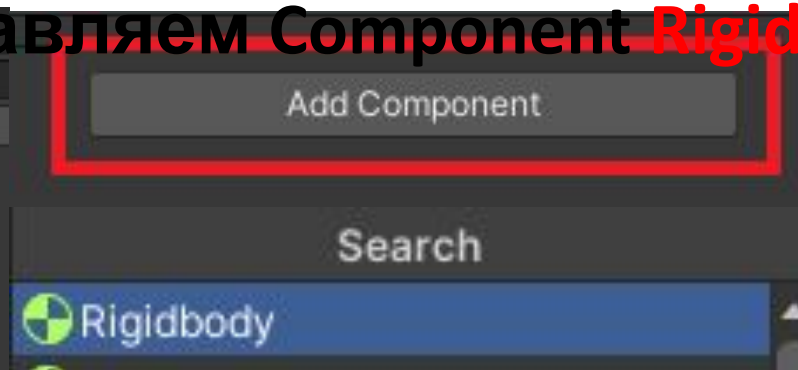
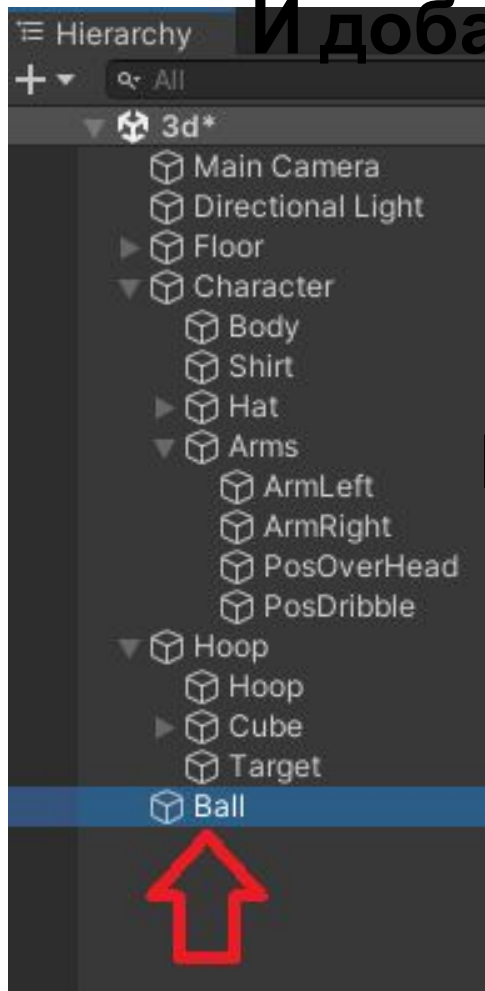


**Перемещаем
Target в Hoop**

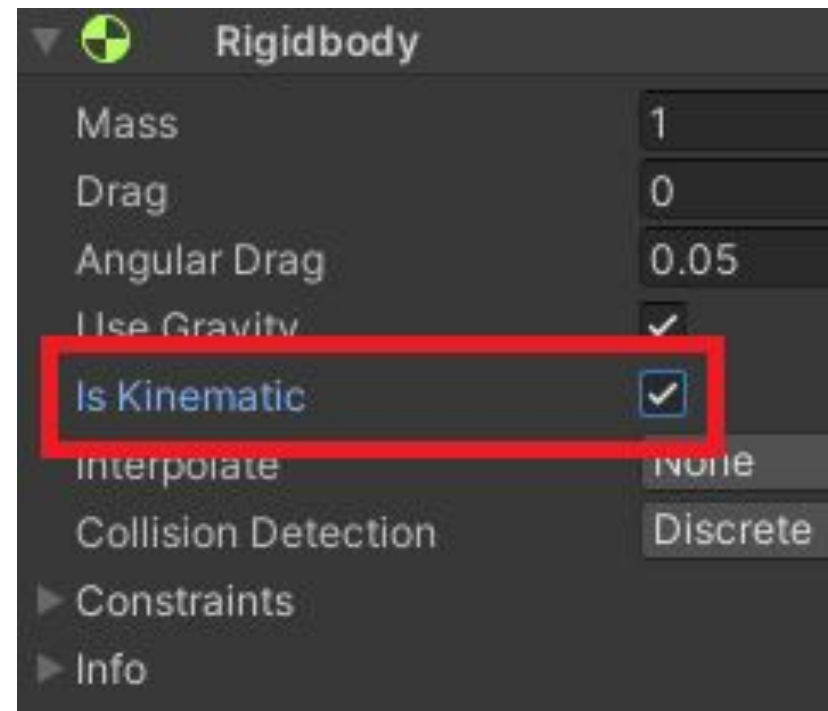


В Иерархии нажимаем на **Ball**, переходим в Инспектор

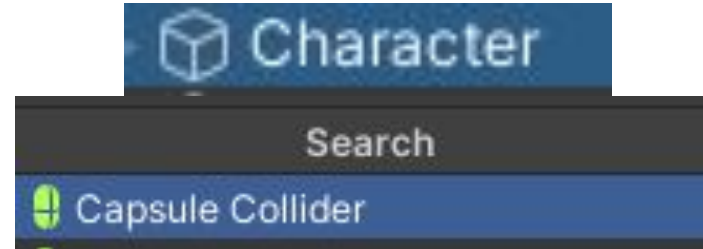
И добавляем Component **Rigidbody**



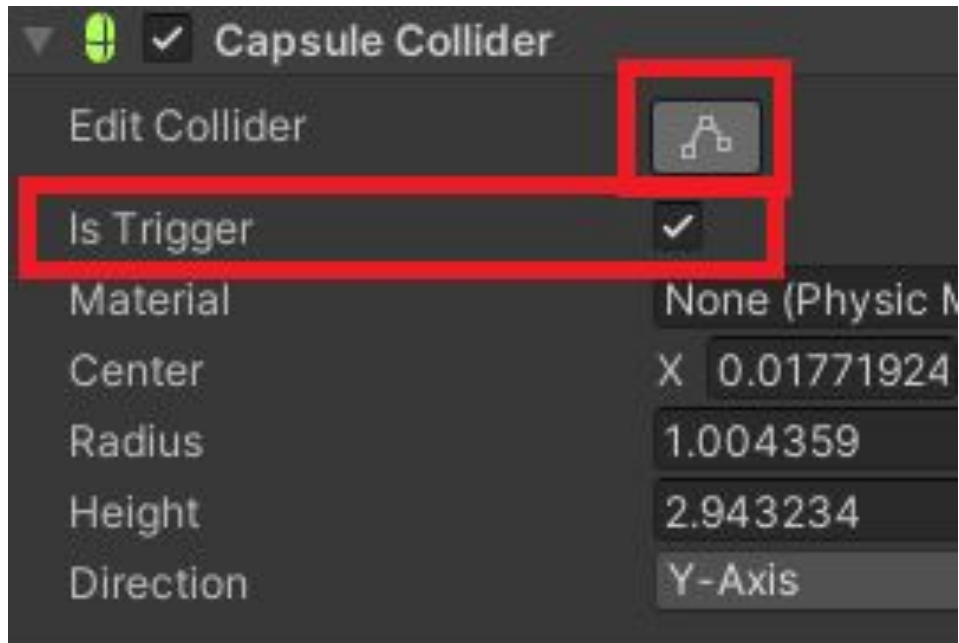
В Rigidbody устанавливаем галочку Is Kinematic



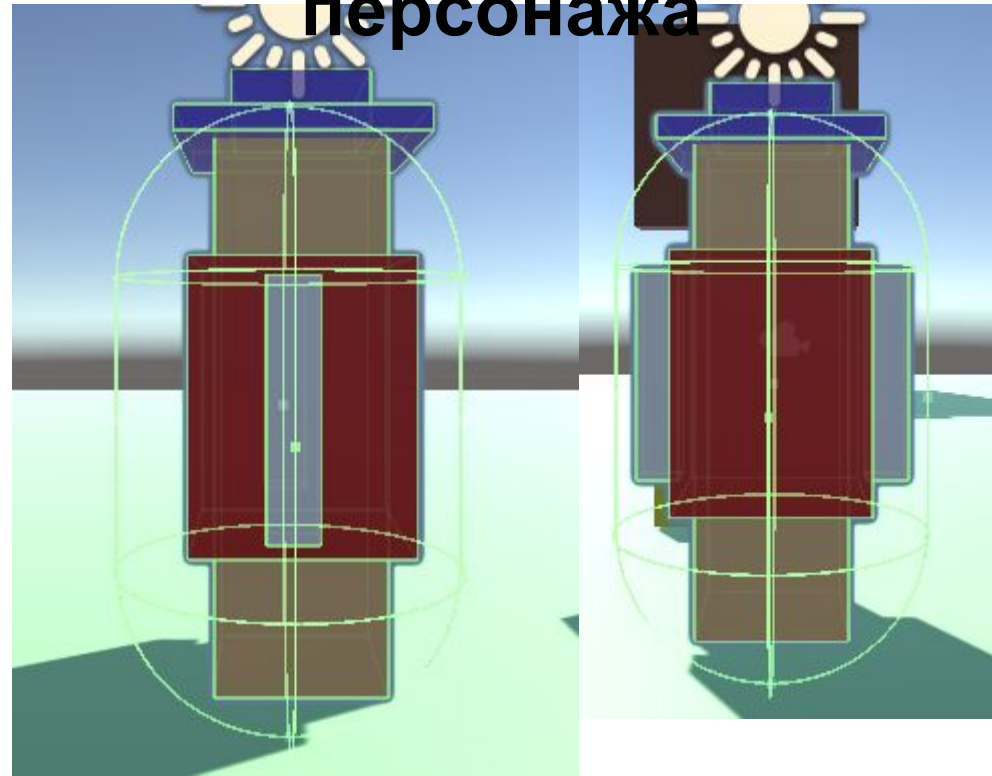
В Иерархии нажимаем на **Character**, переходим в Инспектор и добавляем Component **Capsule Collider**



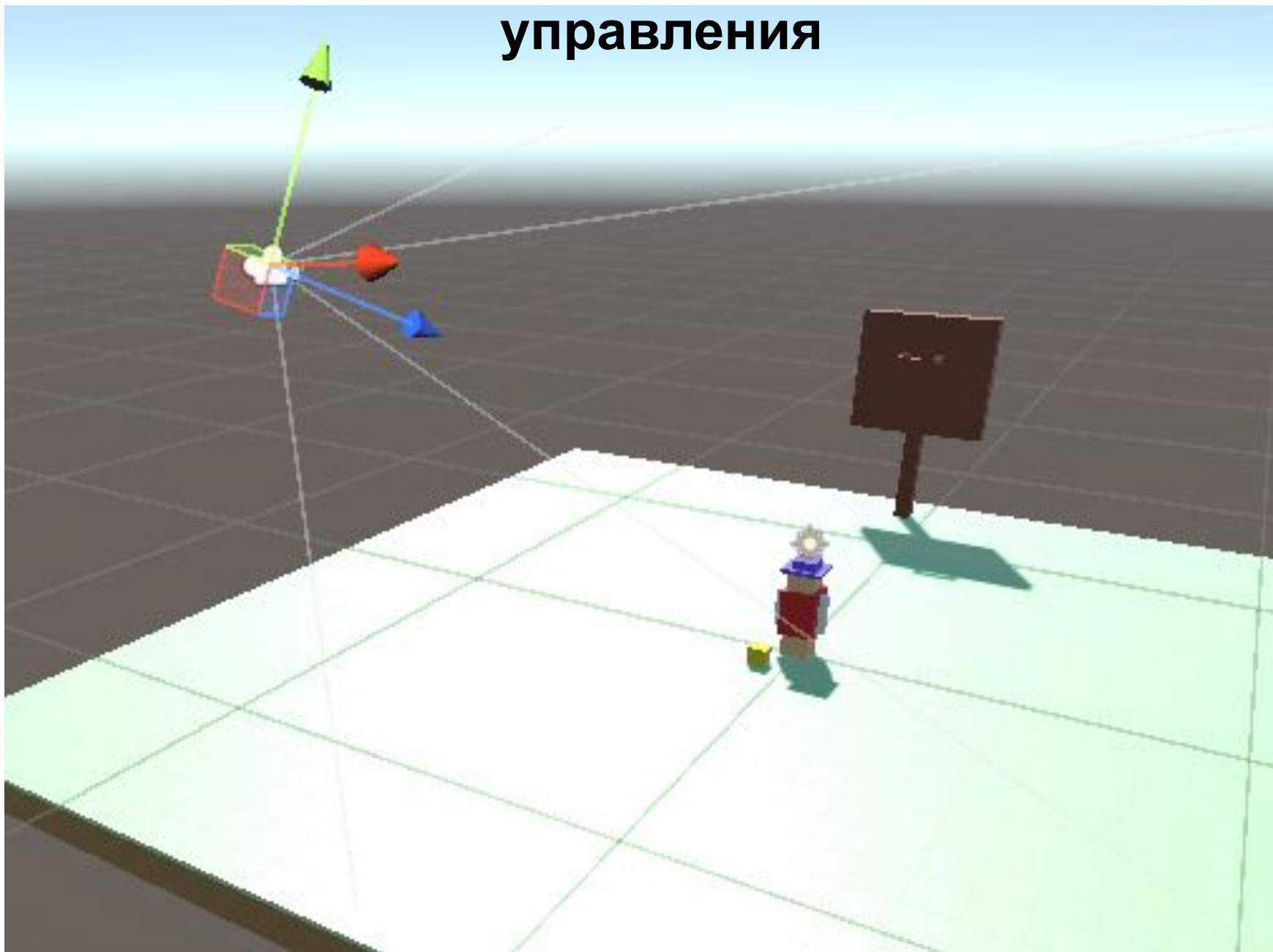
В **Capsule Collider** ставим
Галочку **Is Trigger**
нажимаем **Edit Collider**



Увеличиваем границы
персонажа

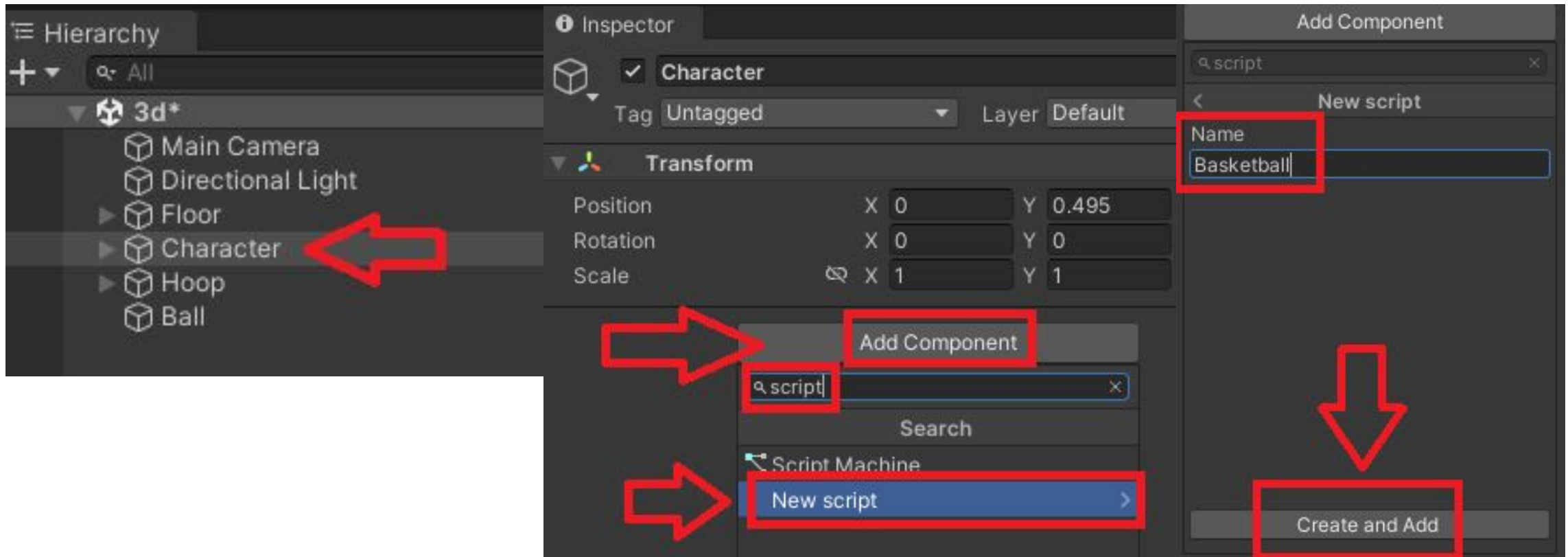


Перемещаем камеру над картой для удобства управления

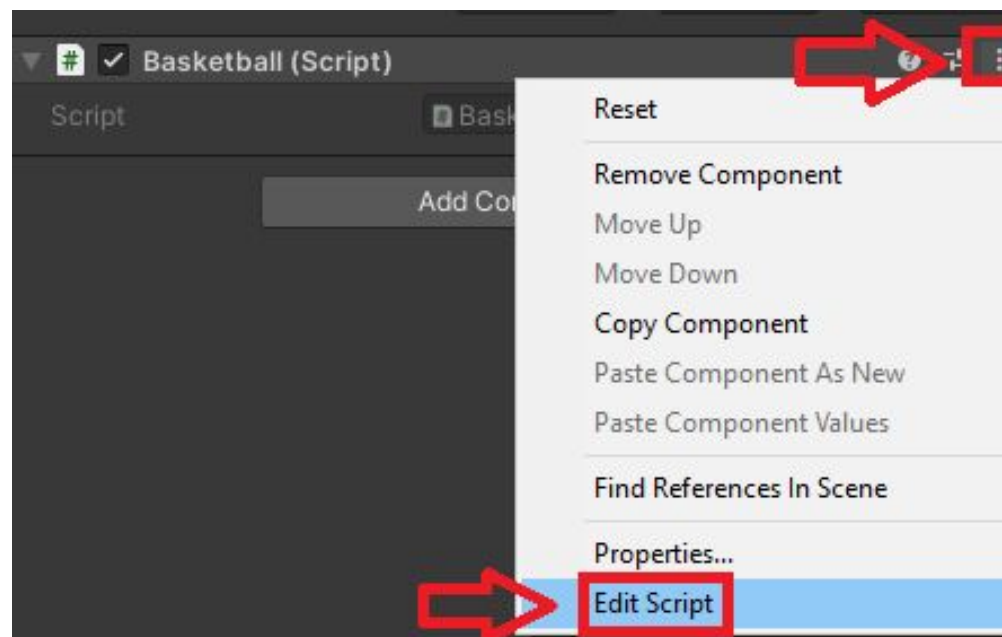


Создаем скрипт нашему Баскетболисту

В Иерархии выбираем **Character**. В Инспекторе нажимаем **add Component** (в строке поиска пишем script). Нажимаем **New script**, задаем имя **Basketball** и нажимаем **Create and Add**



Открываем Скрипт (Edit Script)



Переходим к написанию скрипта

**Первым делом создадим переменные с именами
нам необходимых объектов и переменную
скорости.**

```
public class Basketball : MonoBehaviour  
{
```

```
    [SerializeField] float _speed = 10;  
    [SerializeField] Transform _ball;  
    [SerializeField] Transform _arms;  
    [SerializeField] Transform _posOverhead;  
    [SerializeField] Transform _posDribble;  
    [SerializeField] Transform _target;
```

Создадим булеву переменную true, когда мяч будет в руках, false, когда мяч в полете и время полета.

```
[SerializeField] Transform _posOverhead;  
[SerializeField] Transform _posDribble;  
[SerializeField] Transform _target;
```

```
private bool _isBallInHands = true;  
private bool _isBallFlying = false;  
private float _t = 0;
```

```
void Update()  
{
```


Создаем новый вектор, который будет отвечать за перемещение нашего персонажа по горизонтали и вертикали.

Так же меняем позиции нашего персонажа, умножением вектора на скорость и время, и задаем строчку трансформации положения.

```
private float _t = 0;
```

```
void Update()  
{
```

```
    Vector3 direction = new Vector3(Input.GetAxisRaw("Horizontal"), 0, Input.GetAxisRaw("Vertical"));  
    transform.position += direction * _speed * Time.deltaTime;  
    transform.LookAt(transform.position + direction);
```

Добавляем кнопку пробела:

1) При нажатии на нее мяч оказывается над головой персонажа

2) При отпускании кнопки мяч окажется в кольце.

Добавим строки изменения позиции мяча и угла рук

```
transform.LookAt(transform.position + direction);
```

```
if (_isBallInHands)
{
    if (Input.GetKey(KeyCode.Space))
    {
        _ball.position = _posOverhead.position;
        _arms.localEulerAngles = Vector3.right * 180;

        transform.LookAt(_target.position);
    }
}
```


Определяем для мяча перемещается персонаж или нет

```
{  
    _ball.position = _posOverhead.position;  
    _arms.localEulerAngles = Vector3.right * 180;  
  
    transform.LookAt( target.position);  
}  
else  
{  
    _ball.position = _posDribble.position + Vector3.up * Mathf.Abs(Mathf.Sin(Time.time * 5));  
    _arms.localEulerAngles = Vector3.right * 0;  
}
```

Следующий скрипт будем определять время полета и длительность, вектор полета к кольцу и используется

```
        _arms.localEulerAngles = Vector3.  
    }
```

```
    if (Input.GetKeyUp(KeyCode.Space))  
    {  
        _isBallInHands = false;  
        _isBallFlying = true;  
        _t = 0;  
    }  
}  
  
if (_isBallFlying)  
{  
    _t += Time.deltaTime;  
    float duration = 0.5f;  
    float t01 = _t / duration;
```

```
float t01 = _t / duration;
```

```
Vector3 a = _posOverhead.position;
```

```
Vector3 b = _target.position;
```

```
Vector3 pos = Vector3.Lerp(a, b, t01);
```

```
Vector3 arc = Vector3.up * 5 * Mathf.Sin(t01 * 3.14f);
```

```
_ball.position = pos + arc;
```

```
if (t01 >= 1)
```

```
{
```

```
    _isBallFlying = false;
```

```
    _ball.GetComponent<Rigidbody>().isKinematic = false;
```

```
}
```

```
}
```

```
}
```

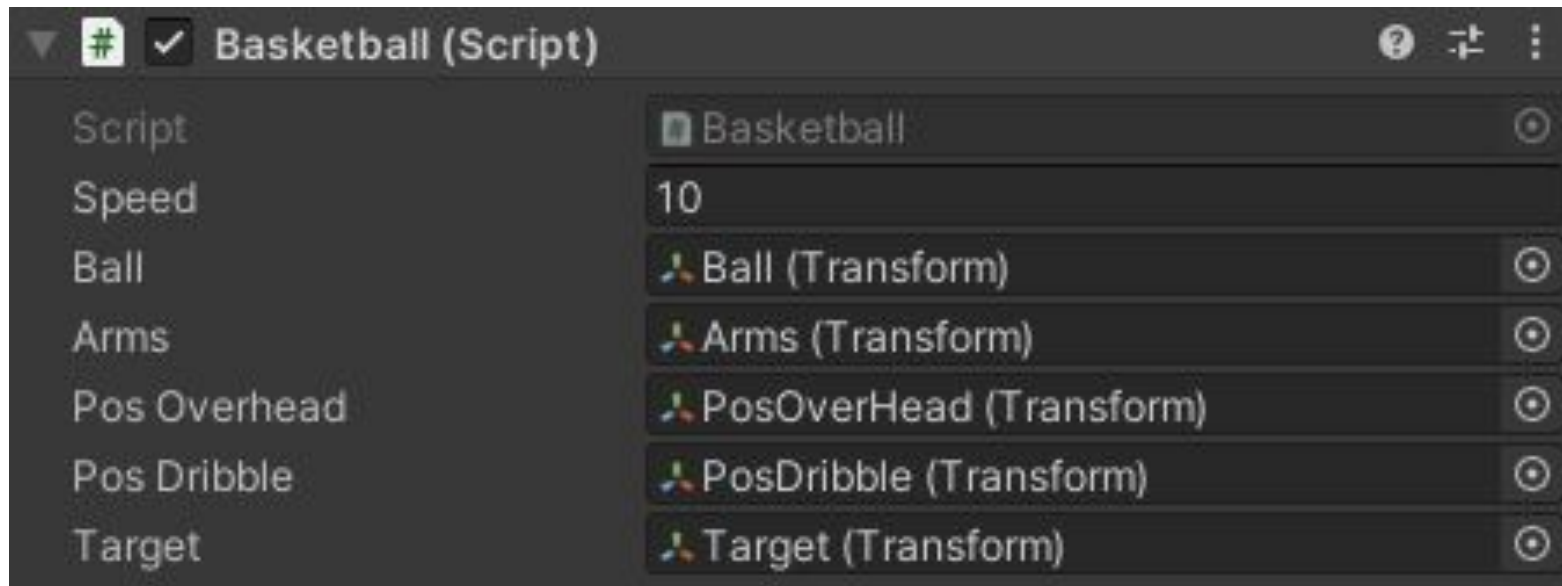
Добавим кинематику, которая будет определять находится мяч у персонажа или нет.

```

        _ball.GetComponent<Rigidbody>().isKinematic = false;
    }
}

private void OnTriggerEnter(Collider other)
{
    if (!_isBallInHands && !_isBallFlying)
    {
        _isBallInHands = true;
        _ball.GetComponent<Rigidbody>().isKinematic = true;
    }
}
}
```

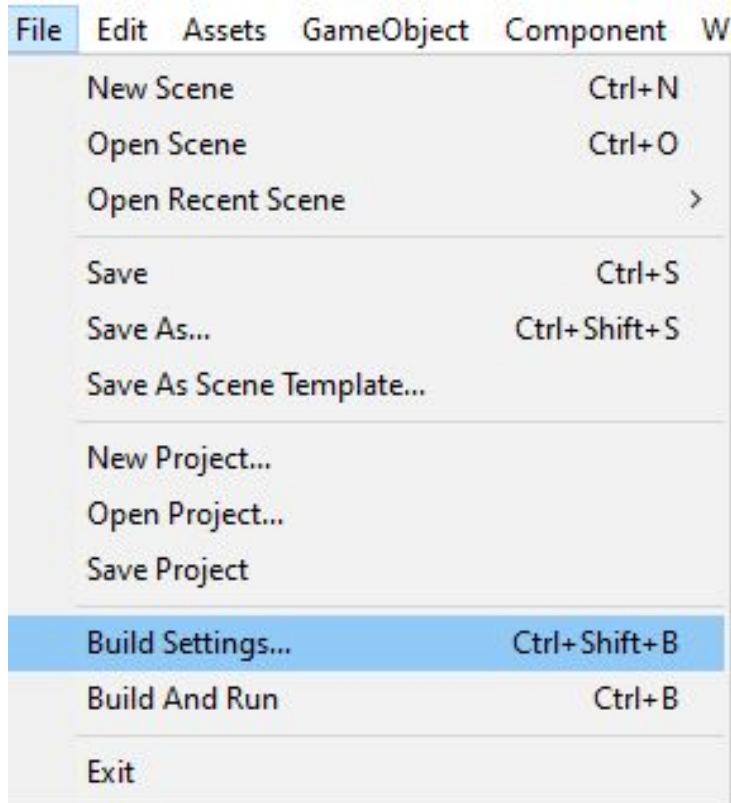
Устанавливаем соответствующие объекты переменным созданным в скрипте



Запускаем и проверяем игру

Экспорт игры на ПК

Нажимаем File – Build Settings



Указываем путь сохранения
и готово

Добавляем сцену Add Open
Scenes и нажимаем Build

