Unity

プログラミング
基礎編

〈発行〉 Programming School ZERO ONE studio プログラミングスクール ゼロワンスタジオ

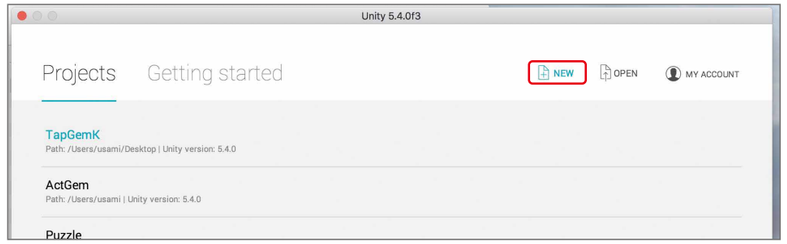
〒470-0136 日進市竹の山3丁目412 ザ・ネクステージ 3F
Tel／Fax.0561-73-0132　E-mail pgs@zero-one-studio.com

Programming School ZERO ONE studio プログラミングスクール ゼロワンスタジオ

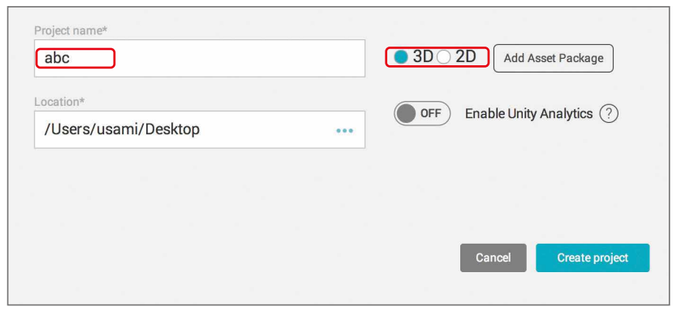
基礎編

プロジェクト作成

新しくプログラムを作成するときは、新しいプロジェクトを作ります。
プロジェクトは、プログラムの材料が全部入っている箱のイメージです。



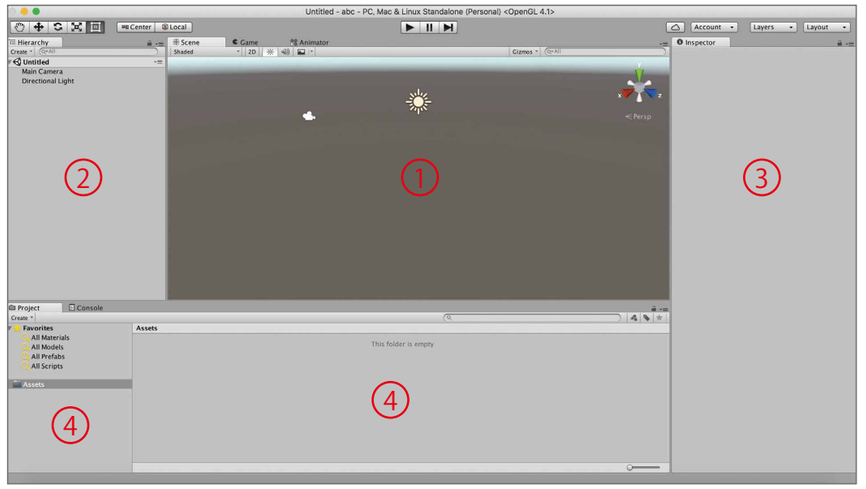
NEW（ニュー：新しい）をクリックします。



Project name（プロジェクトネーム：プロジェクト名）を入力します。
3Dまたは2Dを選択します。
Location（ロケーション：場所）は、プロジェクト名のフォルダを作る場所です（変更する必要はありません）。
Enable Unity Analytics（イネーブル・ユニティー・アナリティクス）はOFFとします。

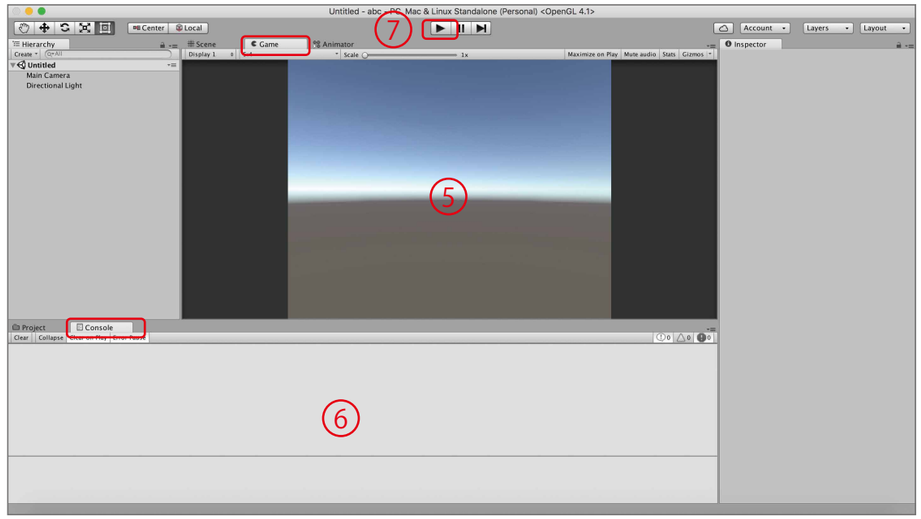
Unityの画面説明

ここでは、3Dの画面で説明します。2Dも同様となります。



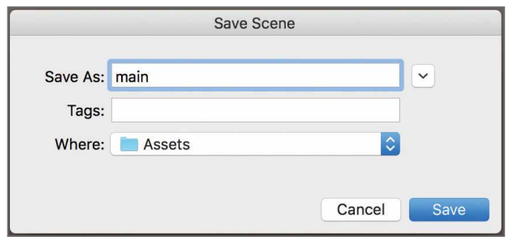
①Scene（シーン：場面）ビュー
　プログラム（ゲーム）の世界を、ここに作ります。
②Hierarchy（ヒエラルキー：階層）ビュー
　シーンビューにある部品（オブジェクト）のリストです。
③Inspector（インスペクター：調査）ビュー
　ヒエラルキービューで選択された部品（オブジェクト）の詳細なデータを表示します。
④Project（プロジェクト）ビュー
　プロジェクトの中に入っているものが表示されます。

以下では、シーン、ヒエラルキー、インスペクター、プロジェクトと表記します。
これらは、重要ですので、最初にしっかりと覚えてください。



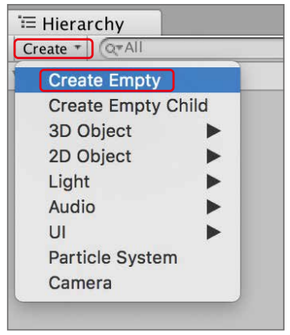
赤枠で示したタブを切り換えると、シーン、プロジェクトとは別の情報が見られます。
⑤Game（ゲーム）ビュー
　Main Cameraで映し出された画像です。これがゲーム画面となります。
⑥Console（コンソール：制御）ビュー
　おもにログと呼ばれる情報を表示します。エラーのログもここに表示されます。
⑦再生ボタン
　右向きの三角マークの付いたボタンが再生ボタンです。
　このボタンを押すことで、作ったプログラムが実行できます。
　もう一度押すと、実行停止します。

プロジェクトを作ったら、シーンのセーブ（保存）をしておきます。
コマンドキーを押しながら、Sを押します。以下、command+Sと書きます。

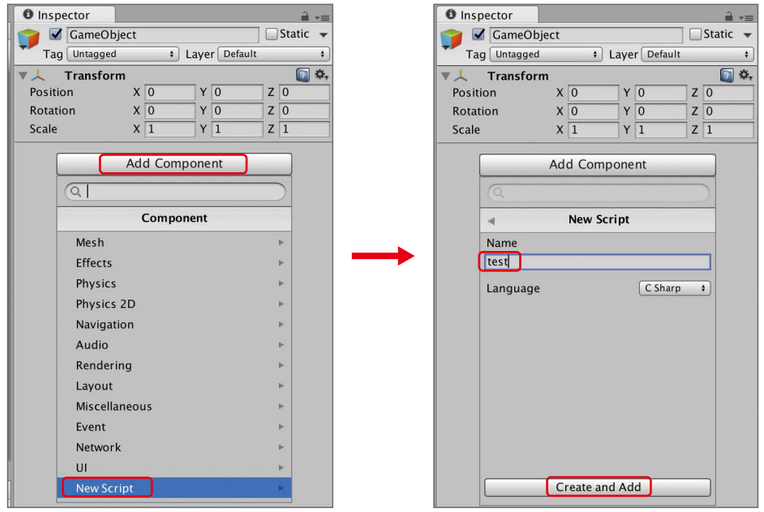


mainという名前でセーブしておきます。
この後は、作業の途中でこまめにcommand+Sでセーブします。
2回目からは上書き保存されますので、名前を入れる必要はありません。

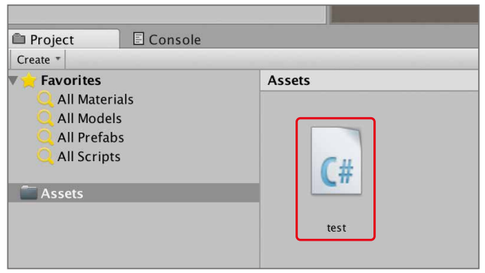
新しいスクリプトを作る



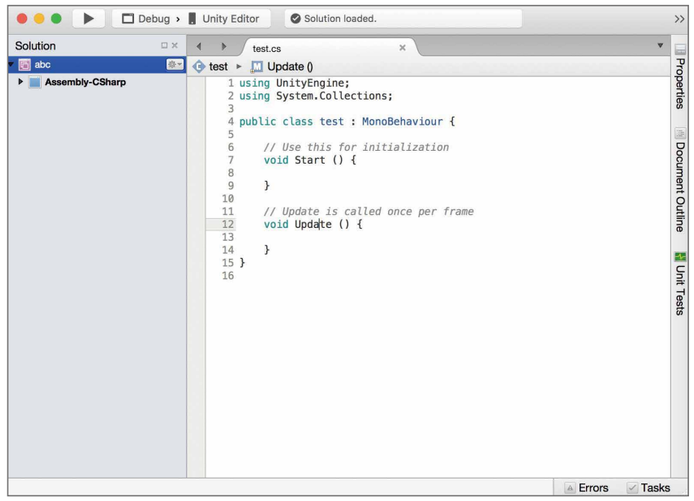
ヒエラルキーのCreate（クリエイト：生成）をクリックします。
Create Empty（クリエイト・エンプティー：空オブジェクトの生成）をクリックします。
ヒエラルキーに、GameObjectという名前のオブジェクトが生成されます。
インスペクターを見ると、Transform（トランスフォーム）というComponent（コンポーネント：構成要素）が付いています。
これは形もなにもない空のオブジェクトですが、トランスフォームと呼ばれる情報は持っています。
GameObject（ゲームオブジェクト）にスクリプトを付けます。



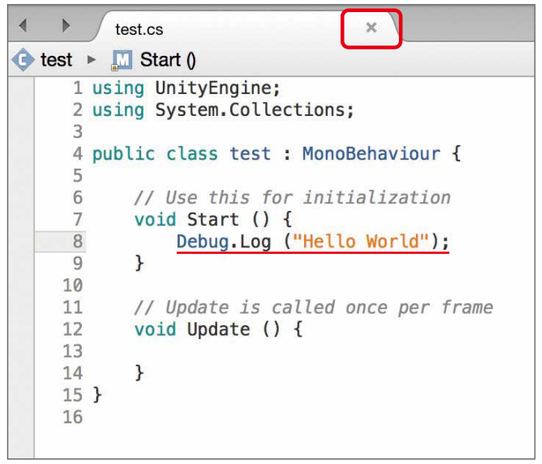
インスペクターのAdd　Component（アッド・コンポーネント：構成要素の追加）をクリックし、New Script（ニュー・スクリプト：新しいスクリプト）をクリックします。
Name（ネーム：名前）をtest（テスト）とします。
Language（ランゲージ：言語）に、C Sharp（シー・シャープ）を選択します。
Create and Add（クリエイト・アンド・アッド：生成して追加）をクリックします。



プロジェクトのAssets（アセット）の中に、testというスクリプトファイルが出来ました。
これが、プログラムの記述されるファイルです。
このファイルをダブルクリックします。
すると、下のようなMonoDevelop（モノデベロップ）の画面が開きます。
これがプログラムを編集するためのエディタです。

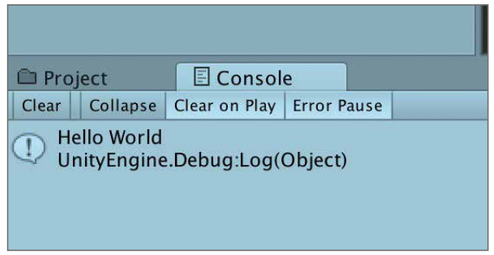


スクリプトを編集する



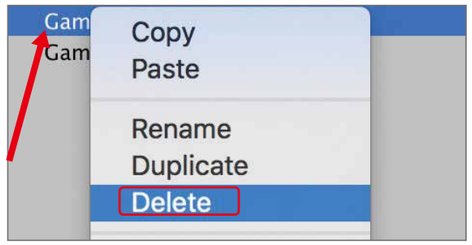
赤線のように1行追加します。
セーブ（保存）していないうちは赤枠の中のマークが丸印となります。
command+Sでセーブ（保存）します。
セーブ（保存）すると、赤枠の中のマークがバツ印になります。

再生ボタンを押すと、コンソールにHello Worldの文字が表示されます。



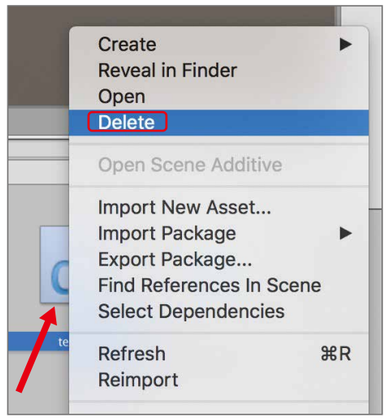
出来ましたか？
課題
（1）もう一つ空のオブジェクトを作ってください。
（2）新しいスクリプトtest2を付けてください。
（3）自分の名前がコンソールに表示されるようにプログラミングしてください。

ゲームオブジェクトを削除する



ヒエラルキーで、GameObjectを右クリックします。
右クリックは、controlキーを押しながらタッチパッドをクリックします。
二本の指で同時にタッチパッドをクリックしても右クリックとなります。
開いたメニューから、Delete（デリート：消去）をクリックします。

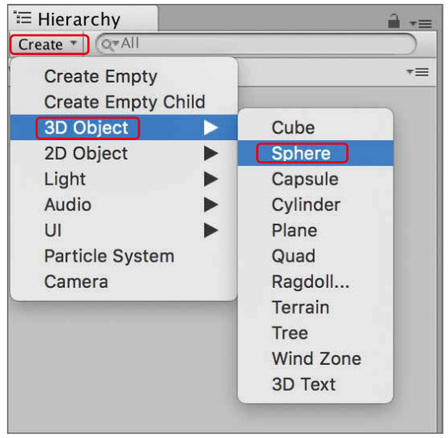
testという名前のスクリプトファイルも消します。



Assets（アセット：資源）の中のtestを右クリックします。
開いたメニューから、Delete（デリート：消去）をクリックします。

出来ましたか？
課題
（１）GameObject (1)を消してみましょう。
（２）test2を消してみましょう。

球を作る



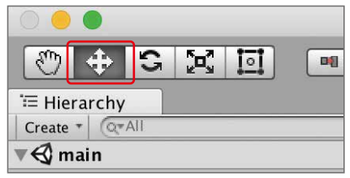
ヒエラルキーで、Create（クリエイト：生成）、3D　Object（3Dオブジェクト）、Sphere（スフィア：球）の順にクリックします。

ヒエラルキーにSphere（スフィア）という名前のオブジェクトが追加されます。
シーン、ゲームのビューでも確認しましょう。

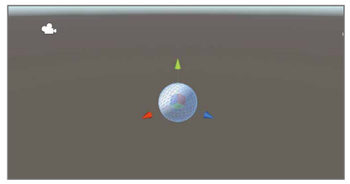
再生ボタンを押してみましょう。球が映っていますか？

球のTramsform(トランスフォーム)を変更する

シーンに球が映っている状態にします。
映っていないときは、ヒエラルキーのSphere(スフィア)をダブルクリックします。
シーンにカーソルがあるときに、タッチパッドを2本指で上下になぞるとシーンが拡大縮小できます。



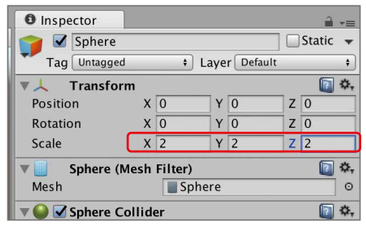
左上のボタンで十字形のボタンを押します。
次のように球が表示されます。



赤い矢印をドラッグ(一方の指でクリックしつつ、もう一方の指でスライドさせます)すると球がX方向へ動きます。
このとき、TransformのXの値が変わることを確かめてください。

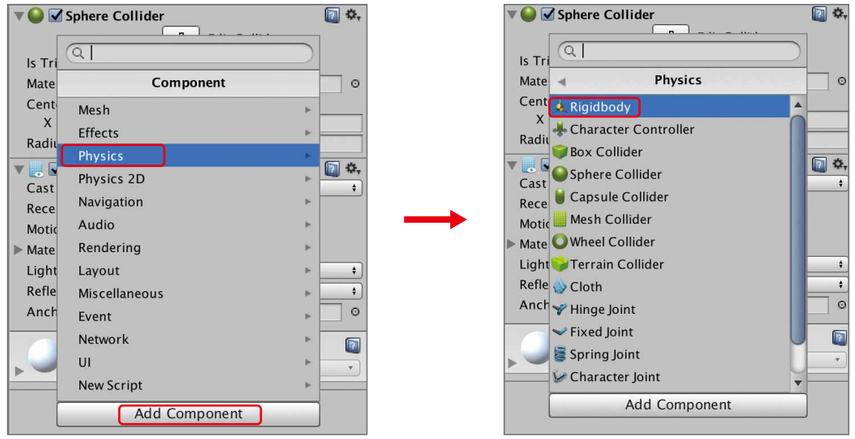
再生してみましょう。動かした分だけ球が横へ移動していますか?

課題
(1)Y方向へも動かしてください。
(2)Z方向へも動かしてください。
(3)もとの位置へ戻してください。Position(ポジション:位置)に、0,0,0を入力します。
Scale(スケール:大きさ)を変えてみます。



再生してみましょう。カメラに映る球が大きくなっていたら成功です。

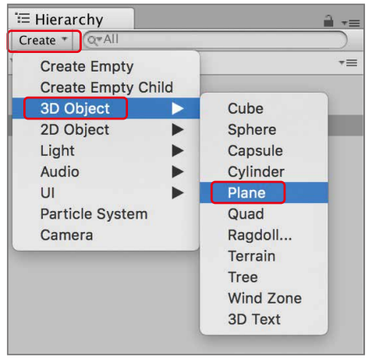
Rigidbody(リジッドボディ:剛体)を球につける



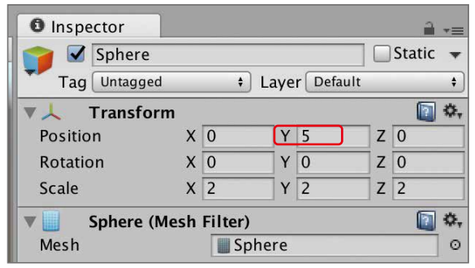
ヒエラルキーでSphere(スフィア)を選択します。
インスペクターで、Add Component(アッド・コンポーネント:構成要素の追加)をクリックします。
Physics(フィジックス:物理)をクリックし、Rigidbody(リジッドボディ:剛体)をクリックします。

再生してみましょう。球が重力で下へ落ちたら成功です。

床を作る



ヒエラルキーで、Create(クリエイト:生成)、3D Object(3Dオブジェクト)、Plane(プレーン:平面)の順にクリックします。Transform(トランスフォーム)のPosition(ポジション:位置)がずれていたら0,0,0にします。

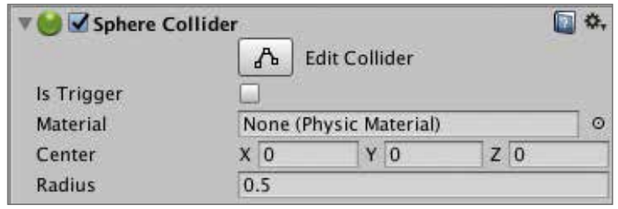


ヒエラルキーでSphere（スフィア）を選択し、インスペクターでTransform（トランスフォーム）のPosition（ポジション：位置）のYを5にします。

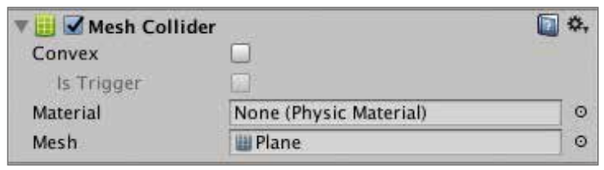
再生してみましょう。球が床まで落ちて止まれば成功です。

床で止まるのは、Sphere（スフィア）にも、Plane（プレーン）にも、当たり判定がついているからです。
この当たり判定のことをCollider（コライダー：衝突装置）と呼びます。

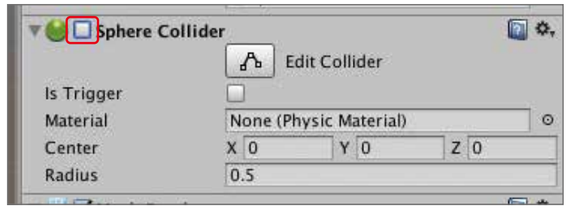
ヒエラルキーでSphere（スフィア）を選択してインスペクターを見ると、Sphere　Collider（スフィア・コライダー：球形の衝突装置）がついていることがわかります。



ヒエラルキーでPlane（プレーン）を選択してインスペクターを見ると、Mesh Collider（メッシュ・コライダー：網形の衝突装置）がついていることがわかります。



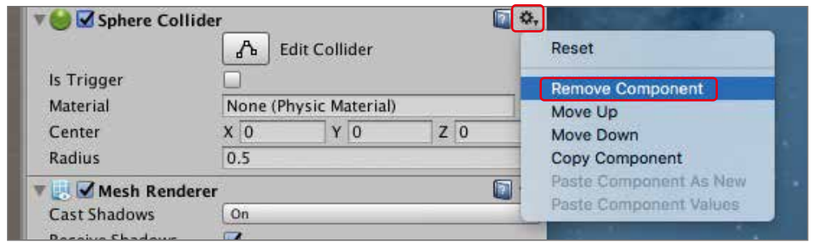
Collider（コライダー:衝突装置）を外す



ヒエラルキーでSphere（スフィア）を選択します。
インスペクターでSphere Collider（スフィア・コライダー：球形の衝突装置）のチェックを外します。
チェックを外すことで、コンポーネント（構成要素）として付いているCollider（コライダー：衝突装置）の機能がオフになります。

再生してみましょう。球が平面を通過すれば成功です。

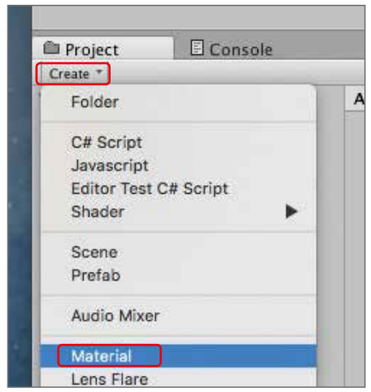
次に、コンポーネント自体を削除する方法を説明します。



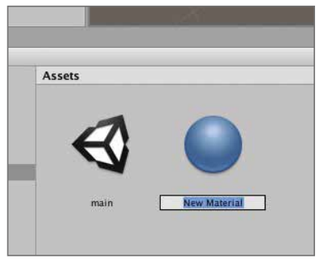
インスペクターでSphere Collider（スフィア・コライダー：球形の衝突装置）の右上の歯車をクリックします。
Remove Component（リムーブ・コンポーネント：構成要素の削除）をクリックします。
これによりコンポーネント（構成要素）自体が取り外されます。
再生してみましょう。球が平面を通過すれば成功です。

このように、オブジェクトの一方のコライダー（衝突装置）を取り外す（あるいは機能をオフ）にすることで、当たり判定がなくなり、ぶつからずにすり抜けます。

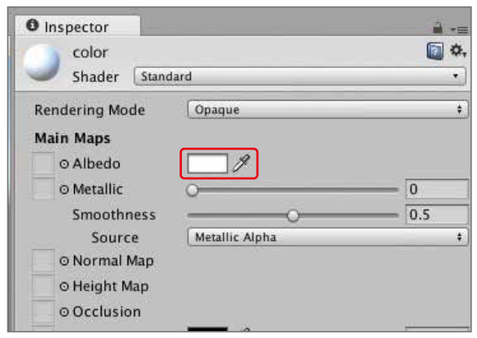
球に色をつける



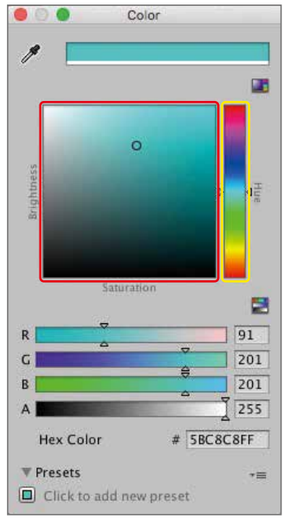
プロジェクトで、Create（クリエイト：生成）、Material（マテリアル：材質）の順にクリックします。
※ヒエラルキーではなく、プロジェクトの方のCreate（クリエイト：生成）です。



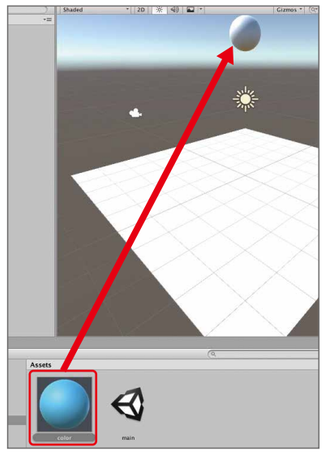
Assets（アセット）の中に、New Material（ニュー・マテリアル）が生成されます。
名前を、colorとします。



インスペクターで、Albedo（アルベド）の隣の四角い枠をクリックします。



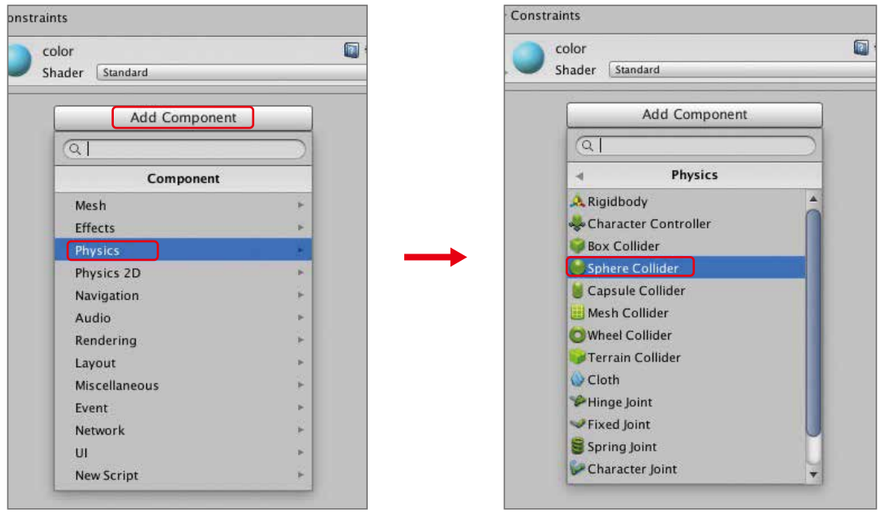
黄色の枠の中で好きな色をクリックし、赤色の枠の中で色の濃淡をクリックします。
好きな色に設定してみましょう。



Assets（アセット）のcolorを、シーン（場面）の球にドラッグ・アンド・ドロップします。
ドラッグ・アンド・ドロップは、片方の指でタッチパッドをクリックしたまま、もう片方の指でスライドさせて目標物の上でタッチしている指を離す操作です。以下、DDと書きます。
DDは重要な操作なので、しっかり練習しましょう。

課題
（1）プロジェクトのCreate（クリエイト：生成）で、もう一つMaterial（マテリアル：材質）を作ってください。
（2）Materialの名前を、color2としてください。
（3）好きな色を設定してください。
（4）これを、シーン（場面）の中のPlane（プレーン：平面）にDDしてください。

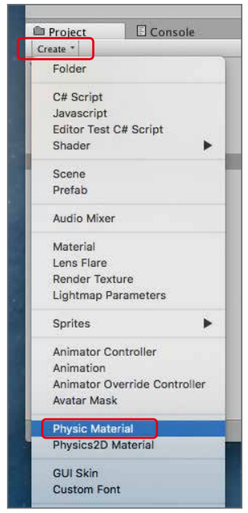
ヒエラルキーでSphere（スフィア）を選択して、Sphere Collider（スフィア・コライダー）をつけます。



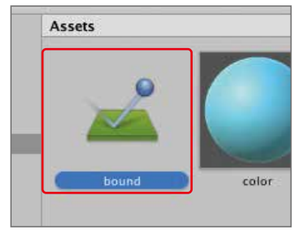
インスペクターで、Add Component（アッド・コンポーネント：構成要素の追加）をクリックします。
Physics（フィジックス：物理）をクリックし、Sphere Collider（スフィア・コライダー：球形の衝突装置）をクリックします。

再生してみましょう。色のついた球が色のついた平面に落ちて止まれば成功です。

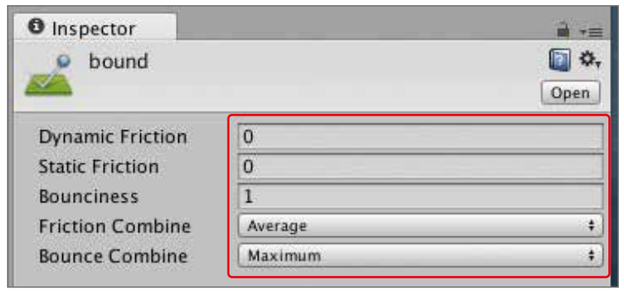
球を弾ませる



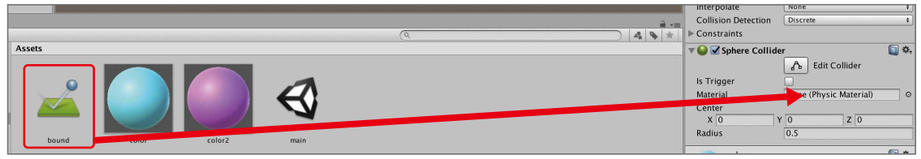
プロジェクトのCreate（クリエイト：生成）をクリックし、Physic Material（フィジック・マテリアル：物理的材質）をクリックします。



Assets（アセット）の中にNew Physic Material（ニュー・フィジック・マテリアル）が生成されます。名前をboundとします。

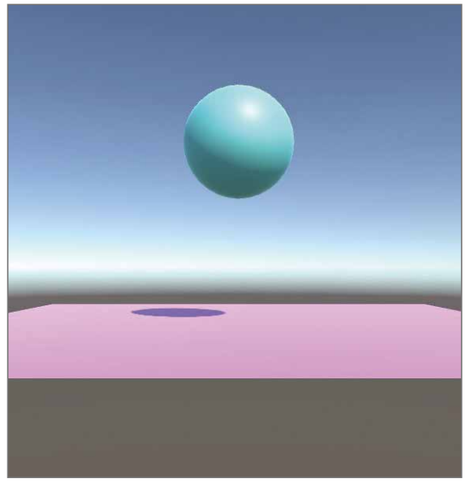


インスペクターで、上のように、設定します。摩擦と跳ね返りの設定です。
Assets（アセット）の中のbound（バウンド）を、Sphere Collider（スフィア・コライダー：球形の衝突装置）に設定します。



インスペクターのSphere Collider（スフィア・コライダー）のMaterial（マテリアル）の右側の枠の中へDDします。

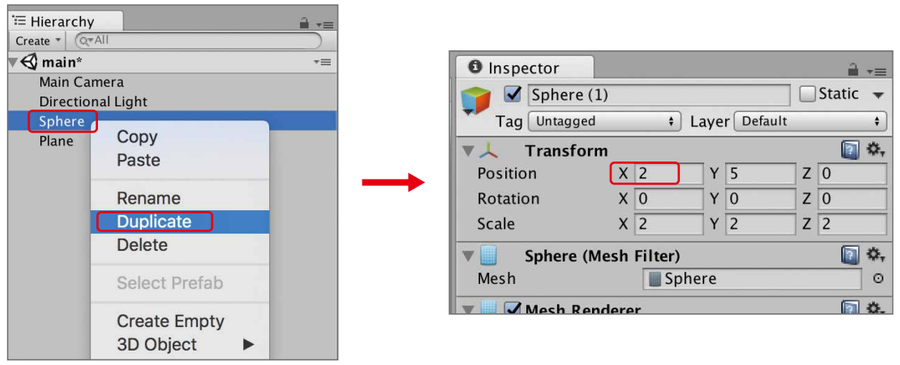
再生してみましょう。球が平面に衝突して勢いよく跳ね返れば成功です。



ボールの影もついていますね。

球を2つにする

ここでは、オブジェクトを複製する方法を覚えます。



ヒエラルキーで、Sphere（スフィア）を右クリックし、Duplicate（デュプリケイト：複製）をクリックします。Sphere (1)が生成されます。
※右クリックは、controlキーを押しながらタッチパッドをクリックします。または、タッチパッドを２本指で同時にクリックしても右クリックとなります。
※複製はヒエラルキーでSphere（スフィア）を選択した状態で、command+Dでも可能です。
インスペクターで、Transform（トランスフォーム）のPosition（ポジション：位置）のXを2とします。

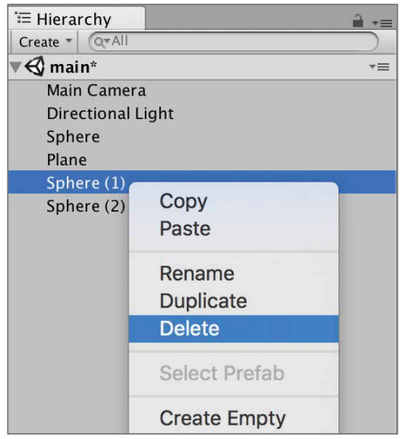
再生してみましょう。２つの球が並んで弾んだら成功です。

課題1
（１）もう一つ球を生成して３つにしましょう。
（２）３つめの球のTransform（トランスフォーム）のPosition（ポジション：位置）のXを-2とします。

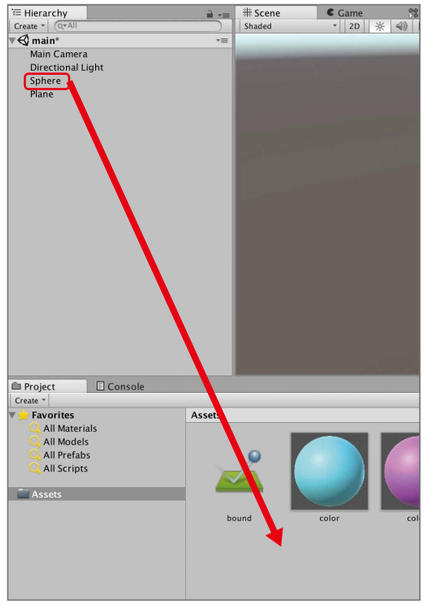
再生してみましょう。

Sphere（スフィア:球）をプレハブ化する

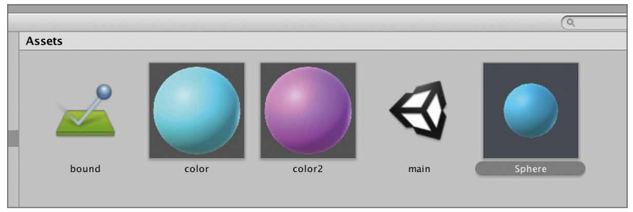
まず、Sphere (1)とSphere (2)を消去します。



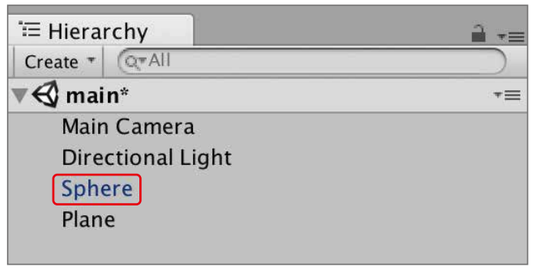
Sphere (1)を右クリックし、Delete（デリート：消去）をクリックします。
Sphere (2)についても同様に、右クリックし、Delete（デリート：消去）をクリックします。



ヒエラルキーのSphere（スフィア）を、Assets（アセット）の中へDDします。



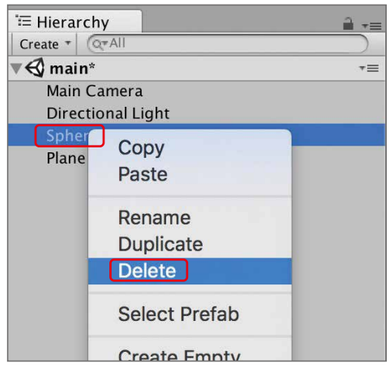
Assets（アセット）の中に、Sphere（スフィア）というプレハブが生成されます。
※プレハブにすることで、このプレハブを使ってスクリプトからSphere（スフィア）を作り出すことができます。



ヒエラルキーのSphere（スフィア）は、紺色になっています。
プレハブが印鑑、オブジェクトが印影というイメージです。

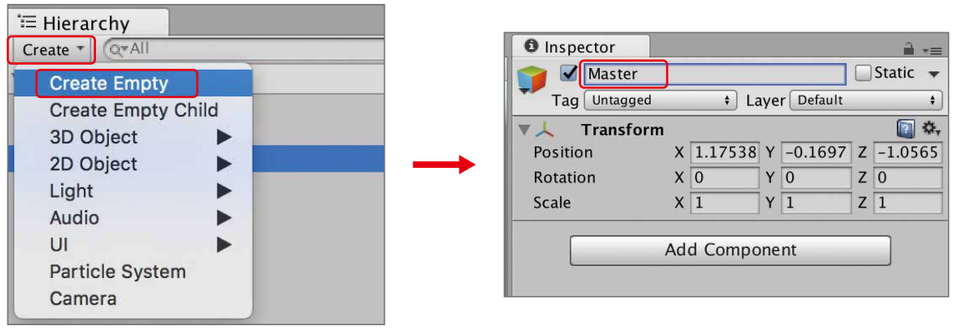
次は、プレハブを使ってスクリプトで球を生成します。

準備として、Sphere（スフィア）をDelete（デリート：消去）します。

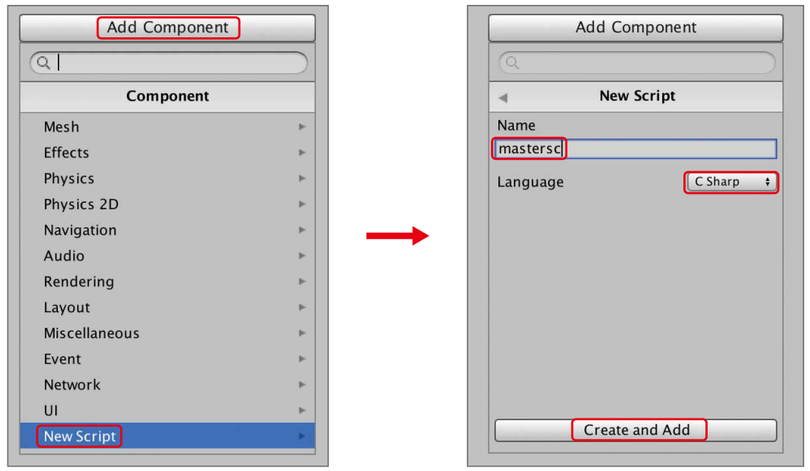


Sphere（スフィア）を右クリックし、Delete（デリート：消去）をクリックします。
※ヒエラルキーからSphere（スフィア）がなくなりますが、Sphere（スフィア）を作り出すプレハブがあるので、大丈夫です。

マスターオブジェクトを作る

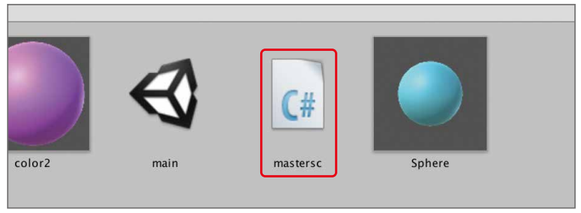


ヒエラルキーで、Create（クリエイト：生成）をクリックし、Create Empty（クリエイト・エンプティー：空オブジェクトの生成）をクリックします。
インスペクターで、名前を、Masterとします。名前の入力後にエンターキーを押します。
※エンターキーを押さないと反映されませんので注意してください。

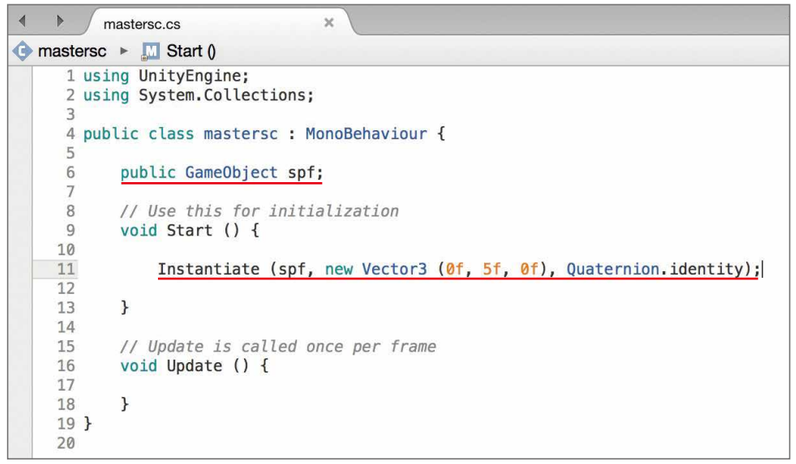


インスペクターで、Add Component（アッド・コンポーネント：構成要素の追加）をクリックします。
New Script（ニュー・スクリプト：新しいスクリプト）をクリックします。
Name（ネーム：名前）を、masterscとします。
Language（ランゲージ：言語）は、C Sharpを選択します。
Create and Add（クリエイト・アンド・アッド：生成して追加）をクリックします。

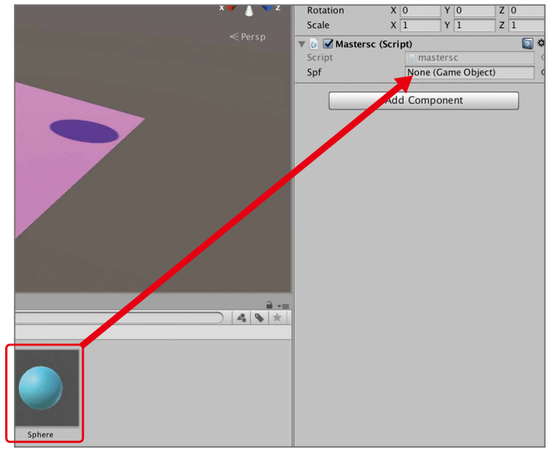
スクリプトを編集する



スクリプトファイルmasterscをダブルクリックします。



上の2行を追加します。追加したら、command+Sでセーブ（保存）します。



ヒエラルキーでMaster（マスター）を選択します。
インスペクターのMastersc(Script)のSpfの枠内にプレハブ化したSphereをDDします。

再生してみましょう。再生と同時にSphere（スフィア：球）が作られます。

解説

```
public GameObject spf;
```

プレハブを入れるためのspfという名前の変数を宣言しています。
GameObject（ゲームオブジェクト）型でspfを準備するという意味です。
public（パブリック）を先頭に書くことで、インスペクターにSpfの枠が出てきます。

```
// Use this for initialization
void Start () {

    Instantiate (spf, new Vector3 (0f, 5f, 0f), Quaternion.identity);

}
```

プログラムの実行時に最初に一回だけ呼ばれるのがスタート関数です。
最初が//のオレンジ色の行は、コメント行です。プログラムに関係のない覚書です。
Use this for initialization（ユーズ・ディス・フォー・イニシャライゼイション）とあり、初期化のために使ってください、となっています。
スタート関数の処理は、黄色枠で示した中かっこの中に書きます。
今は一行だけ処理（メソッド）が記述されています。
Instantiate（インスタンシエイト）は、プレハブからオブジェクトを作り出すときに使います。
小かっこの中に記述されるものを「引数」といいます。
最初の引数「spf」には、プレハブのsphere（スフィア）が入っています。
2番目の引数「new Vector3 (0f, 5f, 0f)」（ニュー・ベクタースリー）は、sphere（スフィア：球）を作る位置を示します。0, 5, 0は、XYZの座標値です。
3番目の引数「Quaternion.identity」（クオーターニオン・アイデンティティー）は、回転させずにそのままの角度でsphere（スフィア）を作ることを示します。

課題
（1）XYZの座標値がそれぞれ、-2, 4, 0となるように、2つ目の球を作ってください。
（2）XYZの座標値がそれぞれ、2, 3, 0となるように、3つ目の球を作ってください。

ランダムな位置（X）に球を作る

```
1 using UnityEngine;
2 using System.Collections;
3
4 public class mastersc : MonoBehaviour {
5
6     public GameObject spf;
7
8     // Use this for initialization
9     void Start () {
10        float x;
11        x = Random.Range (-5f, 5f);
12        Instantiate (spf, new Vector3 (x, 5f, 0f), Quaternion.identity);
13    }
14
15    // Update is called once per frame
16    void Update () {
17
18    }
19 }
20
```

赤線の部分を追加してください。

解説

```
float x;
```

数字を入れる箱（変数）をxという名前で作っています。
floatは、1.3や2.5など小数点のつく変数を作るときに使います。

```
x = Random.Range (-5f, 5f);
```

Random.Range（ランダム・レンジ）のランダムは乱数、レンジは範囲です。
この場合、-5～5の間で乱数を取得します。
取得した乱数は、さきほど作った変数xに入ります。これを代入といいます。
代入は、式の右側から左側へ行われます。

```
new Vector3 (x, 5f, 0f),
```

球を作る位置の指定で、X座標に対応する部分に、変数xを入れます。
変数xは箱ですから、中に入っている数値（-5～5の間の小数値）がX座標となります。

再生してみてください。球が横方向（X座標の方向）の適当な位置に現れます。
再生を止めてふたたび再生すると、また別の位置に現れますね。

球を2つ作る

```
// Use this for initialization
void Start () {
    float x;
    x = Random.Range (-5f, 5f);
    Instantiate (spf, new Vector3 (x, 5f, 0f), Quaternion.identity);
}
```

図の赤い矢印のところへカーソル（入力位置を示す縦棒）を移動します。

```
// Use this for initialization
void Start () {
    float x;
    x = Random.Range (-5f, 5f);
    Instantiate (spf, new Vector3 (x, 5f, 0f), Quaternion.identity);
}
```

キーボードで、shiftキーを左手の指で押しながら、右向き矢印キーを右手の指で押し続けます。上のような範囲が青くなるまで押してください。これで2行分が選択状態になります。
この状態で、commandキーを押しながらCを押します（command+C）。これがコピーです。

```
// Use this for initialization
void Start () {
    float x;
    x = Random.Range (-5f, 5f);
    Instantiate (spf, new Vector3 (x, 5f, 0f), Quaternion.identity);
}
```

図の赤い矢印のところへカーソル（入力位置を示す縦棒）を移動します。
移動したら、Enterキーを押します。
そのあと、commandキーを押しながらVを押します（command+V）。これがペースト（貼り付け）です。

```
// Use this for initialization
void Start () {
    float x;
    x = Random.Range (-5f, 5f);
    Instantiate (spf, new Vector3 (x, 5f, 0f), Quaternion.identity);
    x = Random.Range (-5f, 5f);
    Instantiate (spf, new Vector3 (x, 5f, 0f), Quaternion.identity);
}
```

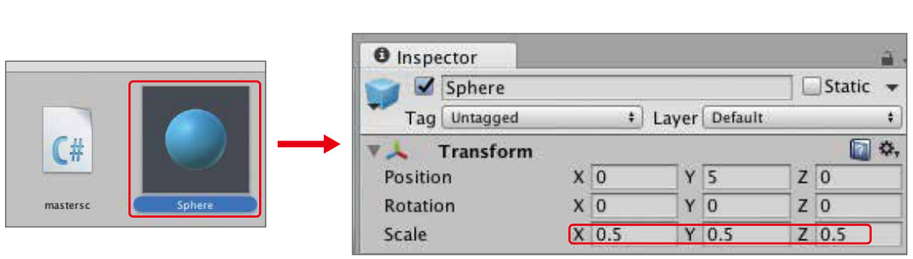
上のようになったら、成功です。
コピー・アンド・ペーストというやり方です。
再生してみましょう。2つの球が出てきたら成功です。

球を4つ作る

```
// Use this for initialization
void Start () {
    float x;
    x = Random.Range (-5f, 5f);
    Instantiate (spf, new Vector3 (x, 5f, 0f), Quaternion.identity);
    x = Random.Range (-5f, 5f);
    Instantiate (spf, new Vector3 (x, 5f, 0f), Quaternion.identity);
    x = Random.Range (-5f, 5f);
    Instantiate (spf, new Vector3 (x, 5f, 0f), Quaternion.identity);
    x = Random.Range (-5f, 5f);
    Instantiate (spf, new Vector3 (x, 5f, 0f), Quaternion.identity);

}
```

コピー・アンド・ペーストを使って、さらに4行を増やしてください。
再生してみましょう。４つの球が出てきたら成功です。4つも作ると、球がぶつかることが多くなりますね。
そこで、Assetsの中にプレハブ化したSphere（スフィア）をクリックして、Transform(トランスフォーム)のScale（スケール：大きさ）を0.5にします。



再生してみましょう。小さな球が４つ出てきたら成功です。ぶつかる場合もあります。
でも、このプログラムには、同じ処理が何度も出てきます。4回くらいならいいのですが、いくつも球を作る場合は大変です。

そこで、for文を使います。

同じ処理を繰り返して球をたくさん作る

```
// Use this for initialization
void Start () {
    float x;
    for (int i = 0; i < 10; i++) {
        x = Random.Range (-5f, 5f);
        Instantiate (spf, new Vector3 (x, 5f, 0f), Quaternion.identity);
    }

}
```

for文は、緑枠で示した中かっこの中の処理を繰り返します。

```
int i = 0;
```

変数iを作って0を代入します。int（イント）は整数の箱を作るときに使います。

```
i < 10;
```

変数iが10より小さい間は、forの処理が繰り返されます。

```
i++
```

繰り返されるたびに、変数iに1を足します。

変数iは最初は0で始まり、1つ目の球が作られます。
次に、変数iは1だけ増えて、変数iが1になり、2つ目の球が作られます。
変数iが2になり、3つ目の球。
変数iが3になり、4つ目の球。
・・・
変数iが9になり10個目の球が作られます。

再生してみましょう。10個の球が出てきたら成功です。球と球がぶつかって不思議な動きになるときもありますね。

球の出てくる高さ(Y)をランダムにする

```
// Use this for initialization
void Start () {
    float x;
    float y;
    for (int i = 0; i < 10; i++) {
        x = Random.Range (-5f, 5f);
        y = Random.Range (3f, 6f);
        Instantiate (spf, new Vector3 (x, y, 0f), Quaternion.identity);
    }

}
```

masterscを開き、上のように編集します。

再生してみましょう。球の出てくる高さが変わりましたか？

球の出てくる奥行(Z)をランダムにする

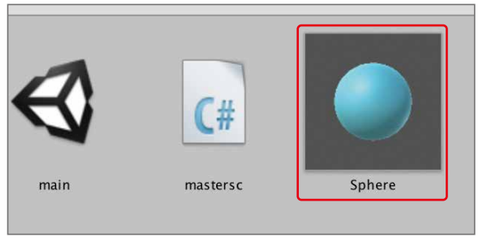
```
// Use this for initialization
void Start () {
    float x;
    float y;
    float z;
    for (int i = 0; i < 10; i++) {
        x = Random.Range (-5f, 5f);
        y = Random.Range (3f, 6f);
        z = Random.Range (-5f, 5f);
        Instantiate (spf, new Vector3 (x, y, z), Quaternion.identity);
    }

}
```

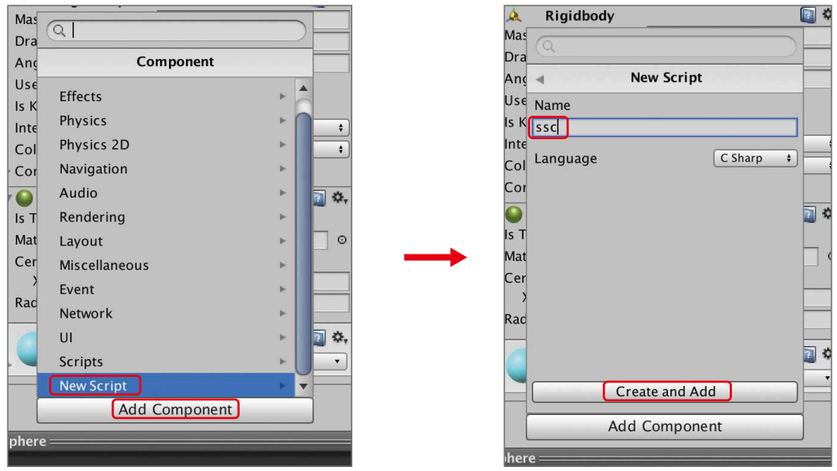
masterscを開き、上のように編集します。

再生してみましょう。球の出てくる奥行が変わりましたか？

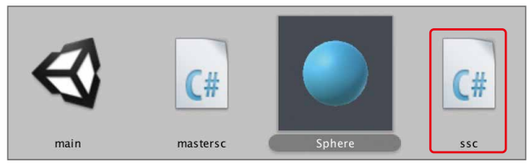
球にスクリプトを付ける



Assets（アセット）の中にあるプレハブのSphereをクリックします。



インスペクターで、Add　Component（アッドコンポーネント：構成要素の追加）をクリックし、New Script（ニュー・スクリプト：新しいスクリプト）をクリックします。Name（ネーム：名前）を、sscとして、Create and Add（クリエイト・アンド・アッド：生成して追加）をクリックします。



Assets（アセット）の中に作られたスクリプトsscをダブルクリックして開きます。

条件成立で球を消す

```
// Update is called once per frame
void Update () {

    if (transform.position.y < -1f) {
        Destroy (gameObject);
    }

}
```

解説
ifは、条件によって処理を行ったり行わなかったりする処理（分岐処理）です。
transform.position.yは、球のY座標の値です。
小かっこの中は、このY座標が-1よりも小さかったら、という条件です。
条件が成立した場合は、中かっこの中の処理を実行します。
中かっこの中、Destroy (gameObject);は、自分自信を消す処理です。
つまり、球のY座標が-1よりも小さくなった場合、球（自分）を消します。

```
// Update is called once per frame
void Update () {

    if (transform.position.y < -1f) {
        Destroy (gameObject);
    }

    if (transform.position.y > 8f) {
        Destroy (gameObject);
    }

}
```

もう一つif分を増やします。上のように編集してください。
ここでは、球（自分）のY座標が8よりも大きくなった場合、球（自分）を消します。

再生してみましょう。

memo