

演習 1 : UNITY スクリプトの基礎

(01) 04/16

1 A | Unityとエディタの連携

(02) 04/23

1 B | Transform・キーイベント・マウスイベント

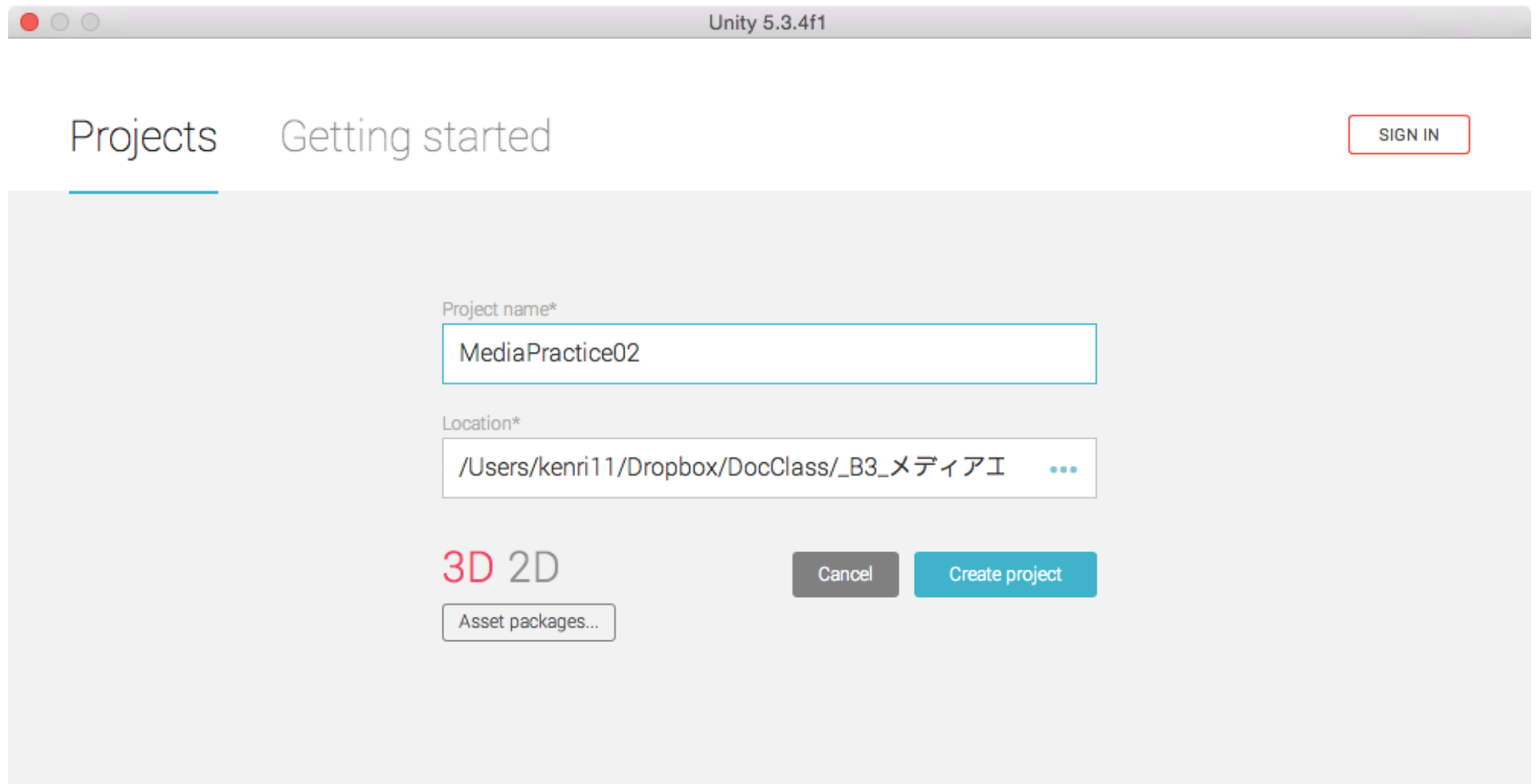
(03) 04/30

1 C | 剛体特性・カメラの視点

(04) 05/07

1 D | プレハブ (gameobjectの雛形) , タグ, その他

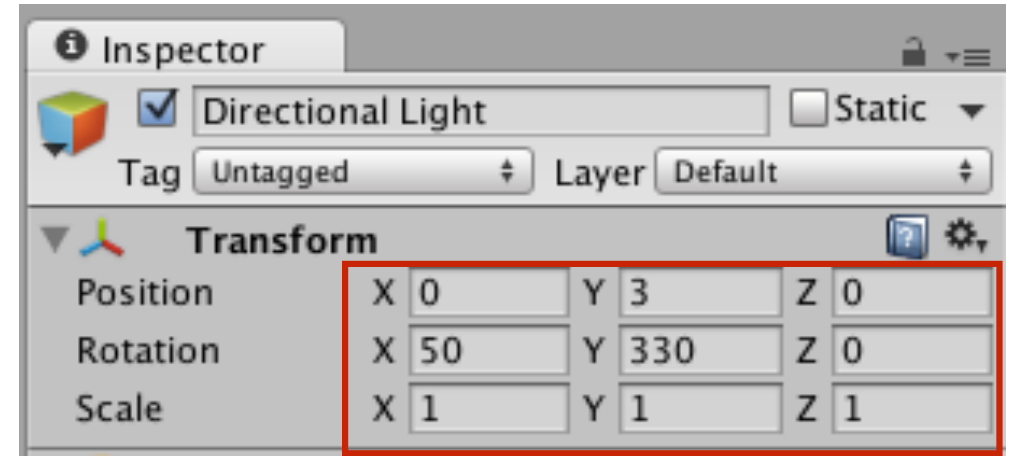
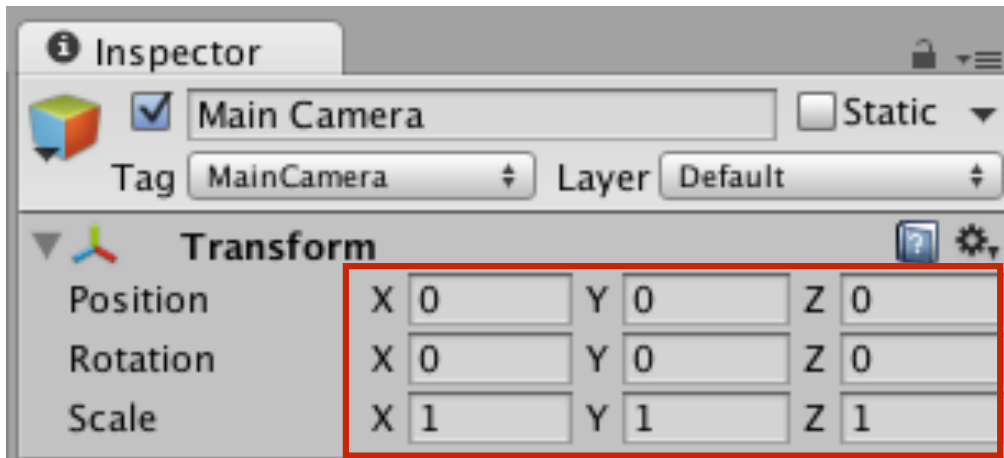
MediaPractice02

Transform ・ キーイベント ・ マウスイベント

1

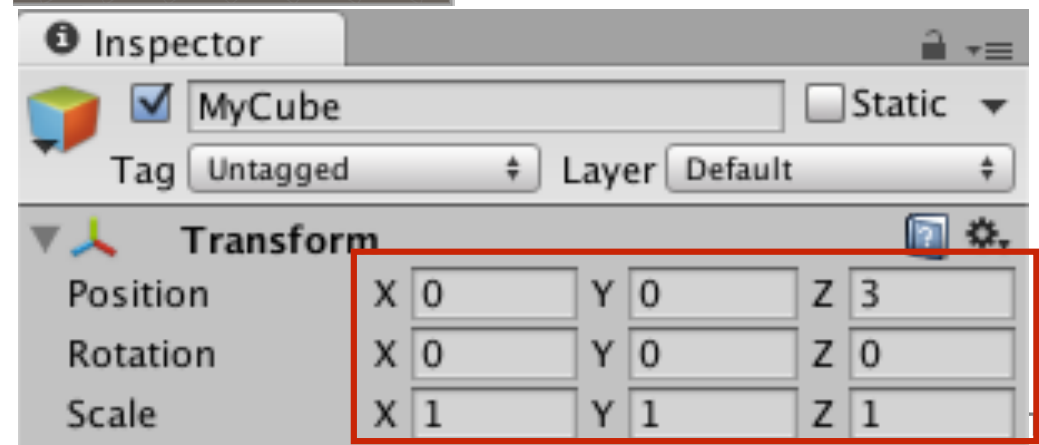
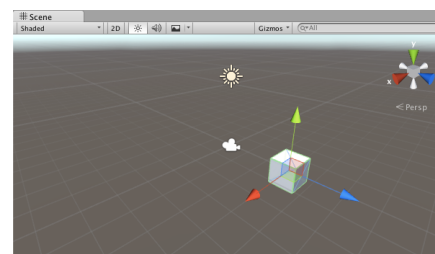
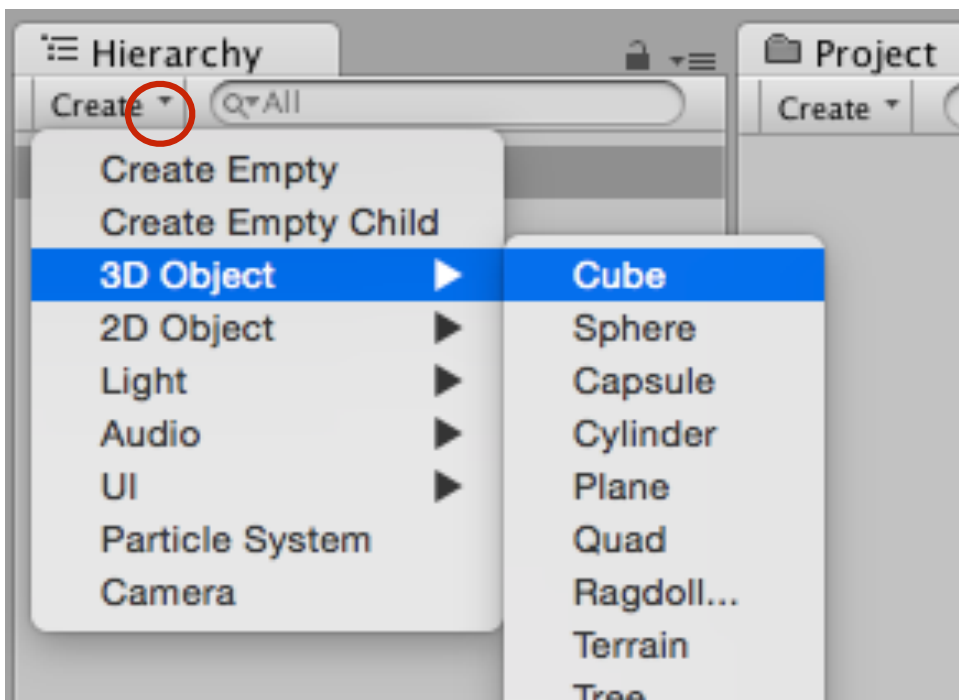
カメラと光のtransform（空間変数）を以下のように設定します。

準備 1



2

ヒエラルキービューから, Cubeを追加して, 名前をMyCubeと変更した後に, transformを以下のように設定します。

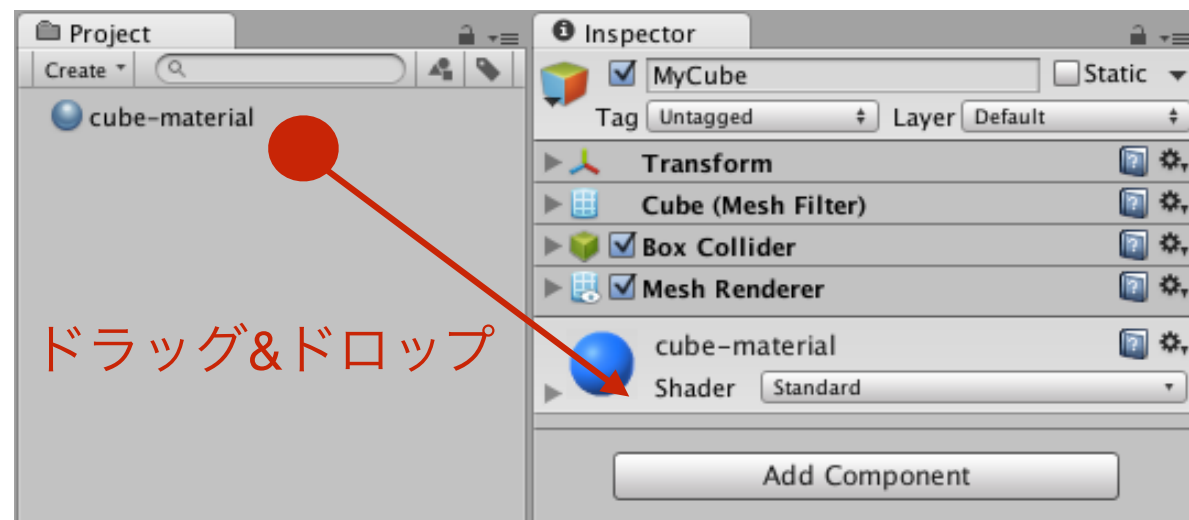
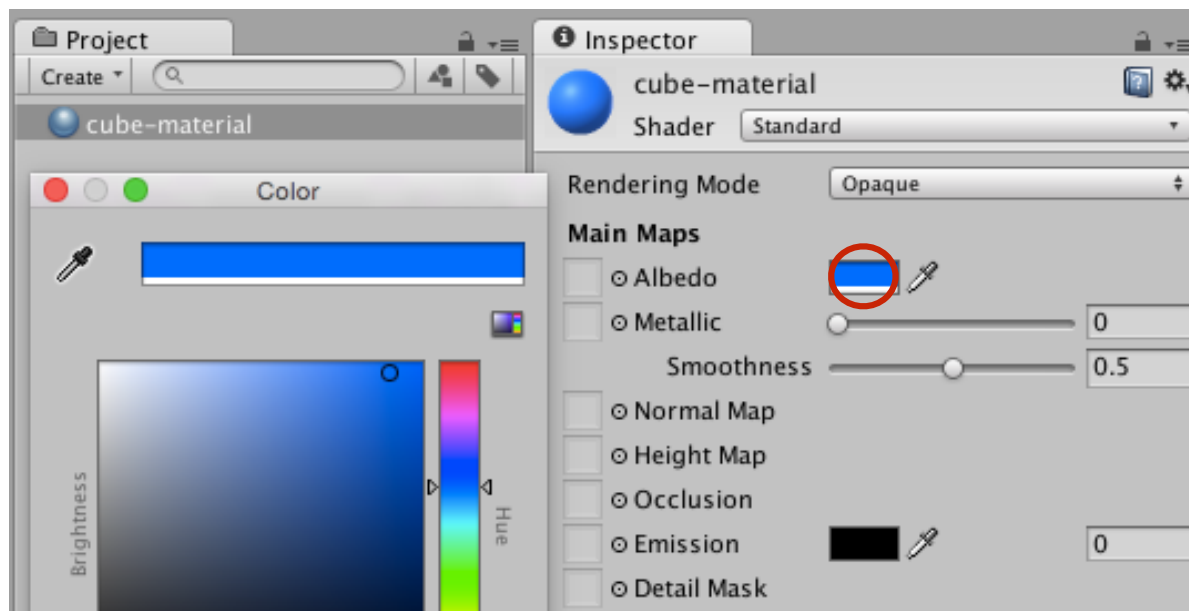
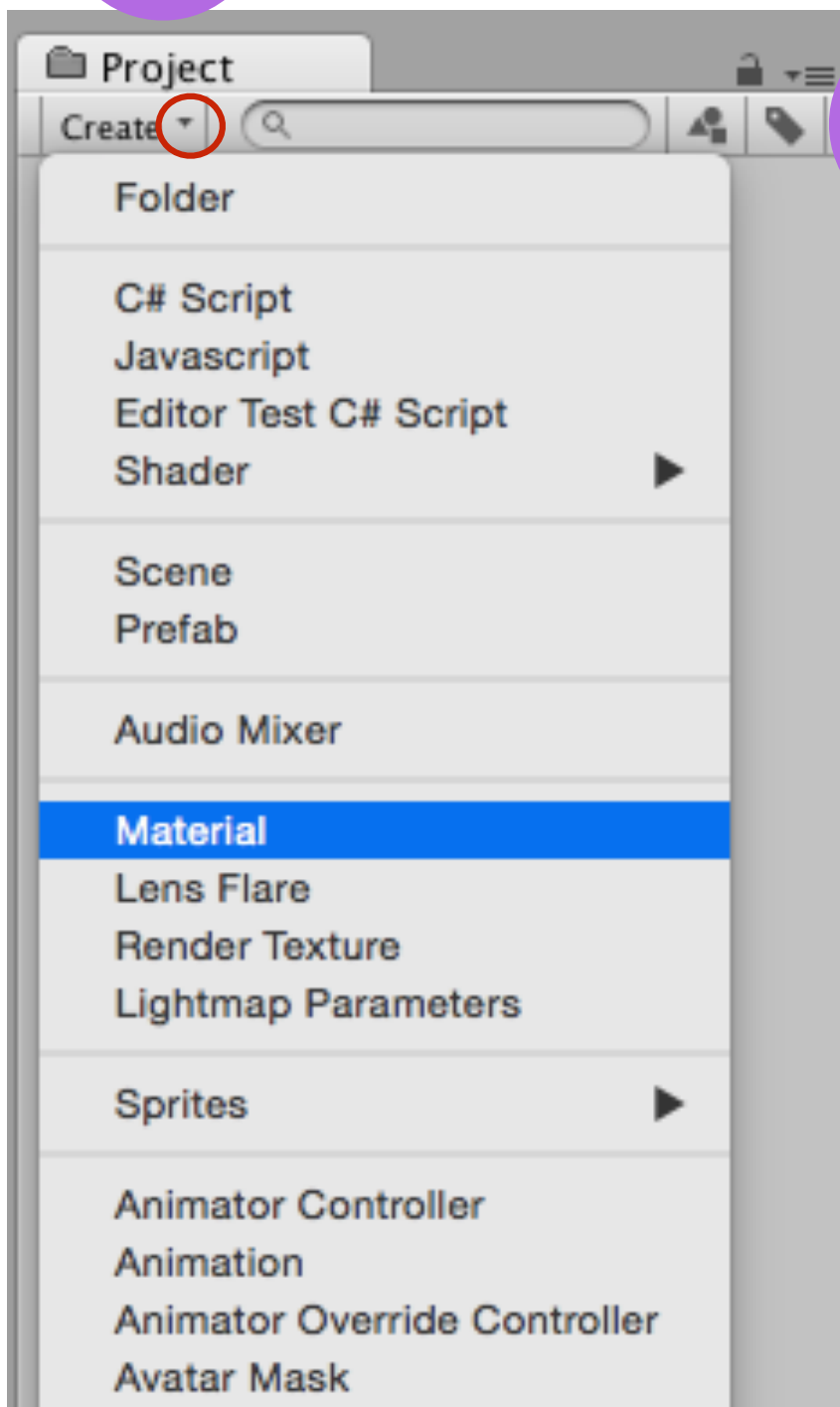


3

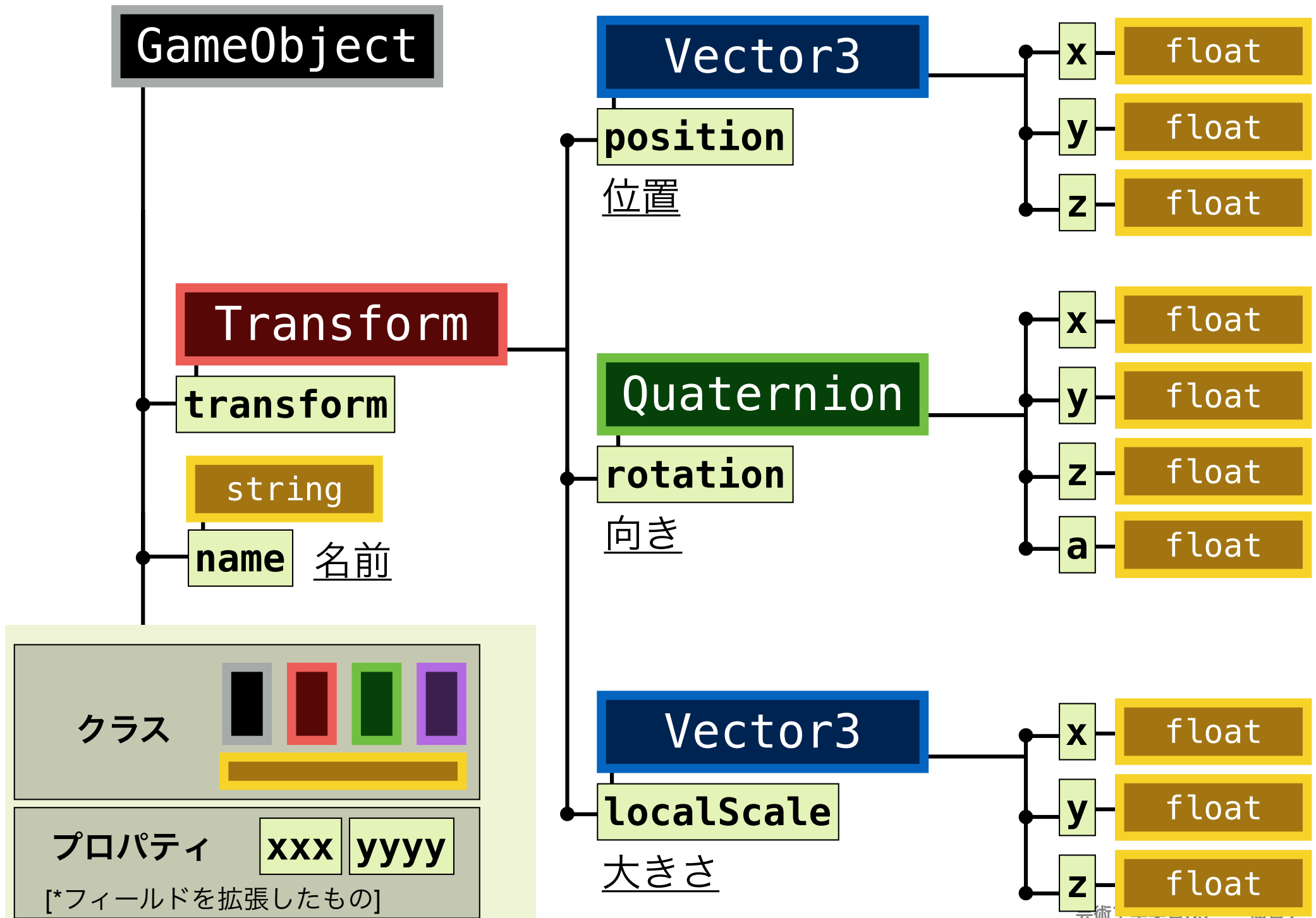
ProjectにMaterial（材質に関するオブジェクト）を追加して、「cube-material」という名前にしてください。

4

cube-materialのインスペクタで適当な色に設定し、MyCubeのインスペクタにドラッグ&ドロップします。

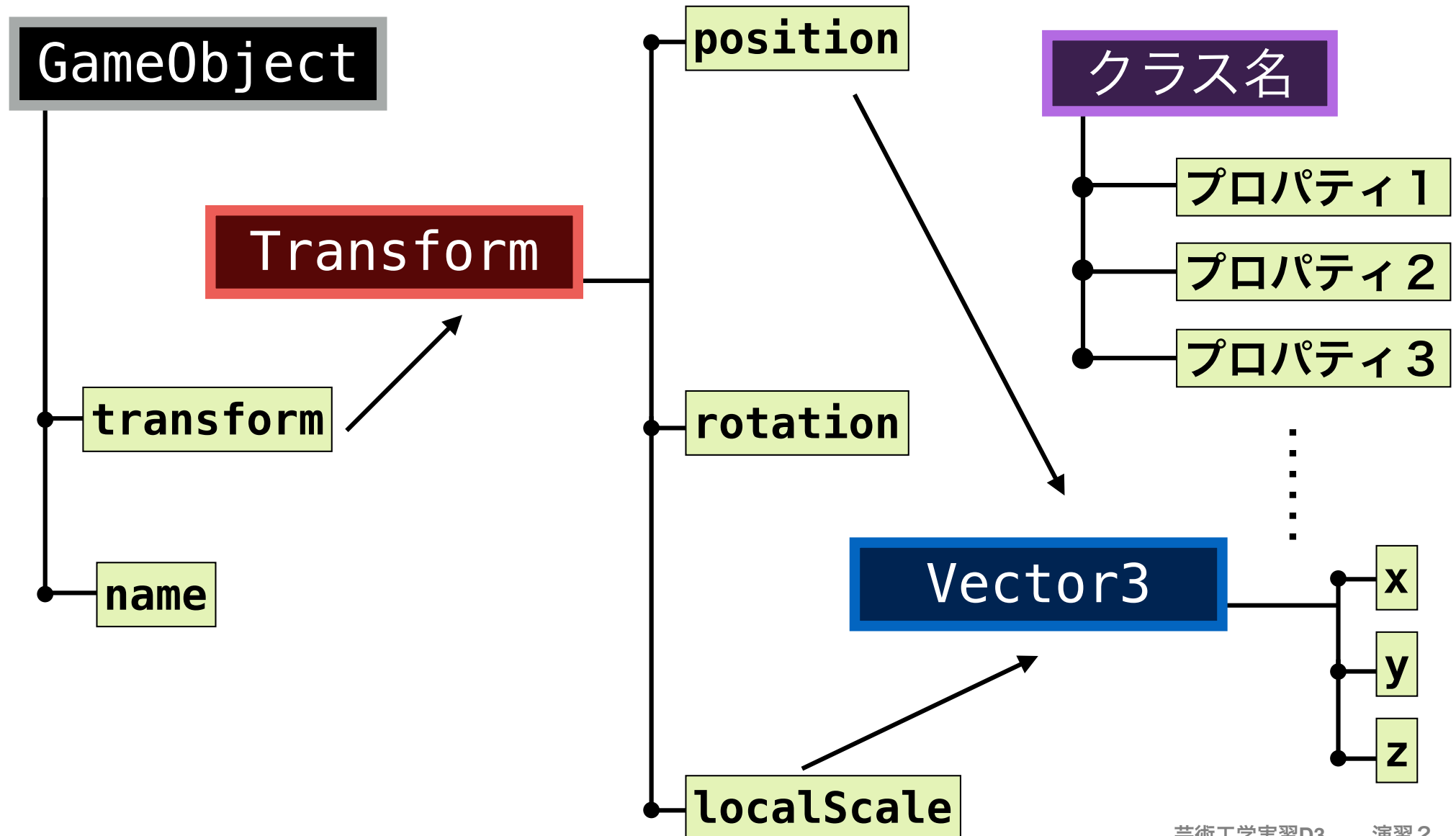


Transformの構造 (クラスとプロパティ*の関係)

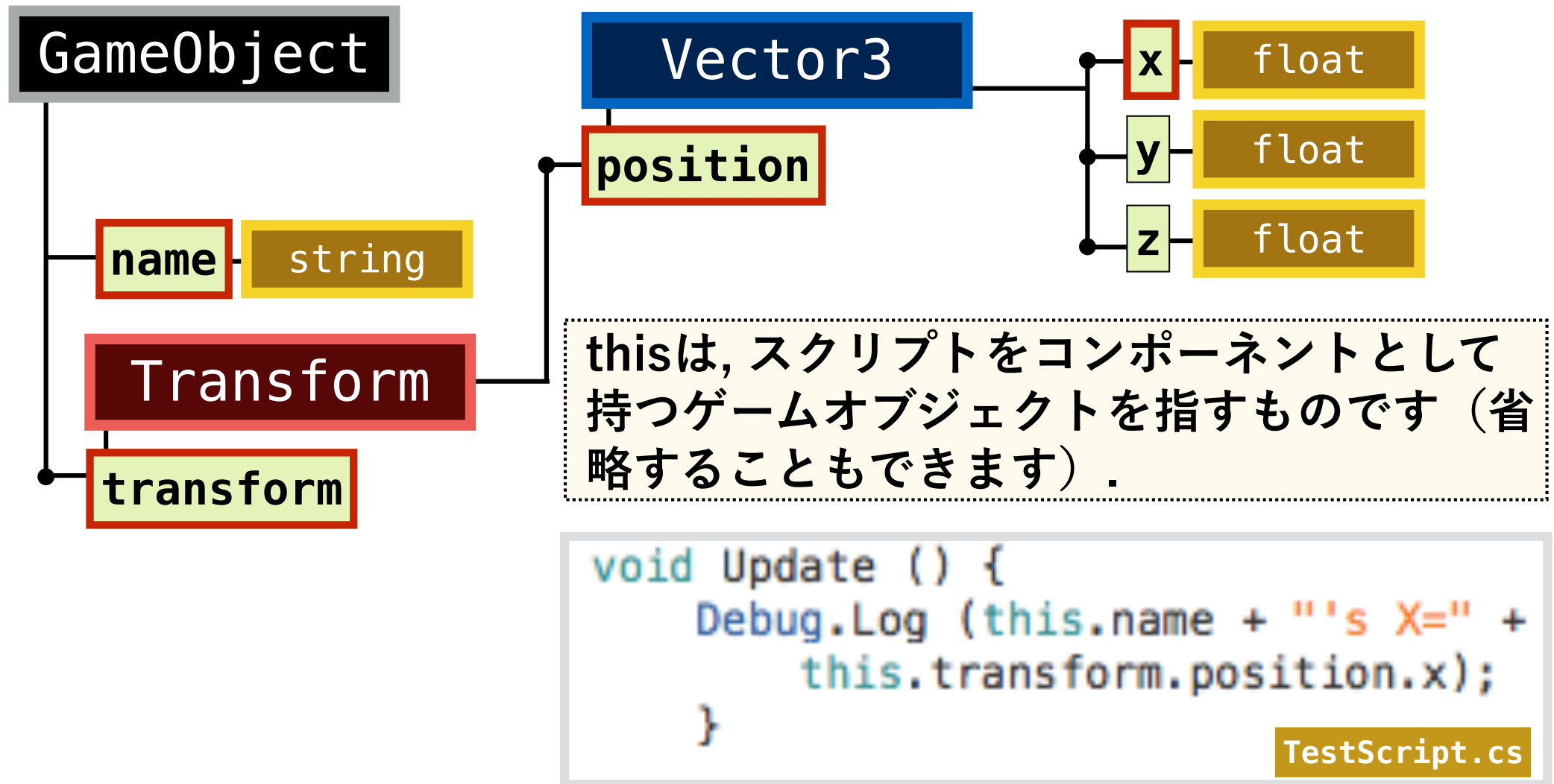


クラスとプロパティの再帰的關係

クラスはプロパティ（メンバ変数のようなもの：フィールドとほぼ同義）を持ち、それぞれのプロパティは、対応する個別のクラスに属し、そのクラスはまた独自のプロパティを持ち、…という関係を理解して下さい。



あるゲームオブジェクトのx座標にアクセスする



移動

- ゲームオブジェクトの移動は, transformプロパティに対して, Translateメソッドを適用します.

Transform

`void Translate(Vector3 v)`

三次元空間の移動

MyScript1.cs

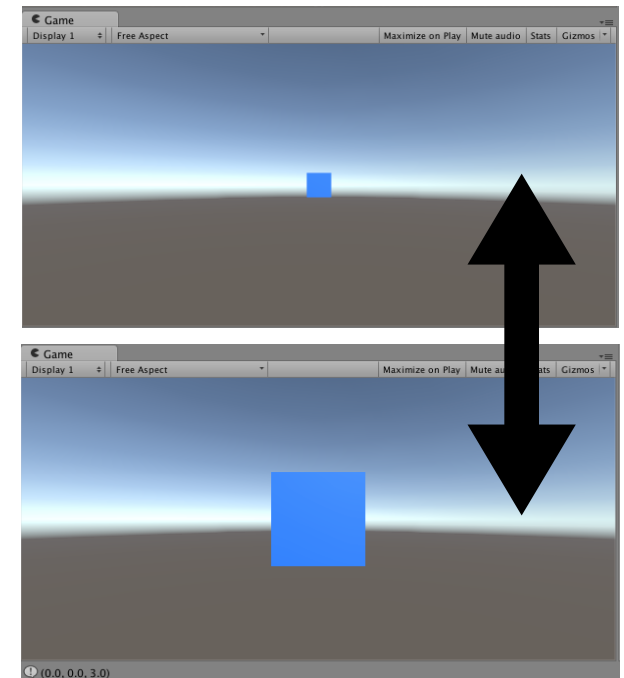
```
4 public class MyScript1 : MonoBehaviour {
5     int counter = 0;    //移動したフレーム数の記憶
6     float plus = 0.1f; //1フレームに進む距離
7
8     void Start () {
9     }
10
11    void Update () {
12        Vector3 p = new Vector3 (0, 0, plus);
13        this.transform.Translate (p);
14
15        // float zval = this.transform.position.z + plus;
16        // this.transform.position = new Vector3 (0, 0, zval);
17
18        // this.transform.position.z += plus;
19
20        counter++;
21        if (counter == 100) {
22            counter = 0;
23            plus *= -1;
24        }
25    }
26 }
```

これでもOK！！

NG！！

ゲームオブジェクトの位置を変えるためには, Translateメソッドを使用する他に, position プロパティに, 直接Vector3変数を代入する方法もありますが, x, y, zに対して, 直接的に値を代入することはできませんので注意してください.

実行結果



回転

- ゲームオブジェクトの回転は, Rotate関数を使います.

```
public class MyScript2 : MonoBehaviour {  
    public float plus = 0.1f;  
  
    void Start () {  
    }  
  
    void Update () {  
        Vector3 v = new Vector3 (plus, 0f, 0f);  
        this.transform.Rotate (v);  
  
        Debug.Log (this.transform.position);  
    }  
}
```

publicとすることで, インспекタビューからリアルタイムに値を変更することができます.

MyScript2.cs

Transform

void Rotate(Vector3 v)

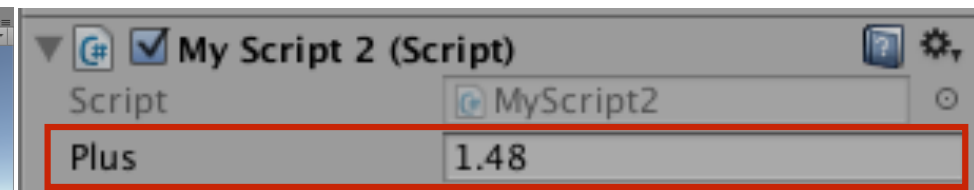
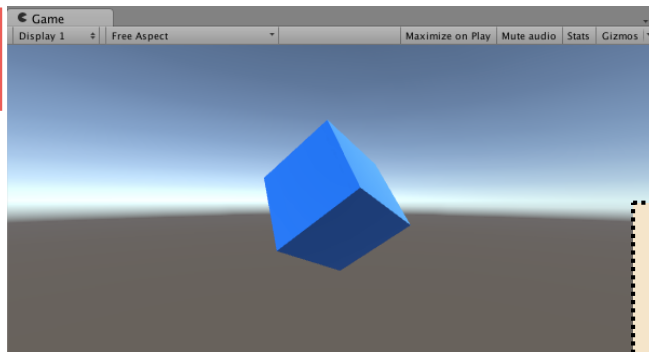
各回転軸に関して回転

Debug

void *Log(Object obj)

Debugクラスのクラスメソッドで, 引数の情報をコンソールに表示します.

実行結果



Plusの値を変えると, 回転のスピードを調整することができます. 360や359にするとどうなりますか?

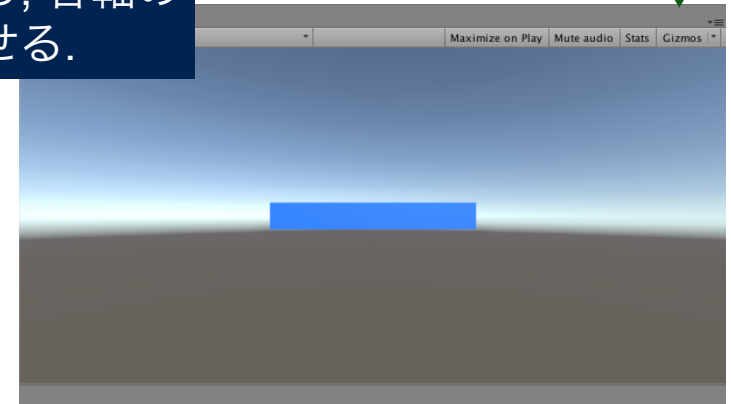
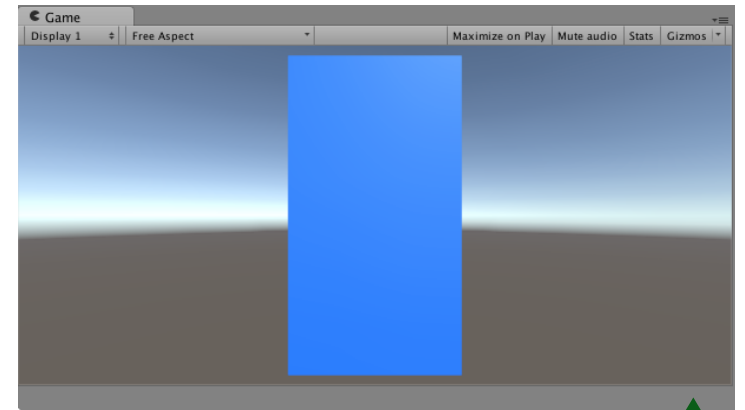
大きさを変える

- ゲームオブジェクトのサイズの倍率を変えるには, transformのlocalScaleプロパティに直接Vector3の値を代入します.

```
public class MyScript3 : MonoBehaviour {  
    float dx = 0.01f;  
    float dy = -0.01f;  
  
    void Start () {  
    }  
  
    void Update () {  
        Vector3 s = this.transform.localScale;  
  
        if (s.x > 3 || s.x < 0.1) {  
            dx *= -1;  
        }  
        if (s.y > 3 || s.y < 0.1) {  
            dy *= -1;  
        }  
        s.x += dx; s.y += dy;  
        transform.localScale = s;  
    }  
}
```

MyScript3.cs

実行結果



cubeのx軸 (y軸) 倍率が, 0.1未満あるいは3.0より大きくなったら, 各軸の変化分を反転させる.

小課題

- X軸方向に関して, 3秒間に 1 周の割合で, MyCubeを回転させてください (MyScript4.csとしましょう) .

ヒント

1フレームの回転量 (deg)

Time

`<float> *deltaTime`

Timeクラスのクラス変数.
Update関数実行間のインターバル (秒), すなわちフレームの時間を保持.

= 回転速度 (deg / s) × 1フレームの時間 (s)

3秒間で360deg, 1秒間では?

Time.deltaTime

Input

bool *GetKey(KeyCode key)

keyが押されている間, true

bool *GetKeyDown(KeyCode key)

keyを押したときに一度だけ, true

bool *GetKeyUp(KeyCode key)

keyを離れた時に一度だけ, true

キーイベント

A

KeyCode.A

→

KeyCode.RightArrow

K

KeyCode.K

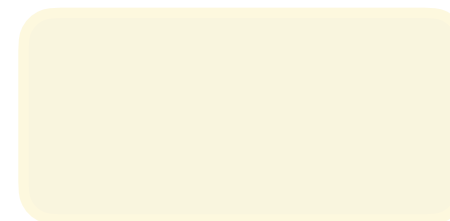
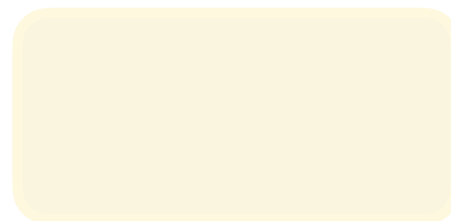
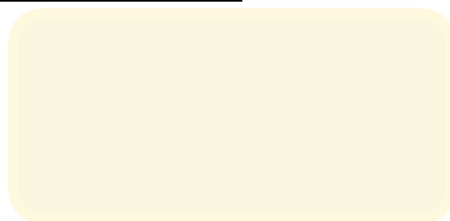
KeyCode.Space

X

KeyCode.X

↵

KeyCode.Return



GetKey

GetKeyDown

GetKey

GetKey

GetKeyUp

Input

bool *GetMouseButton(int n)

マウスが押されている間, true

bool *GetMouseDown(int n)

マウスを押したときに一度だけ, true

bool *GetMouseButtonUp(int n)

マウスを離した時に一度だけ, true

マウスイベント

マウス番号 <int>

0

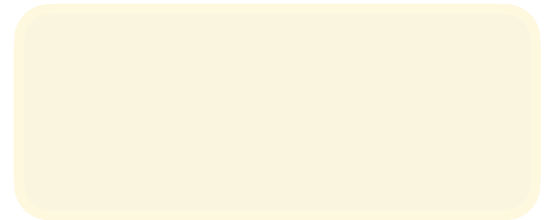
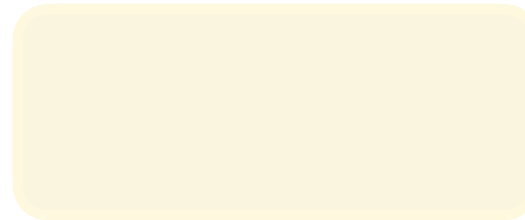
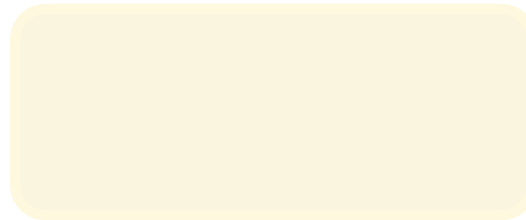
左ボタン

1

右ボタン

2

中央ボタン



GetMouseButton

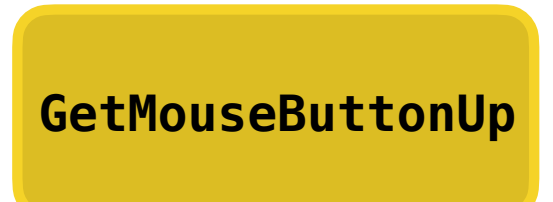
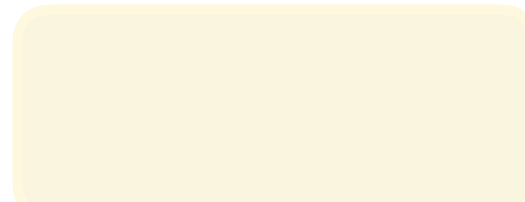
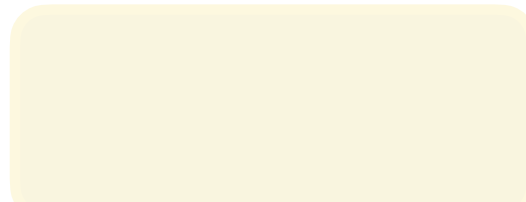
GetMouseDown



GetMouseButton



GetMouseButton



キーイベント

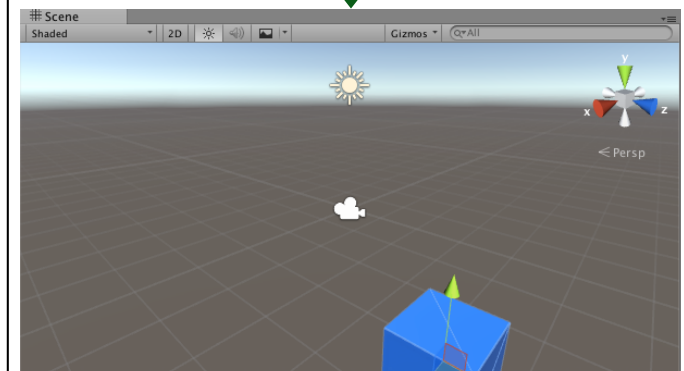
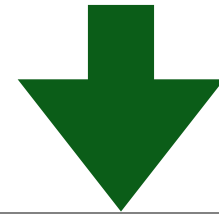
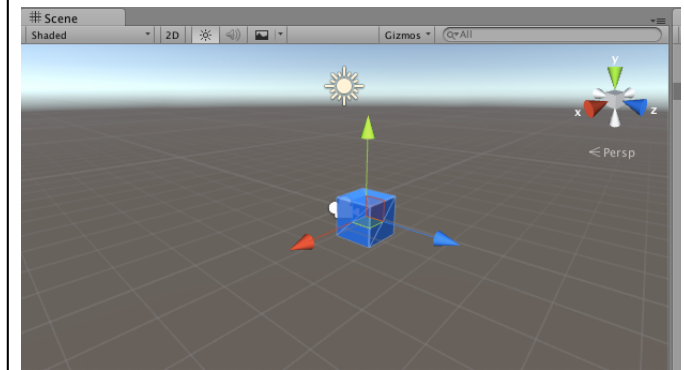
- 上下左右ボタンで、MyCubeをシーン内で動かします。

```
public class MyScript5 : MonoBehaviour {  
  
    public float updvel = 0.1f;  
  
    void Start () {  
    }  
  
    void Update () {  
        if (Input.GetKey (KeyCode.UpArrow)) {  
            transform.Translate(new Vector3(0,0,updvel));  
        }  
        if (Input.GetKey (KeyCode.DownArrow)) {  
            transform.Translate(new Vector3(0,0,-updvel));  
        }  
        if (Input.GetKey (KeyCode.RightArrow)) {  
            transform.Translate(new Vector3(updvel,0,0));  
        }  
        if (Input.GetKey (KeyCode.LeftArrow)) {  
            transform.Translate(new Vector3(-updvel,0,0));  
        }  
    }  
}
```

MyScript5.cs

実行結果

上ボタンと右ボタンを同時に押した時（シーンビュー）



小課題

- MyScript5.csに追記をして, マウスを押したときに, Cubeが初期位置に戻るようにしてください.

ヒント



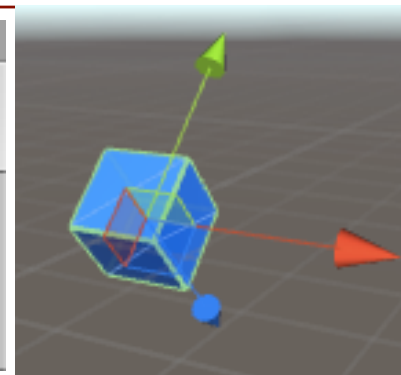
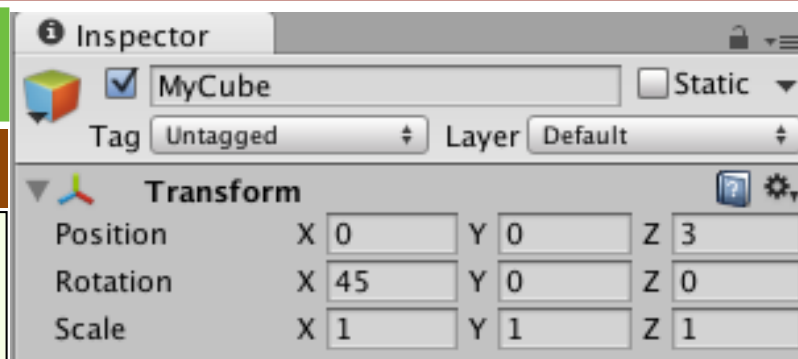
回転について補足

- インспекタビューにおいて、ゲームオブジェクトの角度は三次元（オイラー角）で表現されていますが、**transformオブジェクトのプロパティであるrotationの型は、Quaternionであることに注意が必要です。**

Quaternion

Quaternion *Euler(Vector3 euler_rot)

オイラー角（euler_rot）をQuaternion型の変数に変換する。



QuaternionのクラスメソッドであるEulerを使うと、オイラー角を使って、オブジェクトの角度を指定することができます。以下のスクリプトを起動して、オイラー角とQuaternionとの関係を確認してください。

```
public class MyScript6 : MonoBehaviour
{
    public Vector3 erot = new Vector3 ();
    private Quaternion qrot;
    public float qrot_x,qrot_y,qrot_z,qrot_w;

    void Start(){}

    void Update(){
        this.transform.rotation = Quaternion.Euler(erot);
        qrot = transform.rotation;

        qrot_x = qrot.x; qrot_y = qrot.y;
        qrot_z = qrot.z; qrot_w = qrot.w;
    }
}
```

MyScript6.cs

transform.rotation = erot;
ではエラーとなることに注意！！

実行結果

Script			
# MyScript6			
Erot	X 45	Y 0	Z 0
Qrot_x	0.3826835		
Qrot_y	0		
Qrot_z	0		
Qrot_w	0.9238795		

QuaternionとVector3の変換！！

クォータニオン

Quaternion

Ⓚ

X

0.6706209

Y

0

Z

0

W

0.7418003

Ⓚ = Quaternion.Euler(Ⓥ);

Ⓥ = Ⓚ.eulerAngles;

オイラー角

Vector3

Ⓥ

X

84.23003

Y

0

Z

0