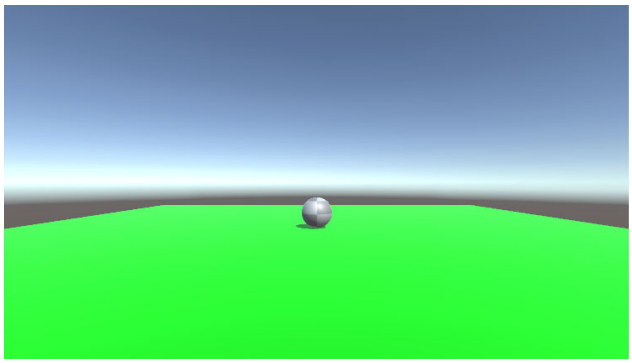
## ボールが転がる理屈について

unity for 2019.2.21

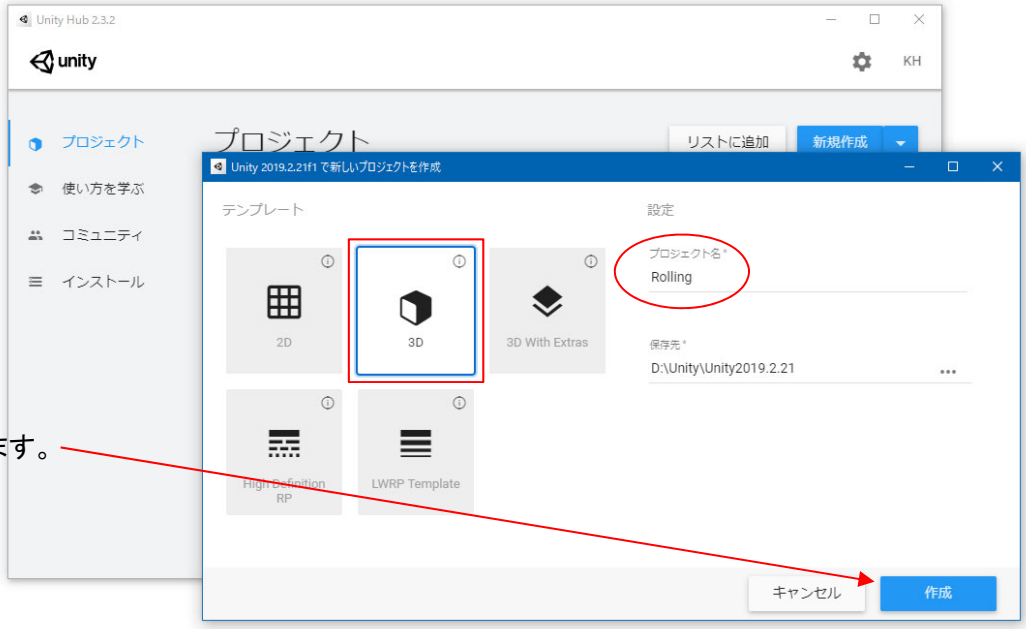
　ゲームエンジンに内蔵されている物理演算機能を用いて物体同士の反射・侵入を検出できるようになりました。

　更に物理挙動の演算機能を用いると、永遠に一定速度で動き続けたり、ボールをジャンプさせたり、転がしたりなどの操作原理に転用できます。



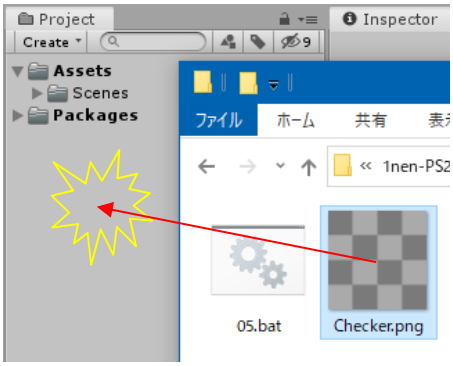
### 【STEP1】 プロジェクトの準備

● UnityHUB を起動し、プロジェクト Rolling を3D モードで準備します。




ボタン作成を押下します。

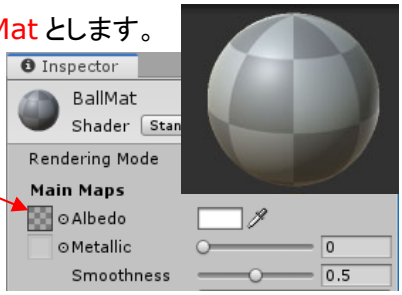
● 配布された素材を読み込みます。
エクスプローラーで Checker.png をプロジェクト欄にドラッグ &ドロップします。



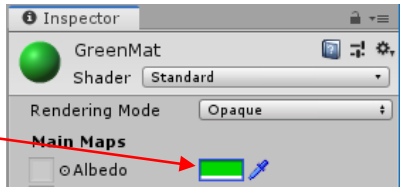

### 【STEP2】 マテリアルの作成

● プロジェクト欄の Create から Material（マテリアル）を選択し、名称を BallMat とします。

項目 Albedo（アルベド）に画像 Checker.png を指定します。

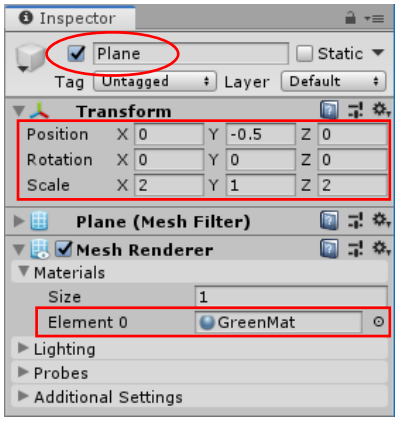


● 同様にして、新しいマテリアル GreenMat を作成します。
こちらは緑色だけを割り当てておきます。

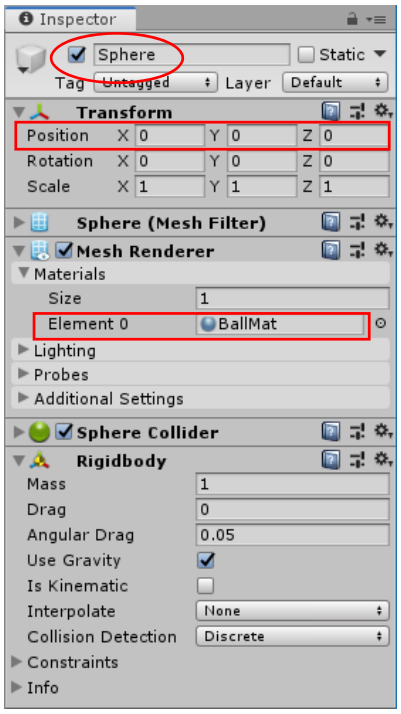



## 【STEP3】 シーンを準備する

● ヒエラルキー欄の Create（クリエイト）から 3D Object > Plane（プレーン）を選択します。
インスペクタでパラメータを設定します。

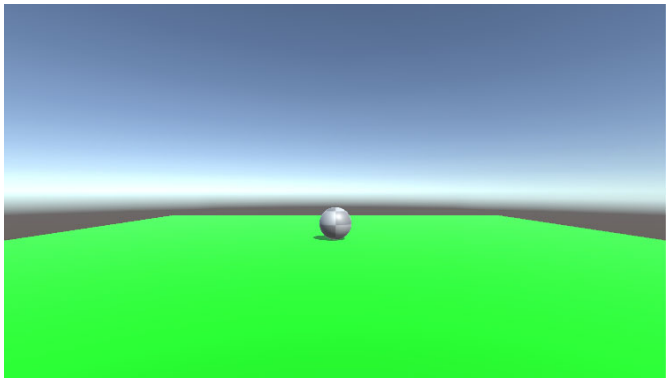



● 同様にして Create（クリエイト）から 3D Object > Sphere（スフィア）を選択します。
インスペクタでパラメータを設定します。

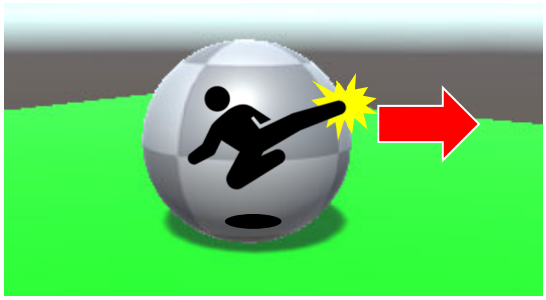



● インスペクタの Add Component から Physics > Rigidbody を選択します。

こんな感じになります。

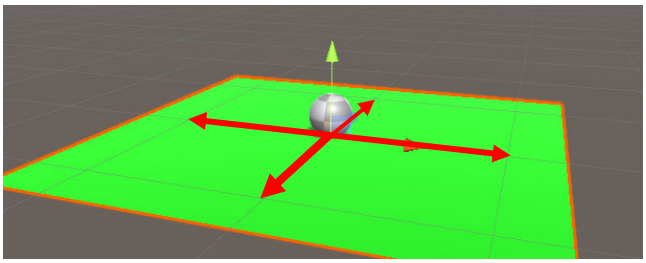


## 【STEP4】 手法①:ボールが自身で転がる



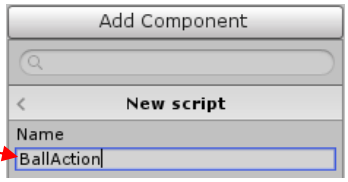
ボールに内蔵されているリジッドボディは物理演算が得意なので、キーボードの上下左右キーで作った方向に向かって、ボール内部でキックのような力を加え続けることが出来ます。

その結果、ボールがどんどん加速し、転がる理屈となります。



キーボードの方向キーからは、左右(－1～1)と、上下(－1～1)の情報が取得できます。その2つの値を用いて、平面的なベクトル Dir(赤い矢印方向)を作成し、ボールの物理演算機能 Rigidbody に AddForce します。

● ヒエラルキー欄の Sphere を選択し、インスペクタの Add Component から New script を選択し、名称を BallAction とします。



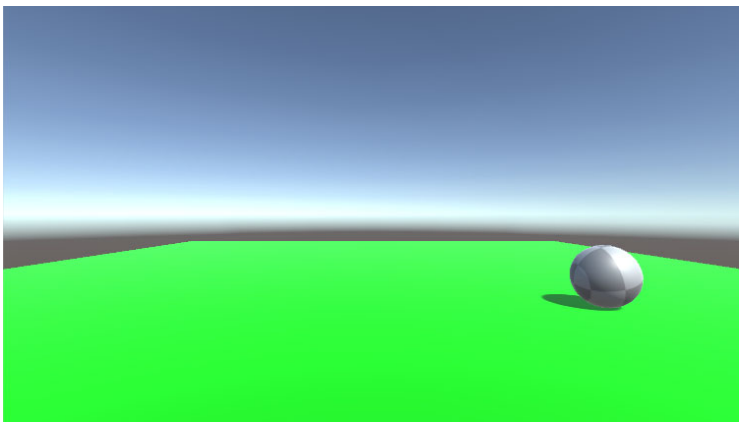

● プログラム BallAction を次のように修正します。

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BallAction : MonoBehaviour {

    Rigidbody MyRB; //自身のRigidbody

    void Start() {
        MyRB = GetComponent<Rigidbody>(); //自身のRigidbodyを取得
    }

    void Update() {
        float h = Input.GetAxis( "Horizontal" ); //左右キー方向の値(-1～0～1)
        float v = Input.GetAxis( "Vertical" ); //上下キー方向の値(-1～0～1)
        Vector3 Dir = new Vector3( h, 0, v ); //方向ベクトルを作る
        MyRB.AddForce( Dir * 4 ); //力を加えている
    }
}
```

1 秒間に100回の頻度でキーの状態を取得している。

● プレイボタンを押下します。

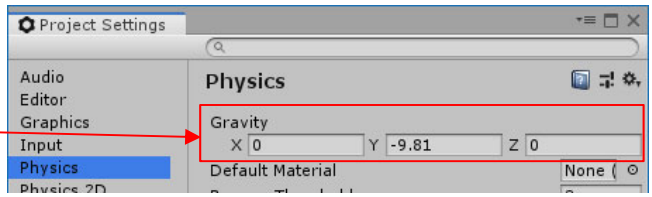
キーボードの上下左右キーを押下すると、その方向にボールは転がる動きをします。



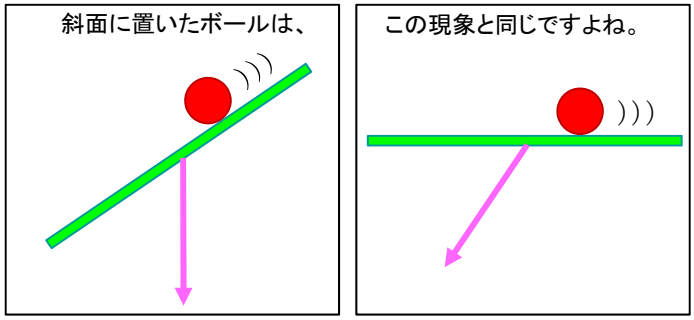
## 【STEP5】 手法②:引力の方向を変える

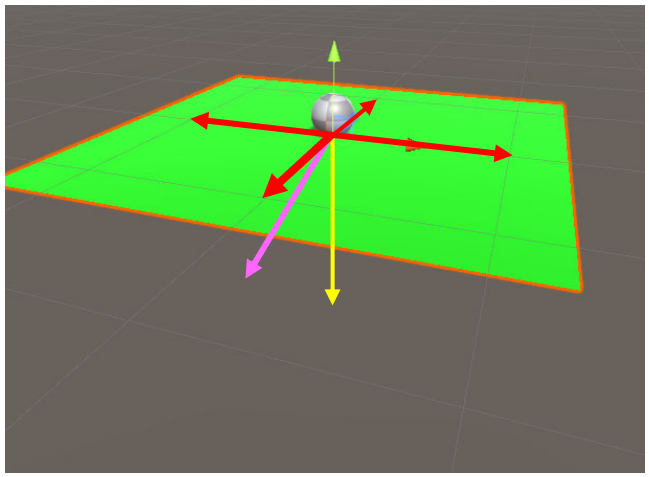
Unity での物理計算エンジンの設定項目には、重力方向のベクトル(向きと強さ)が定義されています。

規定値では黄色いベクトル(方向)です。

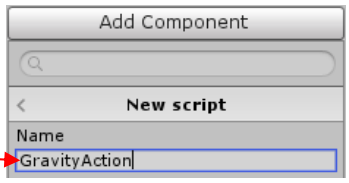



このベクトル(方向)に対して、キーボードの上下左右キーで赤い方向へのベクトルを作り、合成することによって新しい重力方向(ピンク)を作り出す考え方となります。



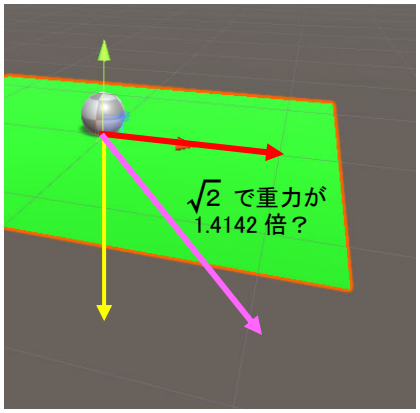
- カメラの仕事ではありませんが、代表者としてヒエラルキー欄の Main Camera を選択し、インスペクタの Add Component から New script を選択します。
  名称を GravityAction とします。



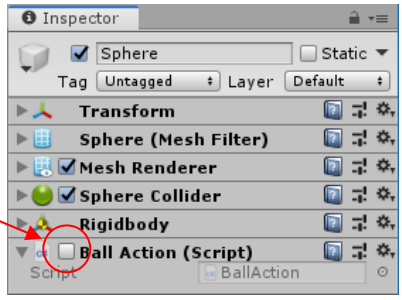

- スクリプト GravityAction を次のように編集します。
  キーボードを操作して作った方向Gの 9.8 倍を、新しい重力とする。

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class GravityAction : MonoBehaviour {

    void Start() {

    }

    void Update() {
        Vector3 G = new Vector3( 0, -1, 0 ); //鉛直下方向ベクトルを作る
        G.x = Input.GetAxis( "Horizontal" ); //左右キー方向の値(-1～0～1)で修正
        G.z = Input.GetAxis( "Vertical" ); //上下キー方向の値(-1～0～1)で修正
        Physics.gravity = 9.81f * G.normalized; //物理演算の重力方向に渡す
    }
}
```

一旦 normalized をするのは、鉛直下方向のベクトルの長さが1なのに、キーボードの情報を加えると1より長くなって重力が強くなってしまう為で、方向は維持しつつも、長さを1にするために調整が必要になります。



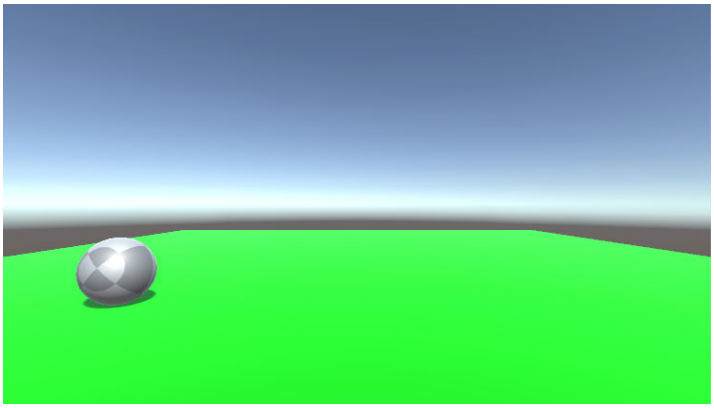
● 先の【STEP4】のスクリプトを停めます。ヒエラルキー欄の Sphere を選択し、インスペクタで BallAction をオフにします。




● プレイボタンを押下します。 

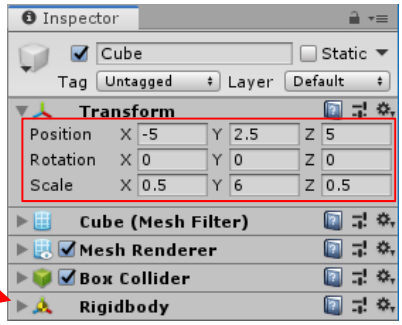
キーボードの上下左右キーを操作して、ボールを操作できることを確認します。

先の例とは、少し転がる様子が違いますが、ボールを転がすゲームの原理としては、こちらの実現方法も有効です。

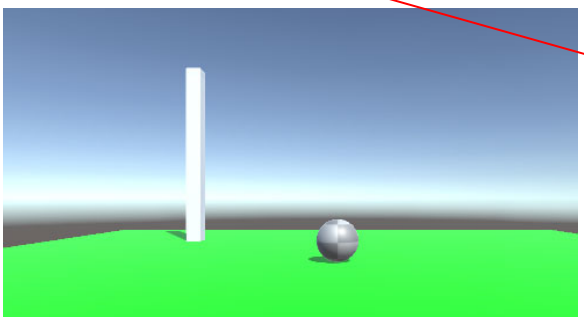


また、ボール自身がキーボードを取得して動いているのではないので、他の物理挙動物体を置いても、スクリプト無しでスグに重力の影響を受けます。シーン内の全物体が正しく重力の影響を受けるので、好都合です。

ただ、ボーリングのような場合、ボールを操作して進み始めると、重力の影響で標的が倒れてしまいます。その場合は**【STEP4】**の方法になるでしょう。




● 試しに、細い直方体を準備して、Rigidbody を取り付けてみます。
プレイした途端に倒れることが判ります。



2通りしかない訳ではありませんが、作るゲームの内容によって、どちらの実現方法が適しているのか？を見極めることが重要です。

今回の原理を利用すると、iPad などのスマートデバイスをターゲットとして、実機本体を傾けた操作でボール転がしゲームを実現できます。考え方としては、キーボードの上下左右キー操作を、実機から取得した情報に置き換える部分を解決すれば可能になります。

以上