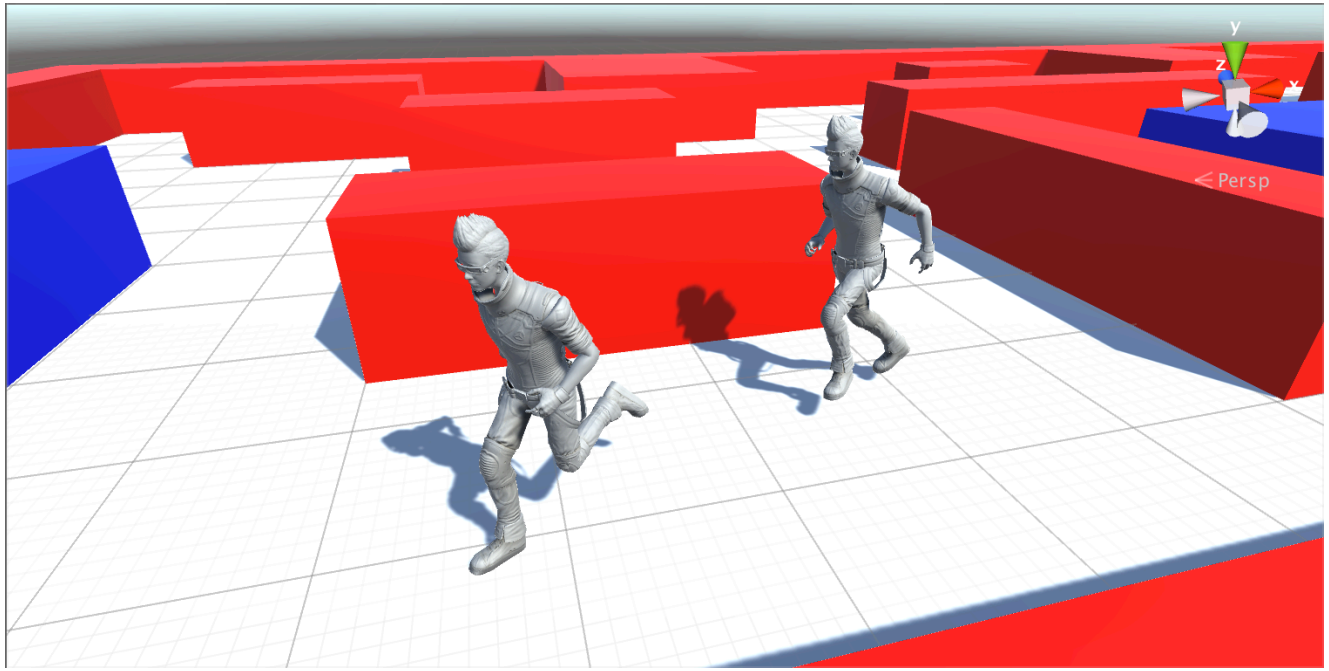
Unity講座1.1

~鬼ごっこ編~

はじめに



- 今回は、Unityに標準搭載されているナビゲーションという機能を使って、鬼ごっこをするゲームを作ります。
- この資料の通りに進めていくと、サングラスをかけたおじさん二人で鬼ごっこができるようになります。

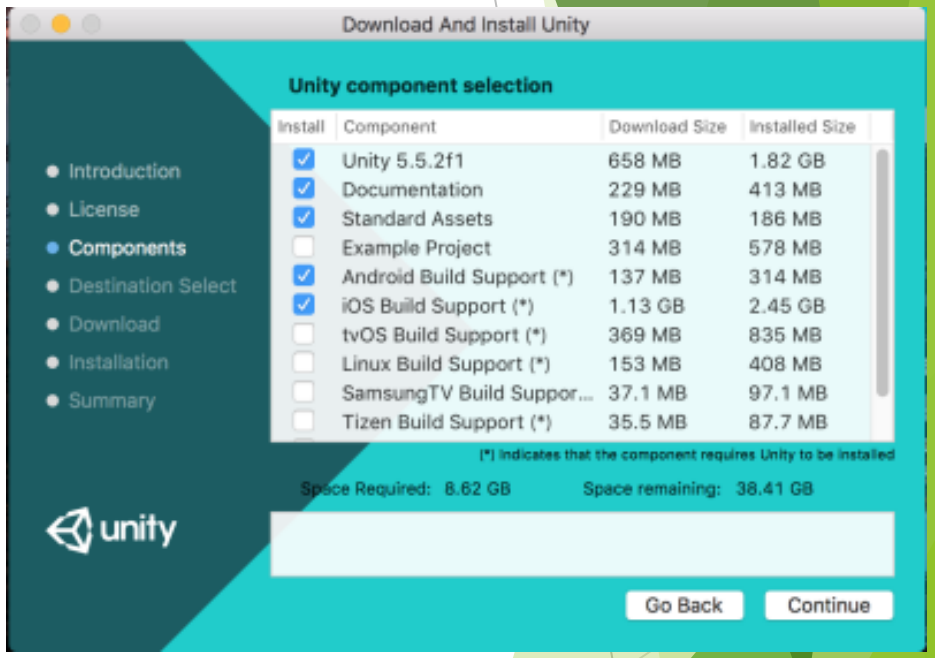
Unityの長所

- マルチプラットフォーム対応
  Android, iPhone, PCなど、対応する機種ごとに作り直す必要がない
- プログラミング無しでも動かせる
- 3Dモデルを使ったゲームが簡単に作れる
- 豊富な”アセット”
  他の人が公開している ゲームのパーツ を利用して、高度なことも簡単に

Unityをインストール

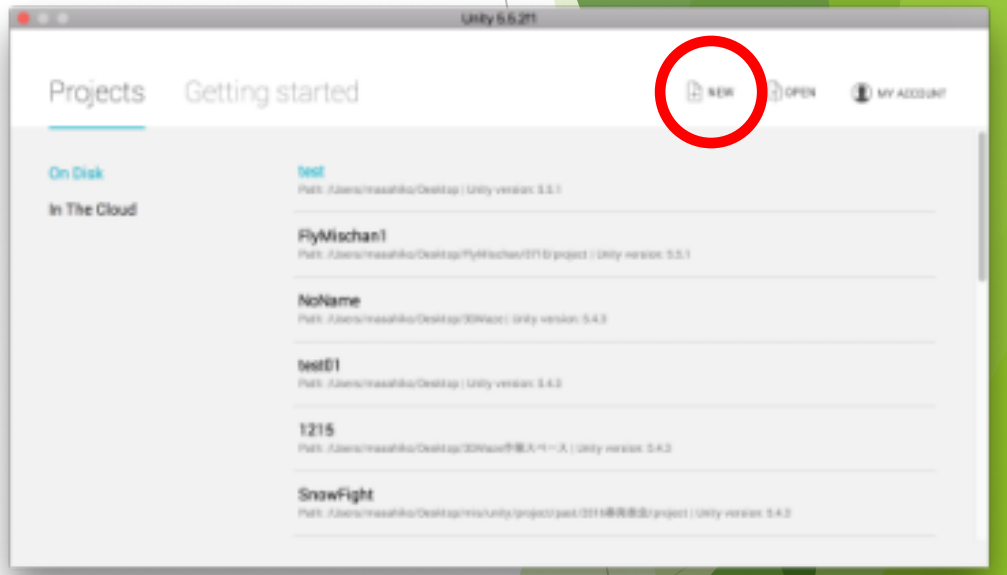
- 公式サイト(https://unity3d.com/jp/)から、Unityをダウンロードします。
- インストーラーを起動して、先へ進めていきます。

- 右図の画面でインストールするものを選択します。
  もともとチェックが入っているもの以外にも、iOSやAndroid, Mac, Windowsのビルドサポートはインストールしておきましょう。



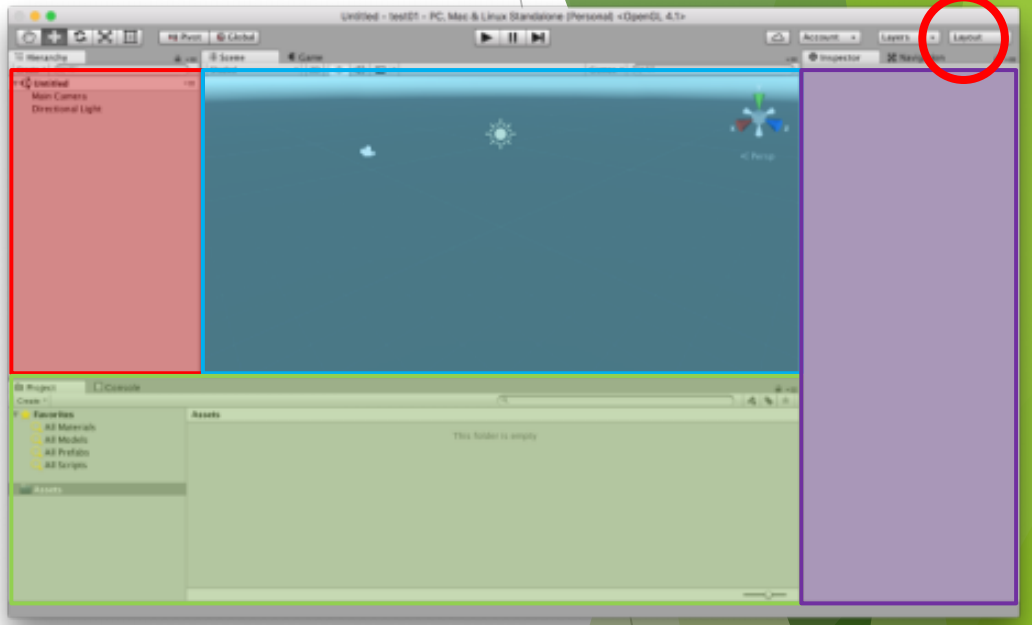
プロジェクトを作成

- Unityを起動します
- 必要事項を記入してアカウント登録
- 画面右上にあるNewボタンから新しいプロジェクトを作成します



画面の説明(1/2)

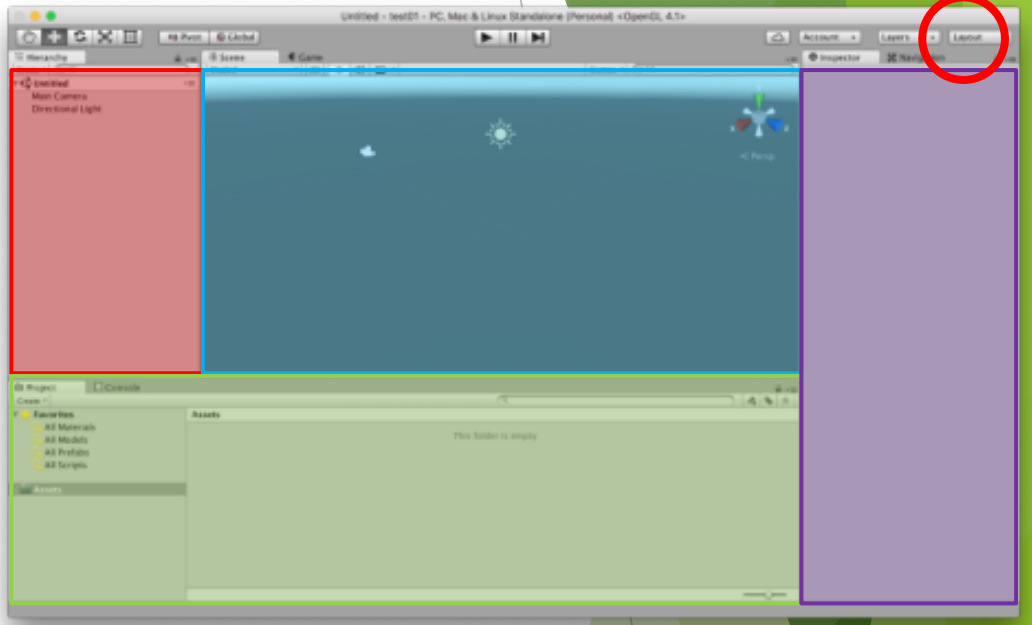
- ヒエラルキーウィンドウ
  シーン内にあるオブジェクトの一覧が表示されます
- シーンビュー
  ゲームの世界を自由に見て回れる画面。オブジェクトの移動や拡大縮小などの操作が行えます
- ゲームビュー
  ゲームのプレイ画面が表示される。シーンビューと同じ場所にあるが、オブジェクトの操作はできません



もし構成が違っても、右上のLayoutボタンからDefaultを選択すると呼び出せます。

画面の説明(2/2)

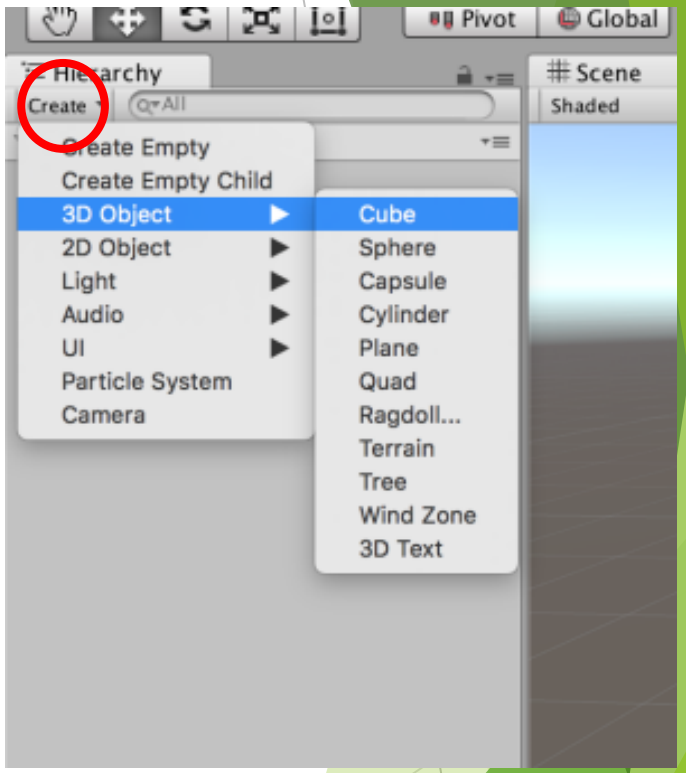
- プロジェクトウィンドウ
  　ゲーム全体の素材倉庫となるassetフォルダの中身が表示される。ここから素材を変更・追加することができます
- コンソールウィンドウ
  　ゲーム実行時のエラーなどが表示されます
- インスペクターウィンドウ
  　ヒエラルキーやプロジェクトウィンドウなどで選択したものの詳細が表示されます。ここから要素の追加や設定の変更の作業ができます



オブジェクトを置く

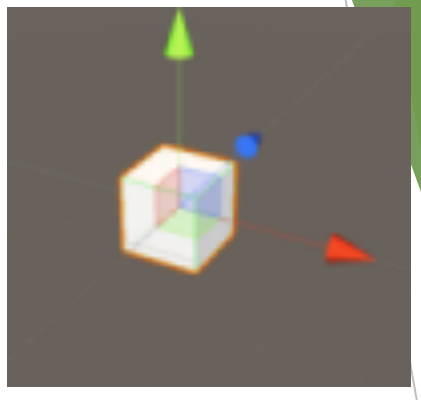
- 新しいシーンには、カメラとライトだけが置いてあります(2Dで作成した時はカメラのみ)。
このカメラによって映される映像が、プレイヤーが見るゲーム画面になります

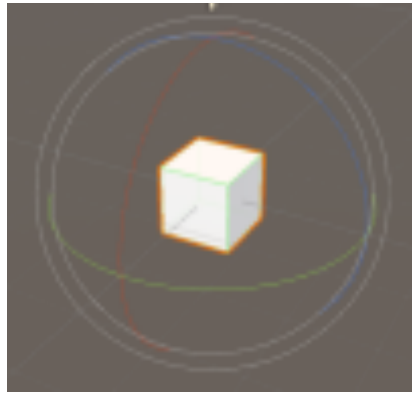
- 簡単な図形ならヒエラルキーウィンドウから呼び出せます
- 右図のように、画面左側、ヒエラルキーウィンドウのCreateボタンから3DObject→Cubeと選択して、新しいCubeを作成します。

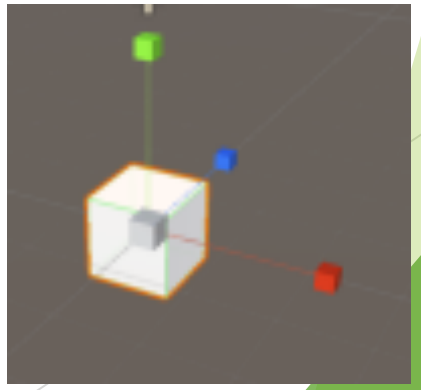


オブジェクトの操作

- 画面左上のツールバーで操作モードを切り替えられます
- オブジェクトの周辺に表示される軸(ギズモという)が変化します。この軸をドラッグすることで、移動・回転・拡大などの対応する操作が行えます
- Altキーを押しながらドラッグすると視点が回転します
- Altを押しながらマウスの右ボタンでドラッグすると視点のズームができます

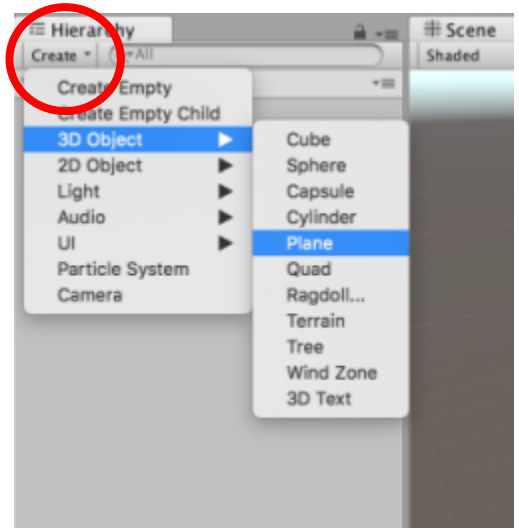


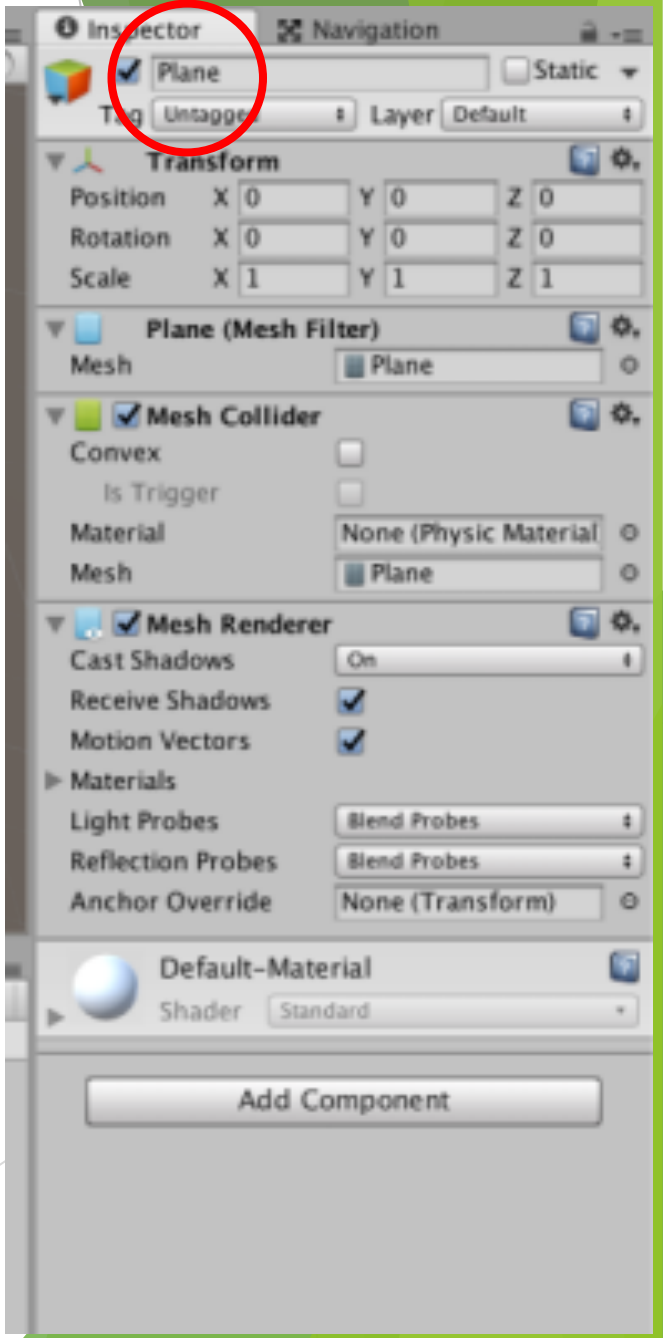




床の準備(1/2)

- 右図のように、ヒエラルキーのCreateボタンから3DObjectのPlane (平面)を呼び出します
- ヒエラルキーでPlaneを選択すると、画面右側のインスペクターに詳細が表示されます
- インスペクター上部でオブジェクトの名前が変更できます。今回は Floor と名付けておきました

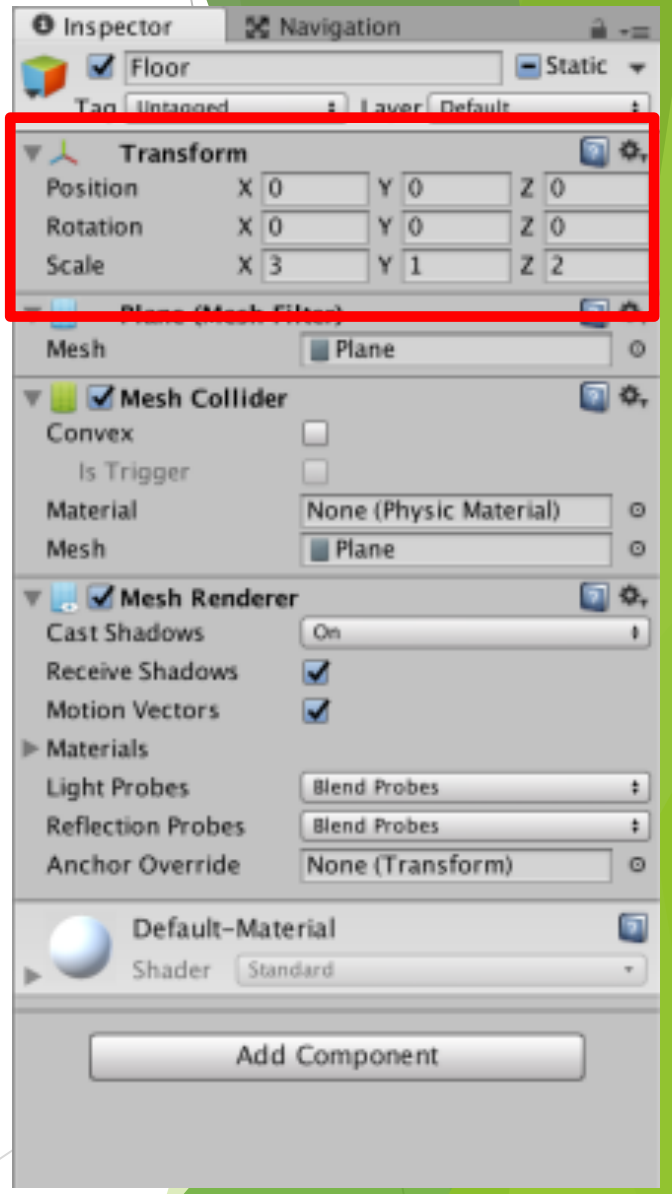




床の準備(2/2)

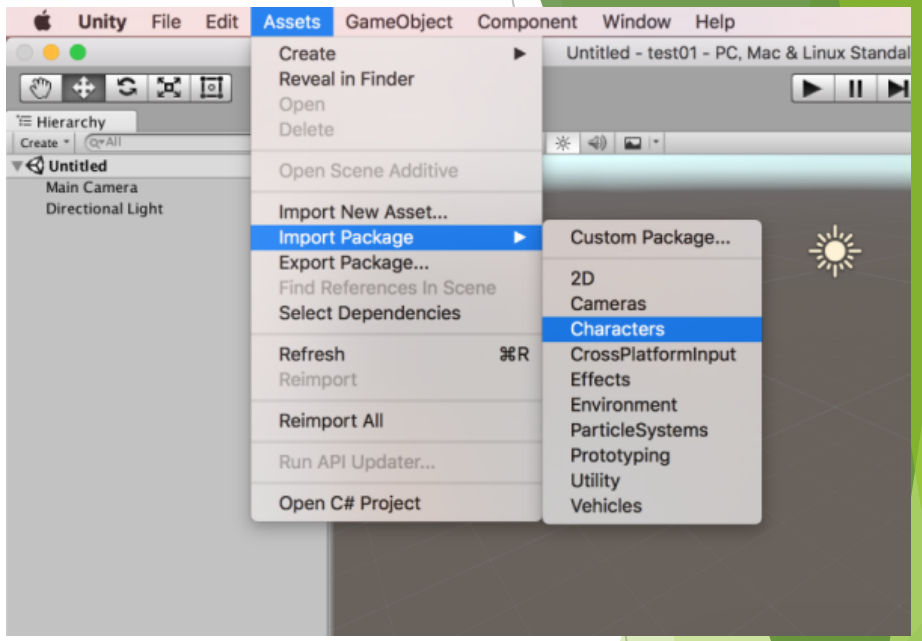
- Unityでは、オブジェクトの持つ色々な機能を、コンポーネントという塊で扱います
- インスペクターには、そのオブジェクトが持つコンポーネントの一覧が表示されます
- Transformは位置・角度・大きさを管理するコンポーネントで、オブジェクトの本体のような役割も持っています

- 今回はPositionを(0,0,0)、Rotationを(0,0,0)、Scaleを(3,1,2)としました。これで20*30の大きさのステージができました



プレイヤーの準備(1/2)

- プレイヤーには、スタンダードアセットに含まれているものを使います。
- まず、右図のようにAssetsメニューのImport PackeageからCharactersパッケージをインポートします
- 画面下側のプロジェクトウィンドウに現れたStandard Assetsというフォルダから、 Standard Assets → Characters → ThirdPersonCharacter → Prefabs と開き、ThirdPersonController.prefab をヒエラルキーにドラッグ&ドロップして追加します
- 名前にAIと付くものも後で使いますが、間違えないように注意しましょう



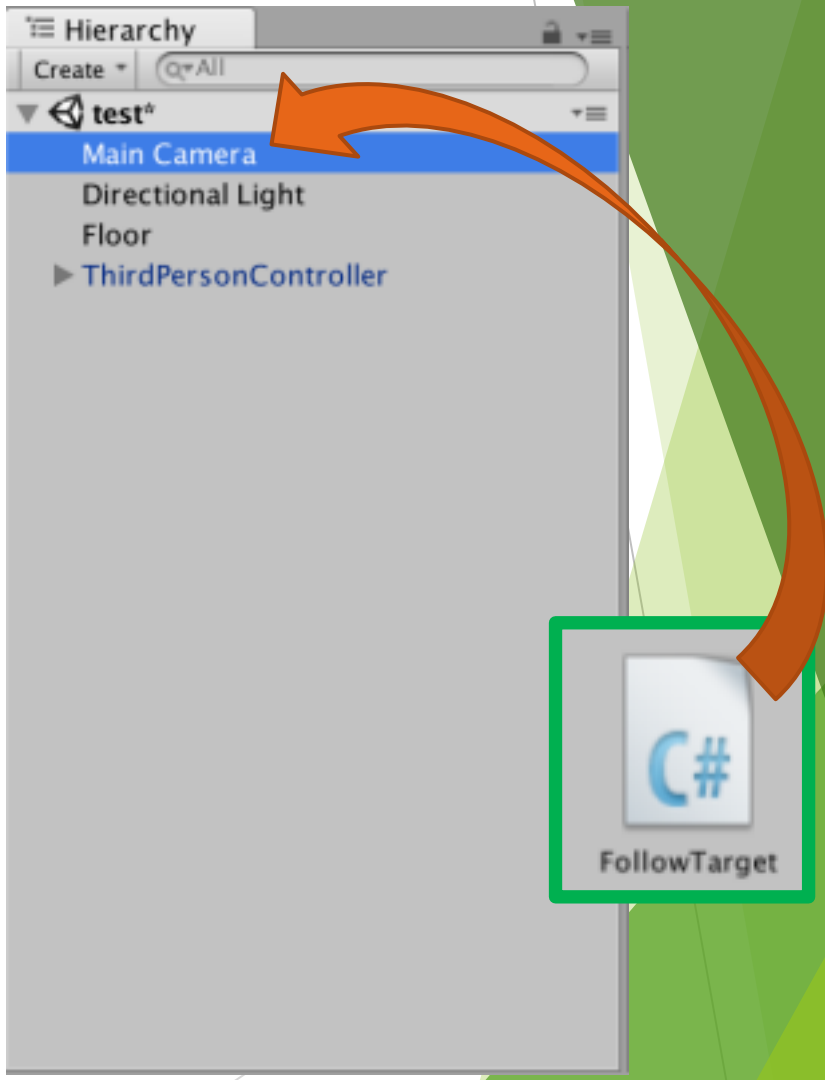
プレイヤーの準備(2/2)

- 以上のようにすると、それだけでもうキャラクターが動かせるようになっています。
- 画面上部中央の再生ボタンを押して、ゲームを実行してみてください。同じボタンを押すと停止できます

- WASDまたは上下左右キーで移動、スペースキーでジャンプできます。

カメラの設定(1/2)

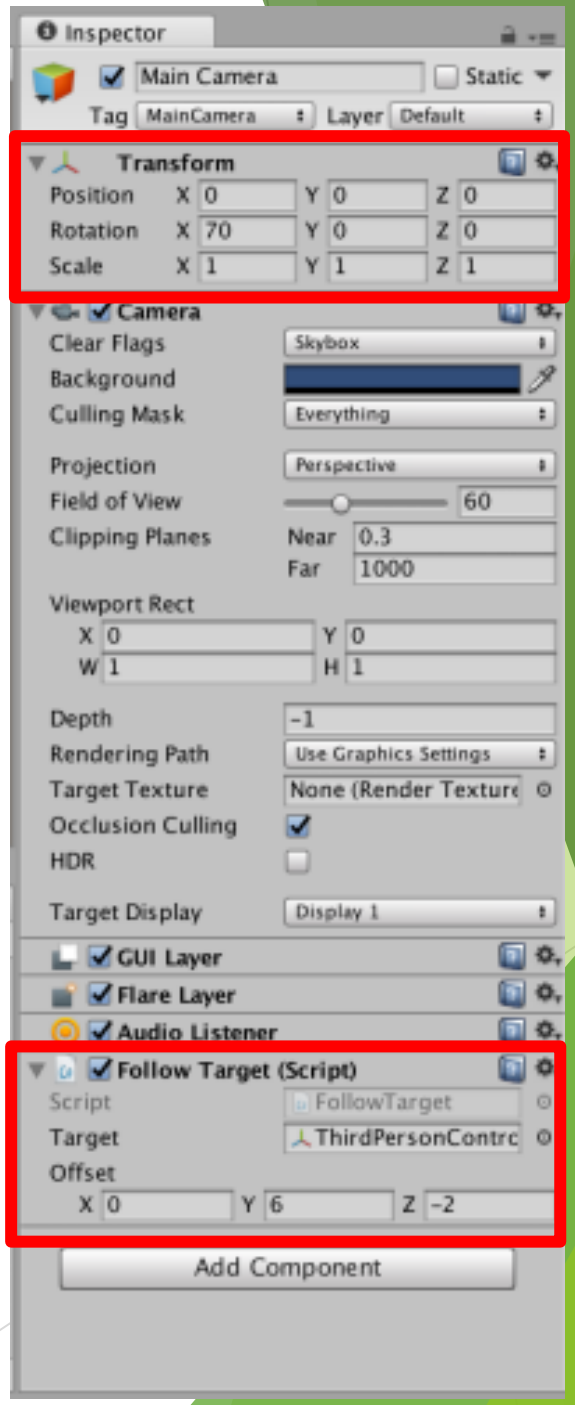
- カメラが固定されて操作しにくいので、キャラクターを追従するようにします
- プロジェクトウィンドウのStandardAssets→Utility内の、FollowTarget.csというファイルを使います
- これをヒエラルキーのMain Caneraにドラッグ&ドロップします



カメラの設定(2/2)

- カメラのインスペクターにFollow Targetという項目が追加されます
- Targetは追従する対象、OffsetはTargetと保つ距離を指定します
- ヒエラルキーのThirdPersonControllerをTargetにドラッグ&ドロップし、Offsetは (0,6,-2)にしました

- メインカメラのTransformは、
  Position (0,0,0), Rotation(70,0,0), Scale(1,1,1)
  にしました
- これでカメラがプレイヤーを映し続けるようになったと思います



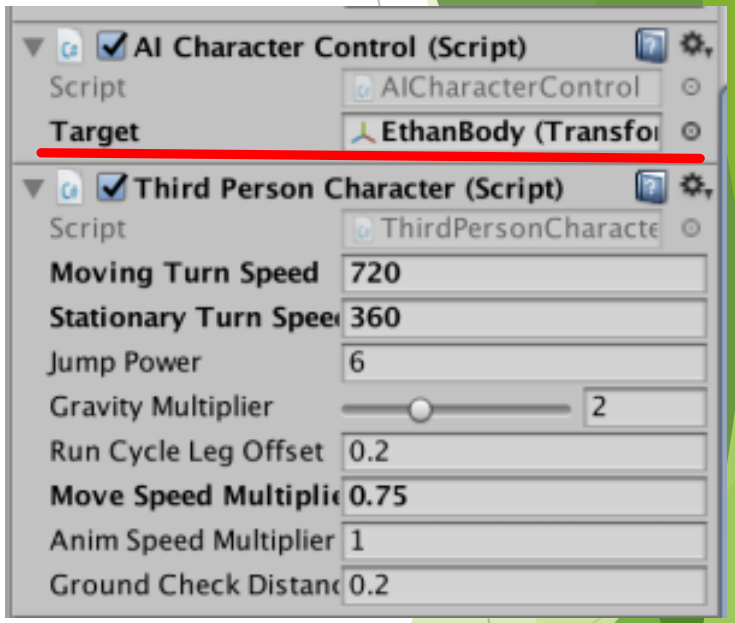
敵キャラの設定

- 敵キャラは、ナビゲーションという機能を使って、自動で動くようにします
- プロジェクトウィンドウの、Standard Assets → Characters → ThirdPersonCharacter → Prefabsと開くと、AIThirdPersonControllerというものがあります。これをヒエラルキーにドラッグ&ドロップして追加します(プレイヤーに使ったものとは別です)
- これはプレイヤーに使ったものとは異なり、ターゲットと、移動可能な範囲を指定してあげないと動きません

ターゲットの設定

- AIのインスペクターの下の方に、AICharacterControlというコンポーネントがあります。ここでターゲットを指定します

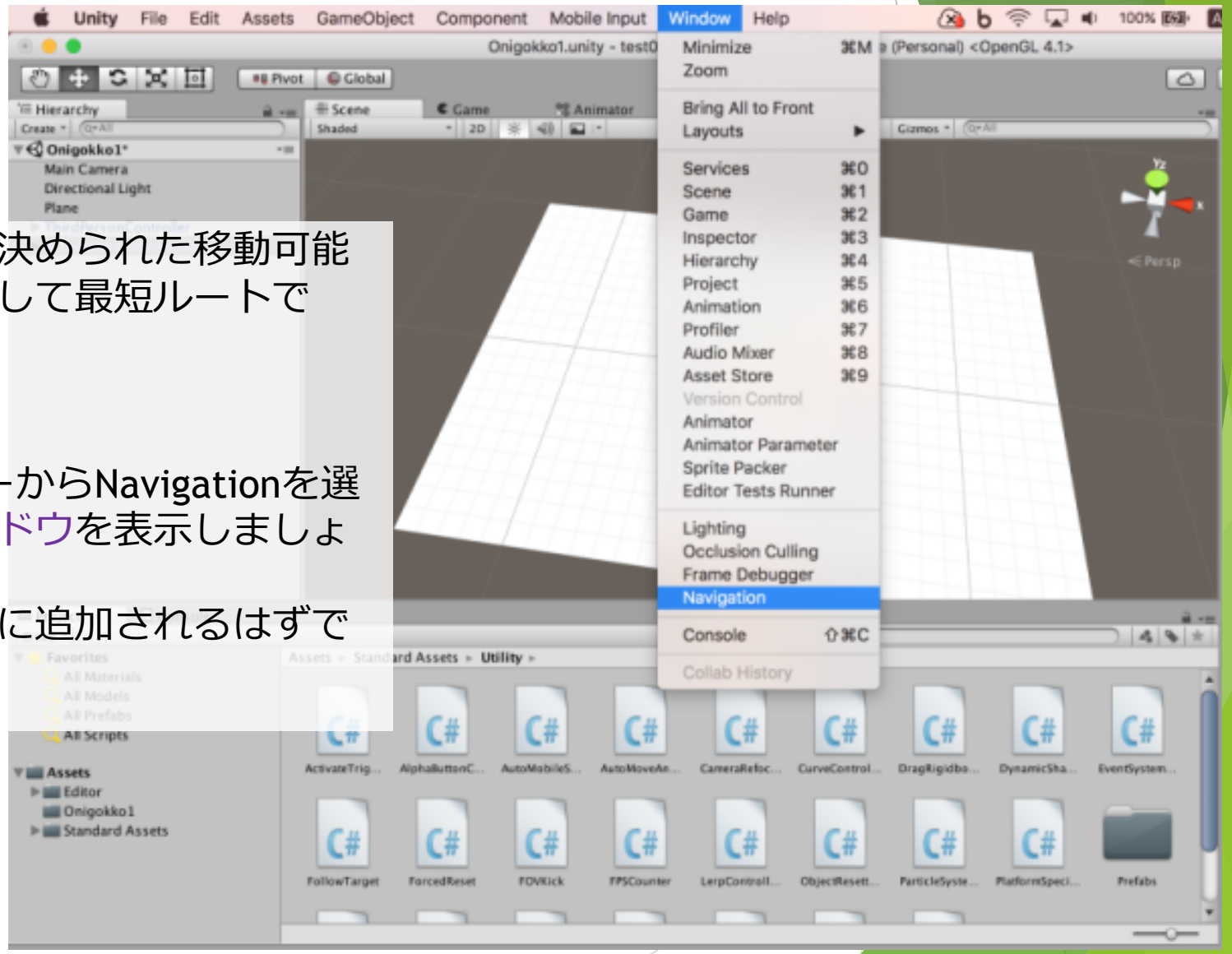
- ヒエラルキーのThirdPersonControllerをドラッグ&ドロップでTargetに設定しましょう
(Targetの欄の右側の丸を押して一覧から選択することもできます)



ナビゲーション

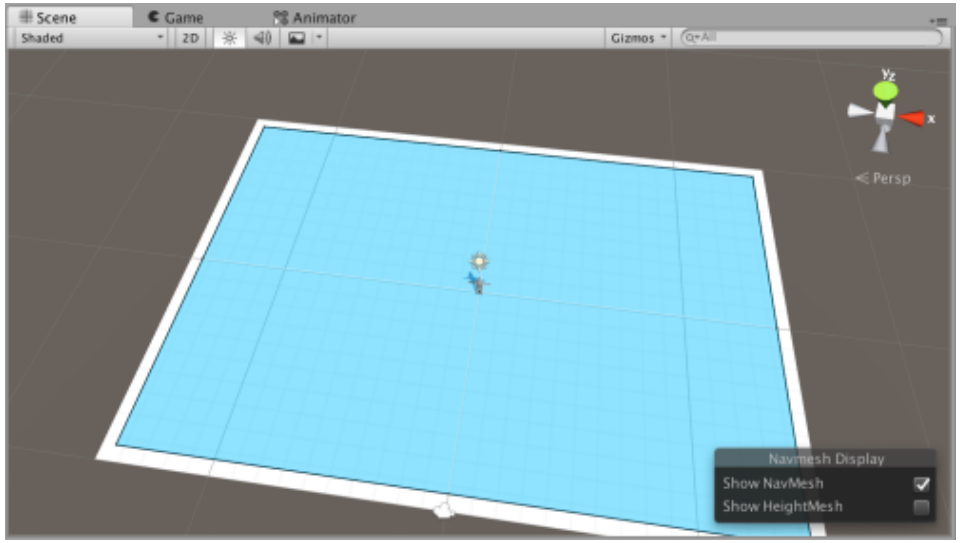
- ナビゲーションは、あらかじめ決められた移動可能な範囲の中で、自動で経路探索して最短ルートでターゲットに近づく機能です

- 右図のように、WindowメニューからNavigationを選択して、ナビゲーションウィンドウを表示しましょう。
  インスペクターと同じスペースに追加されるはずです



ナビゲーションメッシュの設定

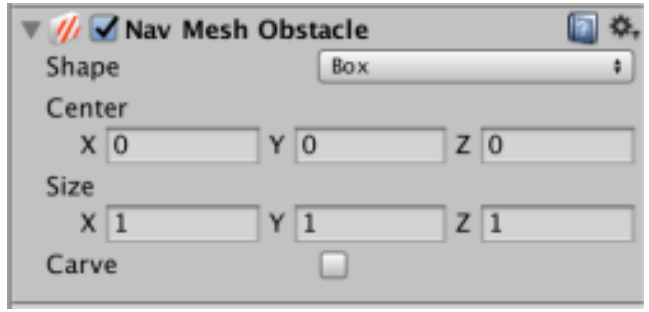
- ナビゲーションでAIが移動可能な範囲を表すものを、ナビゲーションメッシュと言います
- ヒエラルキーでFloor(床のオブジェクト)を選択してナビゲーションウィンドウを見ると、右図のようにFloorが表示されます。NavigationStaticにチェックを入れ、右下のBakeボタンでナビゲーションメッシュを生成します
- 下図のように床より一回り小さい青い面ができたら成功です



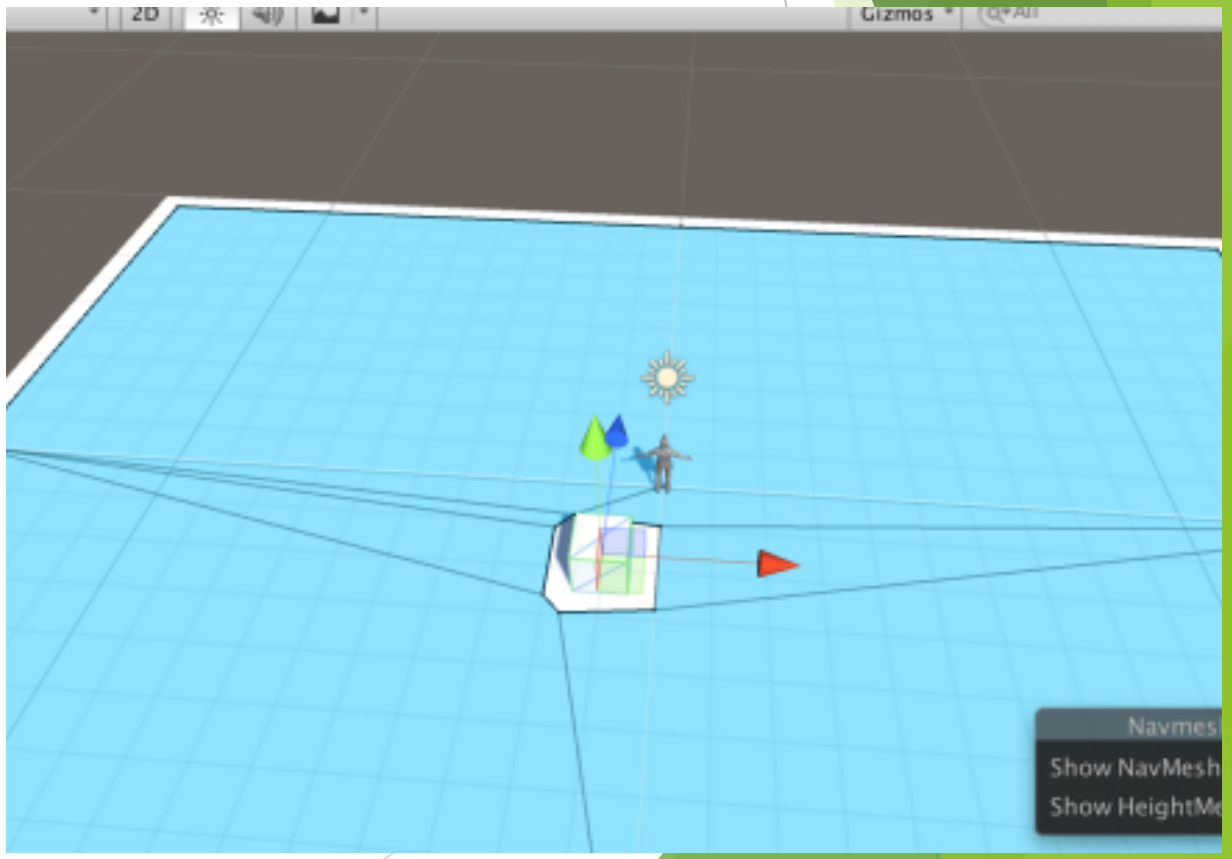
動作確認

- これで、敵キャラを動かす準備が整いました。再生ボタンで実行してみましょう。うまくいけば、プレイヤーに追従するようにピッタリついてきます
- AIが動かない場合は、ターゲットの設定に失敗している可能性があります。プレイヤーではなくAI自身を登録していないか確認しましょう
- AIの動作を確認したら、次は障害物の設定をします。

障害物の設定(1/3)



- ヒエラルキーで、はじめに作ったCubeを選択し、インスペクターからTransformのy座標を0.5にします
- インスペクターのAddComponentから Navigation → NavMeshObtacleを追加します。これを持つオブジェクトはナビゲーションにおける障害物とみなされます

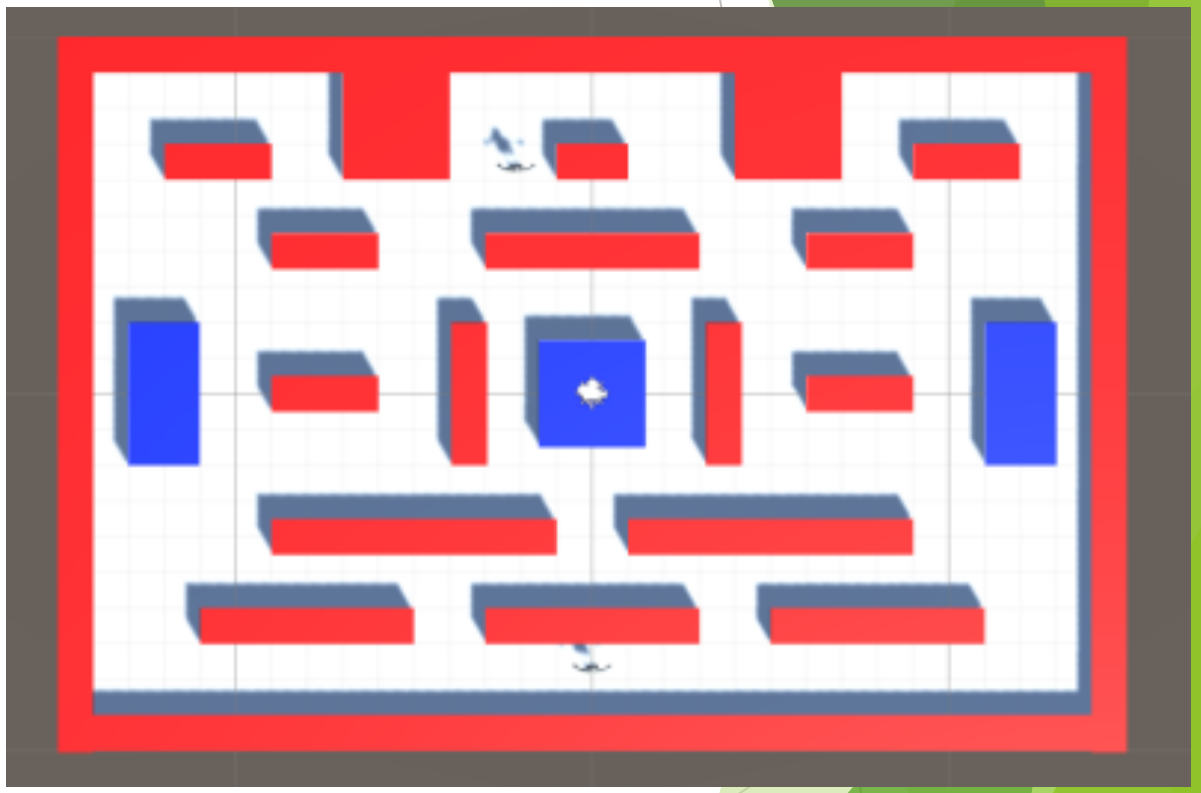


- 床と同様に、障害物もNavigation Staticに設定します
(ヒエラルキーでCubeを選択してナビゲーションウィンドウから設定できます)
- Bakeし直すと、右図のようにCubeの周辺に通過不可能な区域ができます

障害物の設定(2/3)

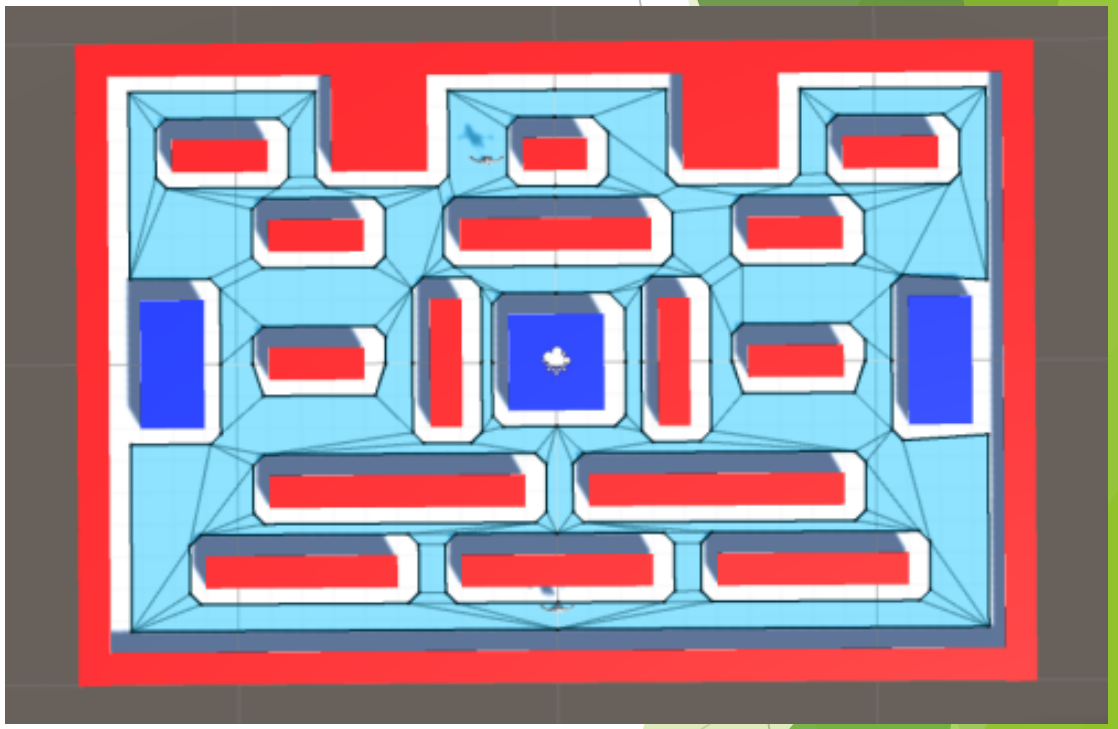
- 全ての壁にNavMeshObstacleを持たせ、NavigationStaticにする必要があるので、今作成したCubeを複製して壁を増やしていきましょう

- 右図は今回サンプルとして作成したステージです



障害物の設定(3/3)

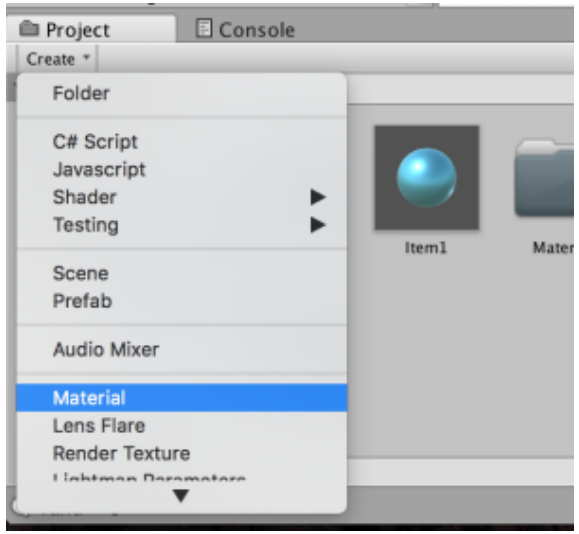
- 障害物を置き終えたら、ナビゲーションメッシュを生成し直します
- 全ての障害物が正しく設定できていれば、右図のように、障害物を避けた面が生成されます
- 壁同士の間はある程度離しておかないと、AIが通過できなくなってしまいます

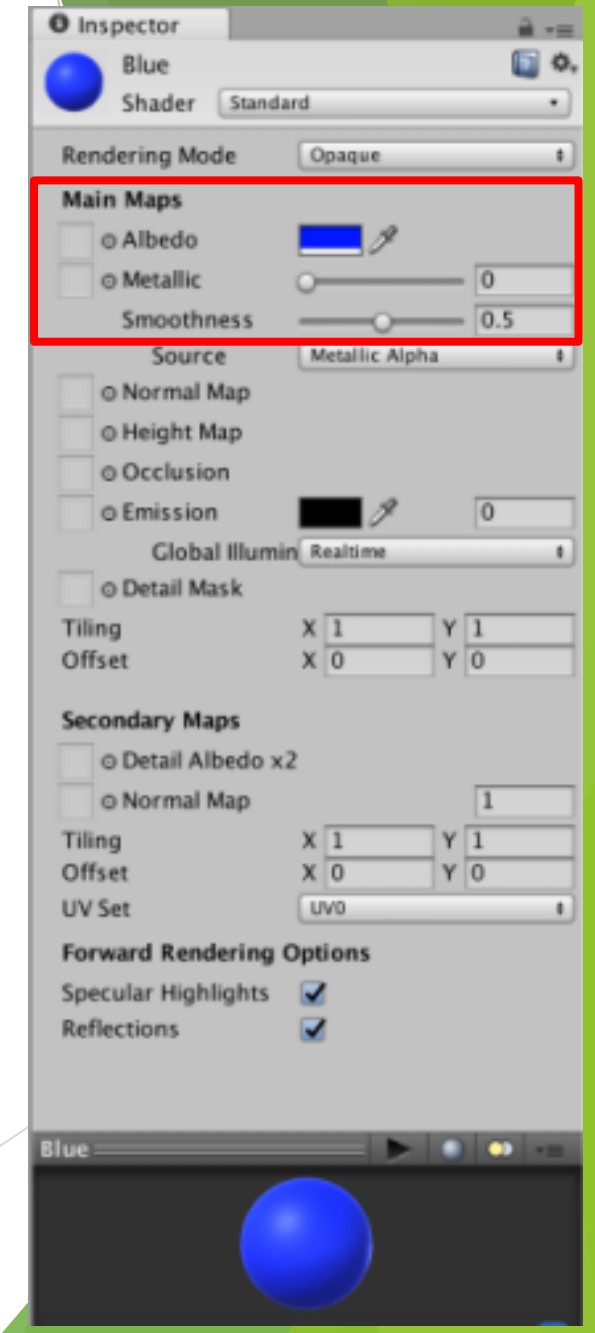


マテリアル

　見本では、壁やが床と同じ色ではわかりにくいので、色をつけました。Unityでは、マテリアルというものを使って質感や色を表現します。

- プロジェクトウィンドウのCreateからMaterialを作成します
- 赤線で囲った部分で基本的な設定ができます。Albedoはオブジェクト全体にかかる色、Metallicは金属らしさ、Smoothnessは表面の滑らかさを表すパラメータです
- 　マテリアルをオブジェクトにドラッグ&ドロップして、色を付けてましょう





最終調整

- これでAIと鬼ごっこができるようになります
- プレイヤーと敵の性能が同じだと逃げきれないので、それぞれのThirdPersonCharacterコンポーネントの数値を変更して調整します
- 現状ではただ歩き回るだけなので、ここから少しプログラムを書いてゲームとして完成させていくことになります。これは別の資料で解説する予定です。

  お疲れ様でした