Unity 講習資料 1

0．はじめに

Unity 部門の講習を始めます。これから皆さんには、Unity というソフトを使ってゲームを作ってもらいます。目標は、文化祭で自作のゲームを一本公開することです。さて、この講習資料についてですが、実際に Unity を動かしながら操作やコマンド、用語などを覚えてもらうことを目的に書いています。が、まあ完璧に覚えるというのは無理なので（自分も覚えてないです)、この資料を片手にゲームを作ってもらえればと思います。また、今はまだわからなくてもいい、本筋からそれる所は「飛ばしてください」と書いてあります。そこは飛ばして、あとで戻ってきたり家で読んだりしてください。

※資料は Unity がインストールされていることを前提としています。

1．Unity をはじめる

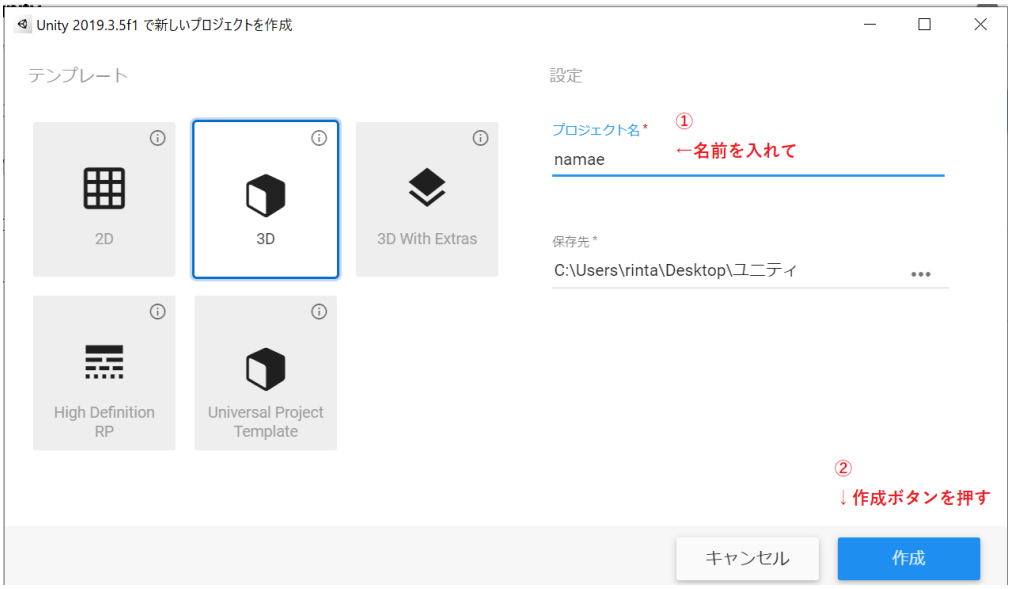
Unity は、世界中のゲームクリエイターが使っているゲーム開発ツールです。「これを使えば簡単にいろいろなゲームを作れる」と思ってくれて構いません。無料で使用できます。さっそく始めてみましょう。

① デスクトップ上の「Unity Hub」のアイコンをダブルクリックして開いてください。するとこんな画面が出ると思います。

Unity Hub は Unity 本体の管理をするソフトです。ここから Unity をインストールしたり、プロジェクトを作ったりできます。

プロジェクト：作るゲームそのもののデータをまとめたもの。

② 新規作成ボタンを押してプロジェクト名に好きな名前を入れて（わかりやすいほうがいいです)、作成ボタンをクリックしてください。しばらく待つと Unity 本体が起動します。



③　上の画面の説明をします。飛ばしてください。

（0）　そもそもこの画面は

Unity Hub で新しくプロジェクトを作るとき、最初に設定ができる画面です。

（1）　テンプレート

２Ｄゲームか３Ｄゲームか、あるいはその設定はどうかを選べます。

２Ｄ：２Ｄゲームに適した設定のプロジェクトを作ります。

３Ｄ：３Ｄゲームに適した設定のプロジェクトを作ります。

３DWithExtras：PostProcessing なる機能を使える３Ｄのプロジェクトを作ります。使うとグラフィックがきれいにできるようです。

HighDefinitionRP：とてもきれいなグラフィックの３Ｄのプロジェクトを作ります。HDRP とか ShaderGraph とかいう機能が使えるようです。

UniversalProjectTemplate：いろんなところで使える３Ｄのプロジェクトを作ります。設定が簡単に変更できるらしいです。

このうち下の三つは、パソコンに負荷がかかりすぎる上、そこまでハイレベルなものを使う必要がないのでやめたほうがいいです。

ということで、２Ｄか３Ｄを選びましょう。

（2）　プロジェクト名

プロジェクトの名前を付けられます。今後も何度か書くつもりですが、名前はわかりやすいものを付けたほうがいいです。「aaa」とかじゃ後になって絶対に「なんだこれ」ってなります。

（3） 保存先

プロジェクトは当然 PC のどこかにデータの形で保存されるのですが、その保存先を決められます。「…」ボタンから保存したい先のフォルダを選びます。部の PC を使っている間はいじらないでください。

（4） プロジェクトの作成

右下の作成ボタンを押すと、いままでした設定に従ってプロジェクトが作られます。そしてしばらく待つと（ここではプロジェクトの構造の作成やプロジェクトの呼び出しが行われています）Unity 本体が新しいプロジェクトを開いて起動します。

飛ばしてください、の部分終わりです。

2．シーンとオブジェクトとコンポーネント

Unity を使う上で重要な概念、シーンとオブジェクト、コンポーネントについて説明します。この先何度も使う用語なので、覚えてください。

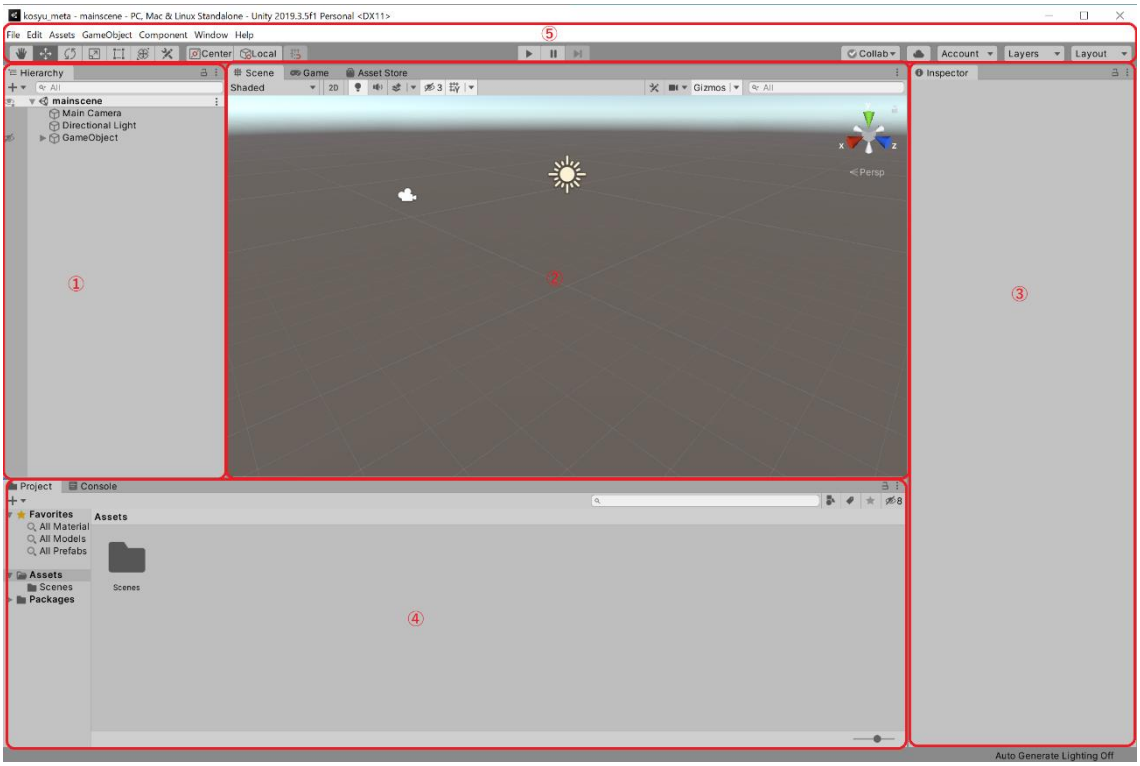
シーン：Unity が作り出した仮想空間のことです。日本語にすれば「舞台」「場面」。このシーンにオブジェクトを配置することで、ゲームが成り立ちます。また、シーンはプロジェクト内で複数制作でき、それぞれを転換することでゲームを作っていきます（タイトル画面、ホーム画面、プレイ画面などを別々のシーンで作り、転換していきます)。

オブジェクト：「もの」のことです。しかし、それ自体では何も為さず（目には見えませんし、形もありません)、「コンポーネント」という情報、機能がつくことで、はじめて役目をもちます。コンポーネントのくっつく場所、というのがより正確です。

コンポーネント：オブジェクトにつく情報や機能のことです。オブジェクトはそれだけでは目には見えないし、動いたりしないし、何かにぶつかることもありません。しかし、そのオブジェクトにコンポーネントを付けると、目に見えるようになり、動き出し、何かにぶつかるようになります。コンポーネントは Unity に初めから設定されているものの他、自分で書いたプログラムもコンポーネントとして扱われます。

3．画面の説明

Unity の画面の説明をします。



① ヒエラルキー

シーンにあるオブジェクトを表示します。

② シーン・ゲームビュー

シーンビューとゲームビューの二つを見れます。

シーンビューではシーンの状況を確認できます。

ゲームビューでは、レンダリング（リアルに表示すること）後のシーンを確認できます。指定のカメラから見た景色が映されます。

シーンビュー、ゲームビューの切り替えは左上のタブでできますが、実行および実行停止時に自動で切り替わるので、必要はないでしょう。

③ インスペクター

選択しているオブジェクトについているコンポーネントを確認、編集できます。

④ プロジェクト・コンソールビュー

プロジェクトビューとコンソールビューの二つを見れます。

プロジェクトビューでは外部ファイルなどの操作ができます。プロジェクトのフォルダ内の Asset フォルダと同期しています。

コンソールビューではバグの表示やデバッグログの表示を行います。

左上のタブで変更できます。

⑤ ツールバー

いろんなことができます。

4．オブジェクトの作成

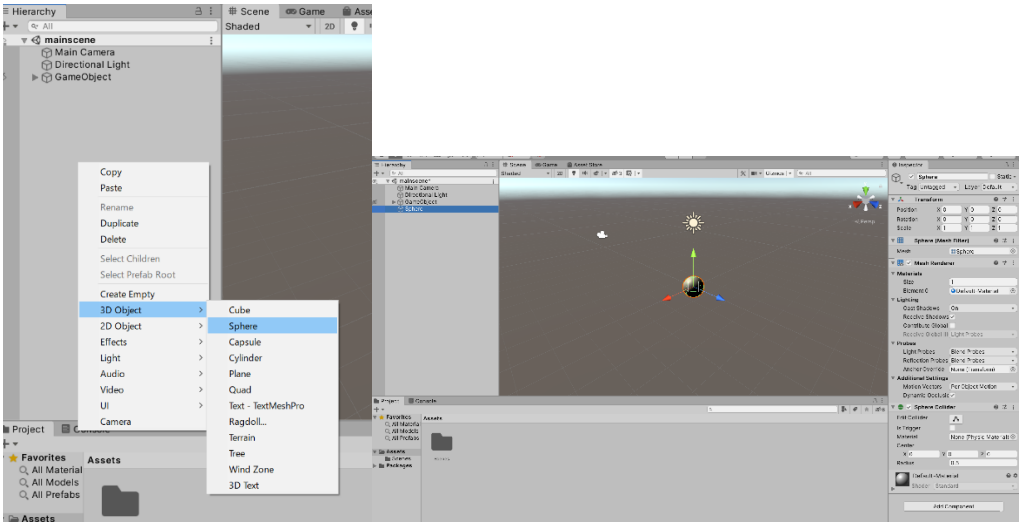
⓪　シーンには作成された時点で既にいくつかのオブジェクトが存在します。簡単にその説明をします。

Main Camera：カメラです。ここから見える景色が最終的なゲームの視点になります。

Directional Light：光源です。Unity は光の当たり方なども計算してくれます。ないと画面が暗くなります。

オブジェクトを作ってみましょう。

①　ヒエラルキー上のどこかを右クリック or 左上の create をクリックし、3D Object>Sphere を選択します。するとシーンに球が作られます。インスペクターにも作った球のコンポーネントが表示されます。



このようにすると、オブジェクトを作れます。

②　作れるものの説明をします。飛ばしてください。

CreateEmpty：実体のないオブジェクトを作ります。ゲームの進行管理などに使います。

Cube：立方体を作ります。

Sphere：球を作ります。

Capsule：カプセル（昔の錠剤みたいな形）を作ります。

Cylinder：円柱を作ります。

Plate：厚さ 0 の直方体を作ります。床や壁に使用します。

Quad：厚さ 0 の目に見えない直方体を作ります。画像を張り付けて使用します。

Text,Regdoll,Terrain,Tree,WindZone,3DText：難易度が高かったりアセットストアの使用を前提としているのでやりません。気になった人は自分で調べて挑戦し

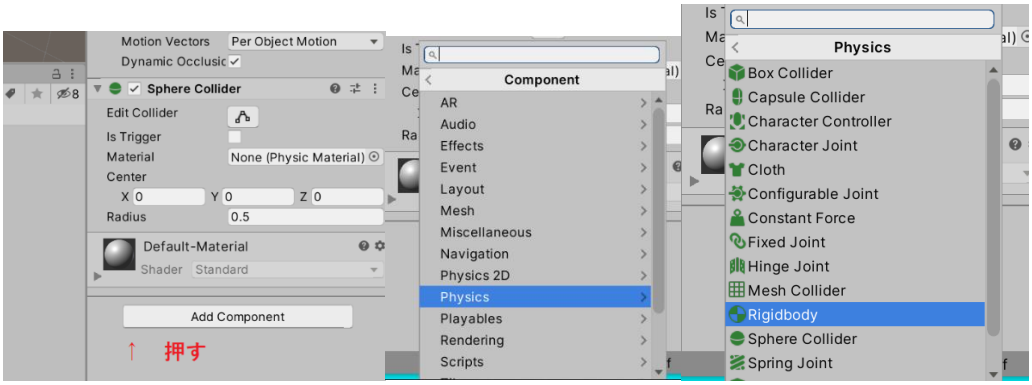
てみてください。
２D Object：２D のオブジェクトを作ります。そのうちやります。
Effect：エフェクトを作ったり制御できます。扱う予定はありませんがおもしろいのでやってみることをおすすめします。
Light：光源を作れます。
Audio：音声にかかわるものを作ったりできます。
UI：UI（入力や表示のこと）に関するものを作れます。（ボタンなど）
Camera：カメラ（視点）を作ります。
飛ばしてください、の部分終わりです。

5．コンポーネントを付ける

⓪　オブジェクトは作った時点ですでにいくつかのコンポーネントがついています。簡単にその説明をします。
Transform：位置の情報。あとでやります。
Sphere(Mesh Filter),Mesh Renderer：どちらもオブジェクトの描画に関するものです。
Sphere Collider：当たり判定です。
Default-Material：オブジェクトの材質や色の情報です。

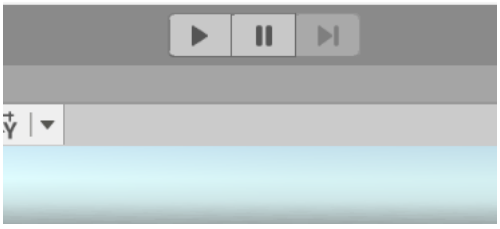
球に物理演算をさせるコンポーネントをつけて、重力をかけてみましょう。

①　球を選択した状態で（シーンの球をクリックすると選択できます）インスペクターの一番下にある Add Component ボタンを押し、Physics＞Rigidbody を選択します。これで球に Rigidbody コンポーネントが追加されました。

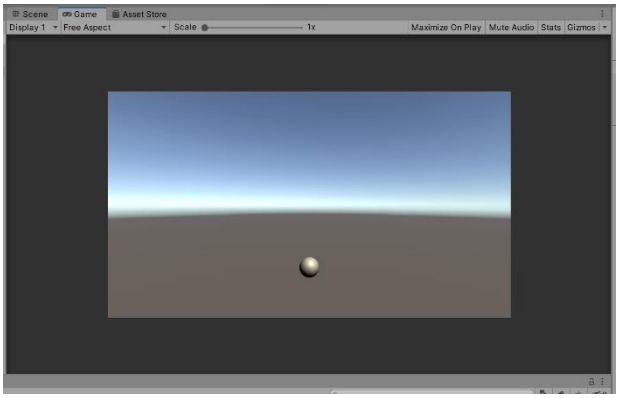


Rigidbody 以外の様々なコンポーネントも Add Component ボタンから追加できます。
Rigidbody：これがついたオブジェクトに物理演算をさせるということを意味するコンポーネント。ここでいう物理演算とは、力を受けた時の挙動のほか、重力を含みます。

② シーン・ゲームビューの上の再生ボタン（▶）をクリックしてください。



すると視点が変わり球が落ちるのが見えたと思います。

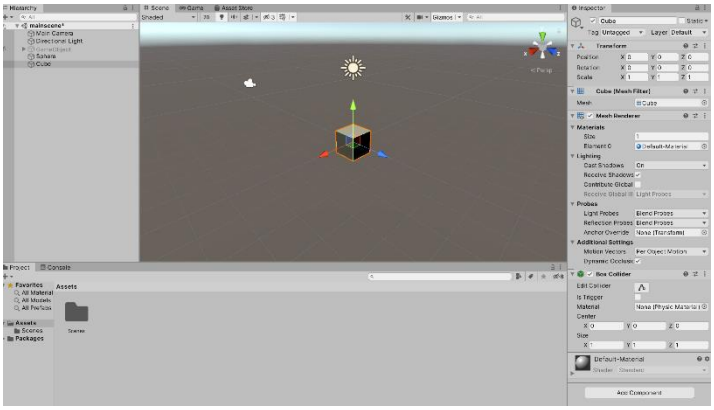


③ もう一度再生ボタンをクリックし、もとの画面に戻りましょう。球ももとの位置に戻ります。
再生ボタン：押すとゲームが実行されます。つまり、入力を受け付けたり、物理演算を開始したりします。

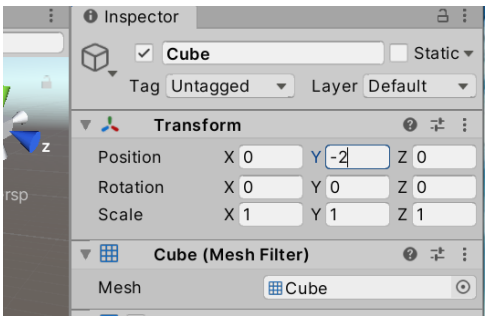
6．足場を作る

球が落ちないように足場を作ります。

① ヒエラルキー上で右クリックし、3D Object > Cube を選択します。すると球があった位置に立方体が出現します。重なっているだけで、球は消えていません。



② インスペクターの上側の Transform の Position の Y を、０から－２にします。欄をクリックしてキーボードで入力してください。



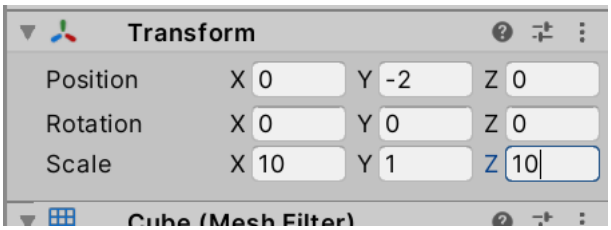
すると立方体の位置が変わりました。コンポーネントの数値もインスペクターから編集することができます。

Transform：オブジェクトの位置や大きさの情報を表すコンポーネント。

Position：Transform コンポーネントのもつ情報の一つで、位置の情報。座標空間上での座標を表す。

座標空間：座標平面（数学でやります）の立体版。ある一点を基準とし、そこから前、横、上方向にそれぞれどれだけ離れているかを表す。

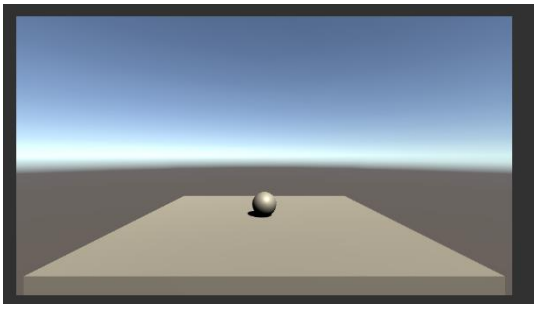
③ このままだと狭いので大きくします。Transform の Scale の X,Z をどちらも１０にしてください。



立方体が縦横に大きくなりました。

Scale：Transform のもつ情報の一つで、大きさの情報。

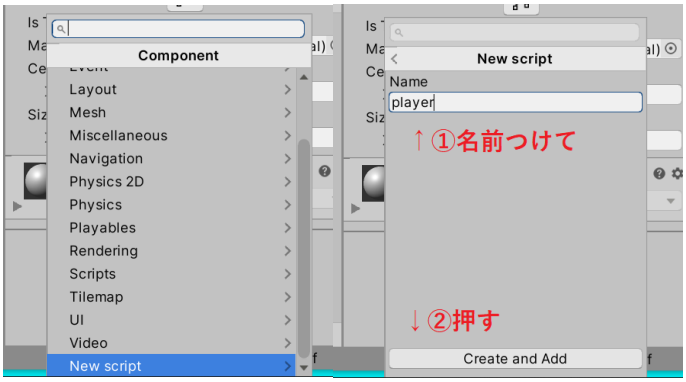
④ 再生ボタンを押して再生してみましょう。球が足場に落ちて止まったのがわかります。



7．動かせるようにする

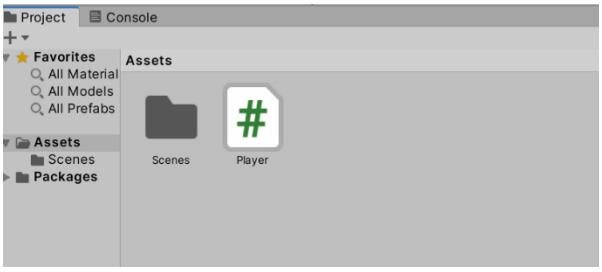
球を自分で動かせるようにします。

① シーン上の球をクリックし、インスペクターから Add Component > New Script を選択し、名前を Player とつけ、create and add を押します。

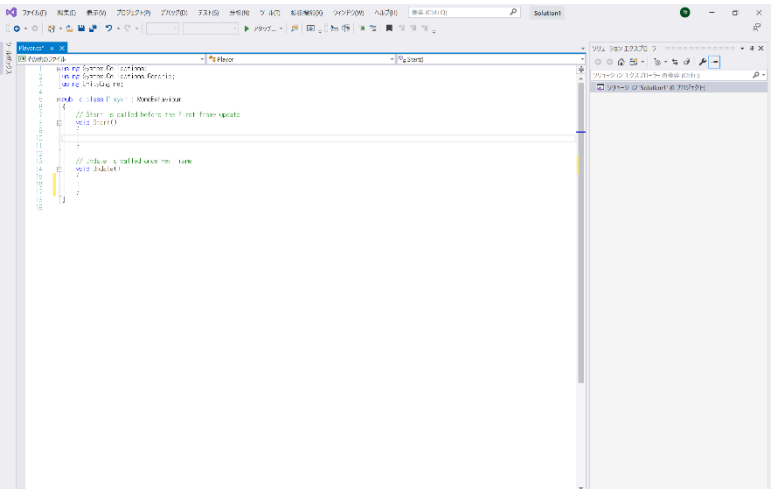


スクリプト：プログラムのこと。よくわからない英語を書いてなんかが動く、という一般的なイメージのプログラムのこと。Unity ではスクリプトもコンポーネントのひとつとして扱われます。

② プロジェクトビュー（画面下）にできた Player と書かれたものをダブルクリックします。

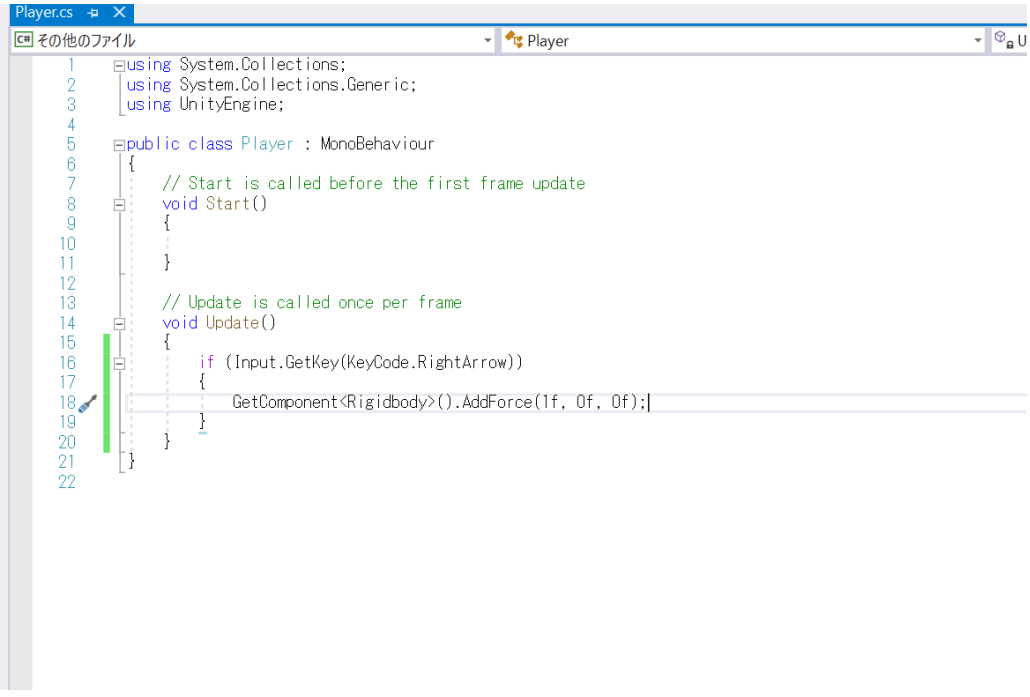


しばらく待つと「Visual Studio」もしくは「Visual Studio Code」という別のソフトが起動します。

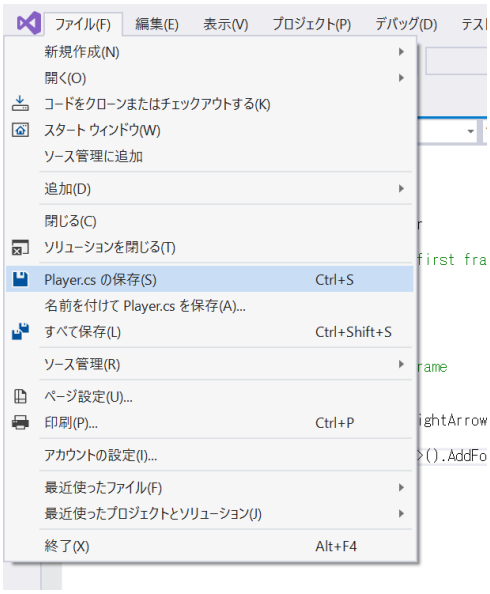


「Visual Studio」：スクリプトを書くソフト。スクリプトはしょせん文字なので、メモ帳でも書けるのですが、これにはプログラムを書くうえで便利な機能がついています。Visual Studio Code も同じです。

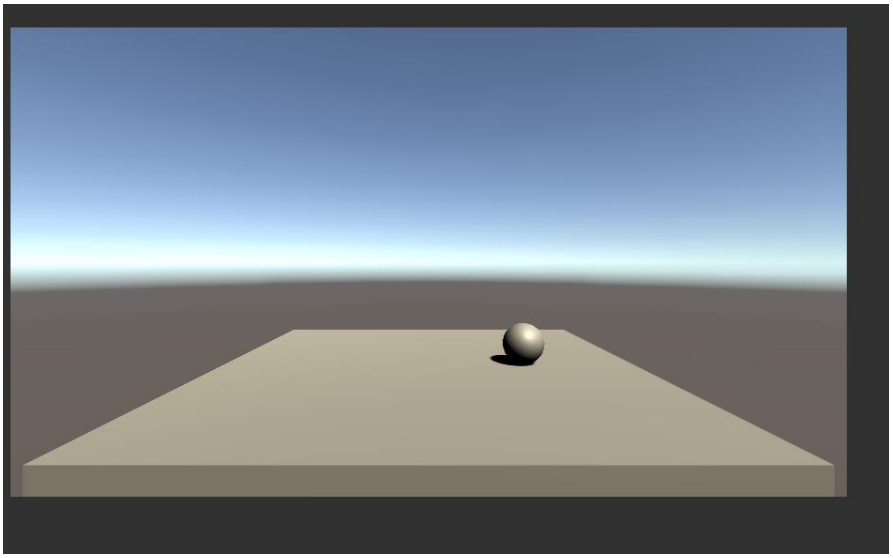
③ 以下のように打ち込んでみましょう。大文字と小文字の区別は正確に。



④ 右上にあるファイル（File）＞Player.cs の保存(save,セーブ)　を選択し、保存します。Ctrl キー＋S キーでも保存できます。保存しないと変更が反映されないので注意。



⑤ シーンを再生して、→キーを押してみましょう。ボールが右側に動いていきました。



8．７の解説

元から書いてあるやつ　こっちは流し読みでいいです。

１～３行目：Unity のコマンドを使用する、という定義。いじらないでください。

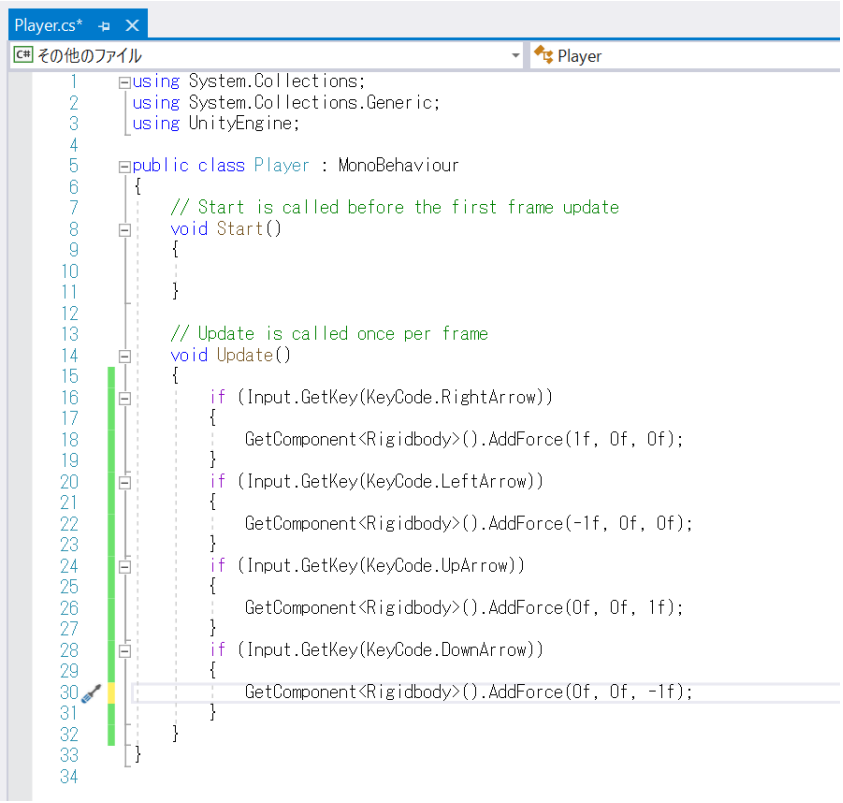
５行目：Player というクラス（わかんなくていいです）の宣言。この中に書いたことがシーンに反映されると思ってください。いじらないでください。

８～１１行目：このスクリプトが呼び出された最初の一回だけカッコ内の処理を実行する、という意味。

１４行目：毎フレームカッコ内の処理を実行する、という意味。
フレーム：ゲームの処理時間の単位。初期設定では１フレーム＝0.0166…秒であり、0.0166…秒に一回処理が行われます。
７、１３行目：コメントアウトされた説明。「//」と打った後の文はプログラムに無視されます。これをコメントアウトといいます。これでプログラムの説明を行ったり、一時的に問題のあるプログラムを処理させなくしたりします。

書き足したやつ　こっちはちゃんと読んでください。
if(~){…}：もし～なら…する、の意味。一番よくつかうプログラムで、基本中の基本。～には真偽を判断できる文が書かれる。
Input.GetKey(KeyCode.~)：～キーが押されているかいないかを返すプログラム。この場合は RightArrow、つまり右矢印キーが押されているかどうか。
GetComponent<Rigidbody>().AddForce(x,y,z);：球に力をかけるプログラム。x,y,z はそれぞれ左右、上下、前後にどれだけ力をかけるかを表します。今回は 1f,0f,0f なので、右方向にのみ 1 だけ力がかかります。f についてはあとで説明します。また、文の最後にある「;」（セミコロン）は、命令の最後に必ずつけなければいけない記号です。
以上より、このプログラムは「もし→キーが押されているなら、球に右方向に力をかける」という意味になります。

９．もっと動かせるようにする
右以外にも動かせるようにします。
①　以下のように加筆して保存してください。コピー＆ペーストを使うと楽です。



② 再生ボタンを押すと、押したキーの方向に球が動きます。

１０．　シーンの保存

① このあたりでシーンを保存しましょう。保存はこまめに行わないと、PC が落ちたときに今までの作業が水の泡となってしまいます。今回は初めての保存なので、ファイルに名前を付けてもらいます。右上の File→Save As を選択し、mainscene と名付けて保存してください。

② 上書き保存の方法は、プログラムと同じように File→Save もしくは Ctrl＋S キーです。これからは保存の指示はしませんが、適宜保存をしてください。

第一回講習はこれで終わりです。