3D ゲームプログラミングⅡ/ゲームプログラミング B
第 2 回　Unity の復習

移動

座標を指定する。

```
transform.position = new Vector3(x, y, z);
```

移動量を指定する。

```
transform.Translate(x, y, z);
```

目標点に向かって速度 v で移動する。

```
transform.position = Vextor3.MoveTowards(transform.position,
        new Vector3(x, y, z), v * Time.deltaTime);
```

Rigidbody に速度ベクトルを与えて移動させる。

```
GetComponent<Rigidbody>().velocity = new Vector3(x, y, z);
```

Rigidbody に力を加えて移動させる。

```
GetComponent<Rigidbody>().AddForce(x, y, z);
```

障害物を避けて目標点まで移動する。

NavMesh の作成

キャラクターに NavMeshAgent コンポーネントを追加

```
using UnityEngine.AI;
GetComponent<NavMeshAgent>().destination = new Vector3(x, y, z);
```

回転

角度を指定する。

```
transform.rotation = Quatenion.Euler(x, y, z);
```

角度の変化量を指定する。

```
transform.Rotate(x, y, z);
```

回転の中心座標、回転軸ベクトル、回転角を指定する。

```
transform.RotateAround(new Vectior3(x, y, z), new Vector3(u, v, w), angle);
```

Rigidbody に角速度を与えて回転させる。

```
GetComponent<Rigidbody>().angularVelocity = new Vector3(x, y, z);
```

Rigidbody にトルクを加えて回転させる。

```
GetComponent<Rigidbody>().AddTorque(x, y, z);
```

拡大・縮小

拡大率を指定する。

```
transform.localScale = new Vector3(x, y, z);
```

キーボード入力

特定のキーが押されているかどうか調べる。

```
Input.GetKey(KeyCode)
```

特定のキーが押された瞬間かどうか調べる。

```
Input.GetKeyDown(KeyCode)
```

特定のキーが離された瞬間かどうか調べる。

```
Input.GetKeyUp(KeyCode)
```

←キー、→キーの状態を調べて、-1～1 の間の値を返す（ジョイスティック対応）。

```
Input.GetAxis("Horizontal")
```

↑キー、↓キーの状態を調べて、-1～1 の間の値を返す（ジョイスティック対応）。

```
Input.GetAxis("Vertical")
```

Ctrl キーが押された瞬間かどうか調べる（ゲームコントローラーのボタン対応）。

```
Input.GetButtonDown("Fire1")
```

マウス入力

マウスの左ボタンが押されているかどうか調べる。

```
Input.GetMouseButton(0)
```

マウスの左ボタンが押された瞬間かどうか調べる。

```
Input.GetMouseButtonDown(0)
```

マウスの左ボタンが離された瞬間かどうか調べる。

```
Input.GetMouseButtonUp(0)
```

マウスのスクリーン座標。

```
Input.mousePosition
```

マウスのワールド座標。

```
Camera.main.ScreenToWorldPoint(Input.mousePosition)
```

マウスでクリックされたオブジェクトを特定する。

```
if (Input.GetMouseButtonDown(0))
{
        Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition);
        RaycastHit hit = new RaycastHit();
        if (Physics.Raycast(ray, out hit))
        {
                GameObject result = hit.collider.gameObject;
        }
}
```

コライダー上でクリックされた場合の処理

コライダーを持つオブジェクトに追加されたスクリプトで、OnMouseDown メソッド

オーディオ

音を再生する。

オブジェクトに AudioSource を追加し、AudioClip を設定

GetComponent<AudioSource>().Play();

音の再生を止める。

GetComponent<AudioSource>().Stop();

テキスト

文字列を表示する。

UI の Text を使用

文字列をスクリプトで書き替える。

```
using UnityEngine.UI;
public Text msg;
msg.text = “書き替え後の文字列”;
```

文字列を入力する。

UI の InputField を使用

ボタン

ボタンがクリックされた場合の処理。

スクリプトに処理を行うメソッド追加

UI の Button を作成

OnClicked()にメソッド名を設定

タグ

オブジェクトにタグを付けることによって、オブジェクトの識別ができる。

```
gameObject.tag
```

プレハブ

プレハブのインスタンス化。

```
public GameObject objPrefab;
Instantiate(objPrefab, new Vector3(x, y, z), Quaternion.identity);
```

はね返り

コライダーに PhysicsMaterial を追加し、Bounciness の値を調整する。

色の変更

Material の色を変更する。

GetComponent<Renderer>().material.color = Color(r, g, b);

透明度

Material の色の α 値を変更する。

Material の Rendering Mode を Fade または Transparent に設定

Renderer renderer = GetComponent<Renderer>();

Color color = renderer.material.color;

color.a = 0.2f;

renderer.material.color = color;

衝突判定

OnCollisionEnter イベントを使用する方法。

OnTriggerEnter イベントを使用する方法。

コライダーの IsTrigger をオンにする。

アニメーション

Animation コンポーネントを使用する方法。

Animation ウィンドウでアニメーションクリップ作成

オブジェクトに Animation コンポーネントを追加

Animation コンポーネントにアニメーションクリップを設定

Mecanim を使用する方法。

Animation Controller を作成。

Parameter を作成。

State を作成。

Transition を作成。

データのロード／セーブ

int 型の変数 i の値を SCORE という名前のキーに保存する。

PlayerPrefs.SetInt("SCORE", i);

PlayerPrefs.Save();

float 型の変数 x の値を PositionX という名前のキーに保存する。

```
PlayerPrefs.SetFloat(“PositionX”, x);
PlayerPrefs.Save();
```

string 型の変数 name の値を NAME という名前のキーに保存する。

```
PlayerPrefs.SetString(“NAME”, name);
PlayerPrefs.Save();
```

SCORE という名前のキーに保存されている int 型の値を読む。

```
i = PlayerPrefs.GetInt(“SCORE”);
```

PositionX という名前のキーに保存されている float 型の値を読む。

```
x = PlayerPrefs.GetFloat(“PositionX”);
```

NAME という名前のキーに保存されている string 型の値を読む。

```
name = PlayerPrefs.GetString(“NAME”);
```

乱数

0～1 の間の乱数を取得する。

```
Random.value
```

min～max の間の乱数を取得する。

```
Random.Range(min, max)
```

オブジェクトの参照

Inspector ウィンドウで指定する方法。

public で GameObject を定義し、Inspector ビューで設定する。

オブジェクト名で探す方法。

```
GameObject.Find(“Object1”)
```

他のクラスの参照

public static で定義する方法。

public static で定義された変数・メソッドは、他のクラスから参照可能

static クラスを使用する法。

static クラス内の変数・メソッドは、他のクラスから参照可能

オブジェクト名とクラス名（スクリプト名）から参照する法。

static クラス内の変数・メソッドは、他のクラスから参照可能

```
GameObject.Find(“Object1”).GetComponent<Script1>().Value
```

シーンの切り替え

Stage1 という名前のシーンに移行する。

```
using UnityEngine.SceneManagement;
SceneManager.LoadScene (“Stage1”);
```

アプリケーションの終了

```
Application.Quit();
```