Unity と Wii リモコンで簡単アーケードゲーム制作

トミー (@Tomy_0331)　著

2019-01-26 版　　発行

目次

第 1 章

はじめに

1.1 この本について

この本では「Unity でサンプルゲームは作ってみたけどそれからどうしたらいいかわからない」「入門レベルより少しクオリティの高いゲームを作ってみたい」，というような Unity 初心者〜中級者手前の人をターゲットにしています．

また，文化祭などで"映える"アーケードゲームを作ることをメインとしています．Wii リモコンを使ったシンプルな操作性は子どもでも遊びやすくウケがいいです (体験談)．

なお，この本では Unity 画面の見方や操作など基本的な部分の説明は省略することが多いです．プログラミングは C#における基本文法も説明を省かせていただくので分からない場合適宜調べるか一度入門サイトでミニゲームを作ってみることをおすすめします．

1.2 開発環境

以下の環境・ソフトで開発しました．

Windows10

一般的な OS．Unity は Mac 版でも操作は基本的に同じですが，今回使う Wii リモコンやペダルの Mac での入力方法は対象外とさせてもらうのでご了承ください．Linux も同様です．

Unity2018.3.2f1

今回のメインで使うゲームエンジン．3D ゲームを作るのに最適で個人からゲーム会社まで広い場所で使われています．

blender

フリーの 3D モデリングソフト．造形からテクスチャのマッピング，アニメーション作成までできます．3D ビューの操作が独特で混乱しがち．

Visual Studio 2017

Unity でスクリプトを書くときに使います．自動で Unity 環境の補完をしてくれます．

第 2 章

Unity でゲーム部分を作ろう

2.1　プロジェクトを作成する

Unity を起動し「New」から新規に作るプロジェクトの設定をします．Project name は「shoot」など任意の名前，Template は「3D」，他パスの設定などして Create project でプロジェクトを作成します．

2.2　地形を作る

ゲームの舞台となる地形を作ります．Unity には山や植物をペイントツールのように作る Terrain という機能があります．

Standard Assets のインポート

Terrain を使う前に，地面に適用するテクスチャや草木のデータを使うために公式の「Standard Assets」をインポートします．

Unity の Asset Store から「Standard Assets」で検索するとパッケージが出てくるのでダウンロードし，インポートします．インポート時に色々選択できますが，必要なテクスチャや草木データが入っている「Environment」のみでいいでしょう．

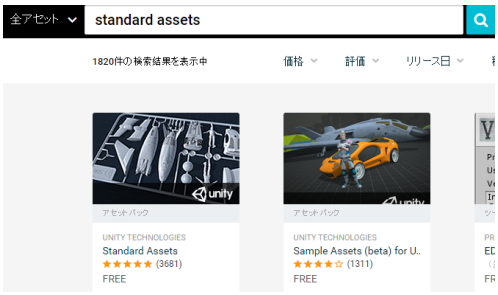

図 2.1: Standard Assets

Terrain の設置

「Hierarchy」の「Create」から「3D Object>Terrain」で作ります．「Inspector」の「Terrain」で設定ができ，「Paint Texture」を選択し「Edit Terrain Layers」から草のテクスチャ画像を指定しましょう．

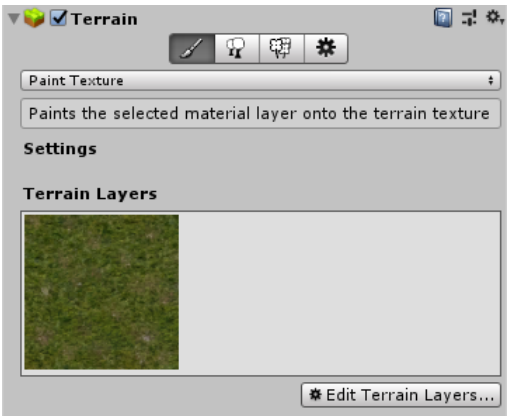

図 2.2: 草テクスチャの指定

2.3　敵 (ゾンビ) の用意

恐らく一番難しい敵キャラの動作を作ります．

ゾンビが今回唯一登場するキャラクターです．アニメーションやラグドール，移動 AI など初心者には難しいところがあるかもしれませんが，ここを作ることができると”ゲーム感”がとてもよく出てそれっぽくなります．

ゾンビのモデルをインポート

Asset Store で「zombie」と検索すると出てくる一番人気のアセットをインポートします．

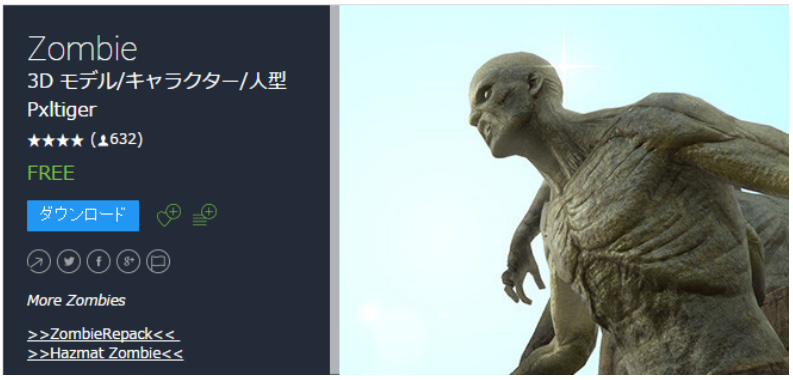

図 2.3: ゾンビの素材

筆者はこの時インポートしたファイル名の一つが大文字小文字の区別に関するエラーがなぜか出てきたので手動で修正しました.

ゾンビの死体を作る

恐ろしい見出し名ですが，ゾンビを撃った後に物理法則に従って倒れるようにラグドールを適用します．ラグドールとは，洋ゲーとかで敵が死ぬとぐったり人形のように崩れていくアレです．インポートしたアセットの Prefabs というフォルダにゾンビのプレハブがあるのでシーンに追加し，メニューの「GameObject>3D Object>Ragdoll...」から Create Ragdoll ウィンドウを開きます.

頭，腕，足など指定するためシーンのゾンビの階層を展開してドラッグアンドドロップで一つずつ体のパーツを当てはめていきます.

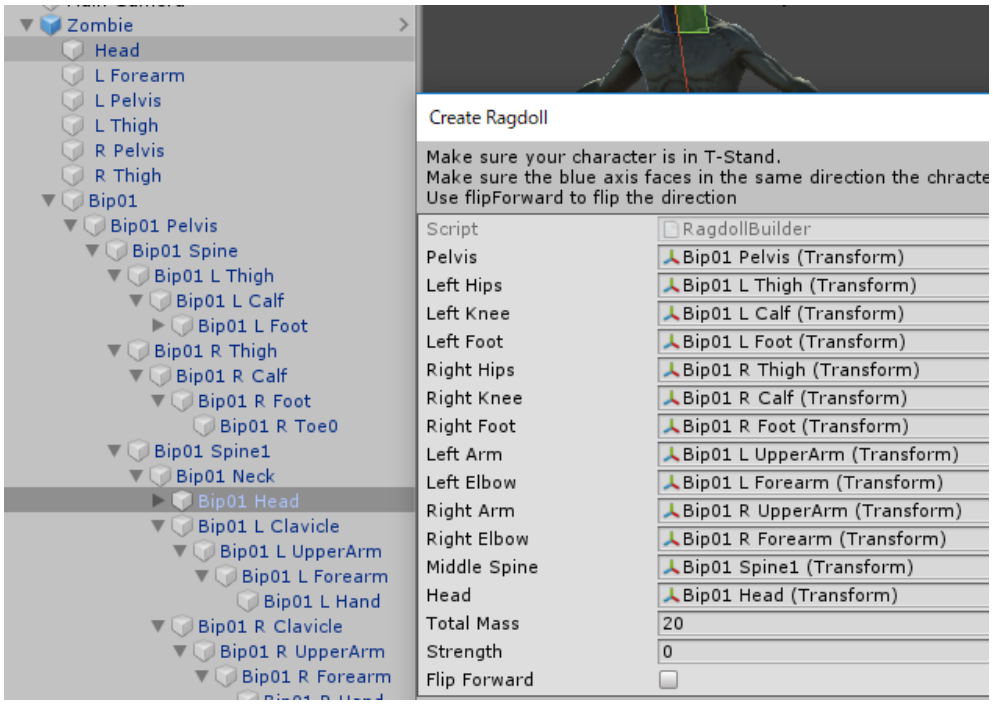

図 2.4: ラグドールの設定

タグを付ける

この後ゾンビの撃つスクリプトを書きますが，プログラムで識別するためのタグをつけます．インスペクタの「Tag ＞ add tag」から「enemy」と名付けたタグを追加し，ゾンビのコライダがついているオブジェクト全てに適用しましょう．階層を shift キーで推しながら複数選択してから Tag を設定すると一度に付けることができます．

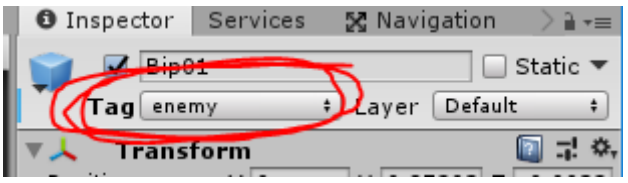

図 2.5: tag の設定

2.4　ゾンビを撃つスクリプトを書く

ここでゲームの動作部分になるプログラムを作ります.「ShotCam」という名前でスクリプトを新規作成し[*1]，以下のコードを書きます.

スクリプトは Main Camera にアタッチします.

ShotCam.cs

```
public class ShotCam : MonoBehaviour {
  void Start() {
  }
  void Update() {
    GameObject clickObject=getClickObject();
    //クリックしたのが敵なら
    if (clickObject!=null && clickObject.gameObject.tag == "enemy") {
      //アニメーション無効
      clickObject.transform.root.GetComponent<Animator>().enabled = false;
      Vector3 vec = clickObject.transform.position - this.transform.position;
      //射撃した部位に力を加える
      clickObject.GetComponent<Rigidbody>().velocity = vec.normalized*15;
      //ゾンビ側のスクリプトの death() 呼び出し
      clickObject.transform.root.GetComponent<Zombie>().death();
    }
  }
  // 左クリックしたオブジェクトを取得する関数
  public GameObject getClickObject() {
    GameObject clickObject = null;
    if (Input.GetMouseButtonDown(0)) {  //左クリック
      Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition);
      RaycastHit hit = new RaycastHit();
      if (Physics.SphereCast(ray, 0.1f, out hit)) {
        clickObject = hit.collider.gameObject;
      }
    }
    return clickObject;
  }
}
```

続いてゾンビのオブジェクト (一番上の階層) に Zombie というスクリプトを作成し以下のクラスを追加しアタッチします.

Zombie.cs

[*1] Unity では C#の他 JavaScript,Boo といったプログラミング言語を使えましたが，廃止となっています.

```
public void death() {
  Destroy(gameObject);  //オブジェクトを消す
}
```

プログラム解説

まずは ShotCam の説明をします.

getClickObject 関数は左クリックされると画面上に Ray と呼ばれるレーザーのような線を出し，衝突したオブジェクトを取得し return で返しています．毎フレーム呼び出される Update 関数では getClickObject 関数でクリックしたオブジェクトを検知し続け，if 文にて enemy タグが付いているかを判別します．enemy タグはゾンビの胴体，手足，頭についており clickObject.transform.root でいずれからも一番上の階層のオブジェクト「Zombie」のアニメーションを参照でき，無効にしています.

変数 vec は撃った部位に力を加えて吹き飛ぶ演出をするためのベクトルを保持しています．撃った部位の座標からゲーム画面を映すカメラの座標を減算することでベクトルが取得できます．そして normalized で単位ベクトル化し Rigidbody コンポーネントで力を加えています．「*15」は力の大きさです．試しに大きくして遊んだり自分好みの値にしてみるのもいいでしょう.

ベクトルが分からない，という人もいるかと思いますが下の図の「矢印」がベクトルだと思ってください．ベクトルは方向と力の情報を持っていて座標点 b-座標点 a で a から b へのベクトルになります．その「矢印」をゾンビ撃った部位に付けることでその点に力が加わる，ということになります.

最後にゾンビ側のスクリプト Zombie.cs の関数 death を呼び出して一定時間後オブジェクトを消すようにしています.

Zombie.cs では Destroy でゾンビ本体を丸ごと削除しています.

シーンのカメラに見える位置にゾンビを設置し，Unity の三角ボタンで実行してみましょう．クリックしてゾンビが消滅すれば現状は OK です.

2.5　ゾンビが歩くようにする

ゾンビをプレイヤーまで歩かせるために，経路に沿って移動していく Agent とアニメーションとして体が動く Animation の二つの機能を使います．=== 経路探索 AI の実装 Unity には嬉しいことに経路探索 AI が自動で入っています．まず，ゾンビに「Nav Mesh Agent」のコンポーネントを追加し図のように設定します.

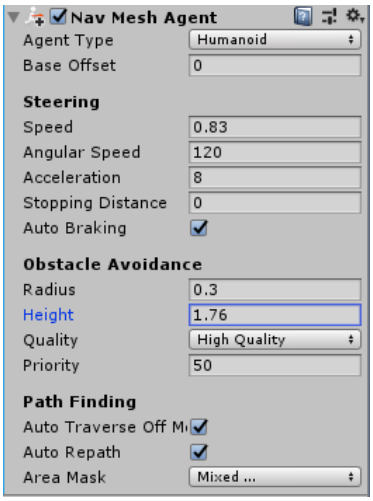

図 2.6: Nav Mesh Agent の設定

次に Terrain に経路探索のために必要な作業をします．Terrain を選択するとインスペクタの隣に「Navigation」というタブが出てきます．選択し「Bake」からパラメータを調整して Bake することでゾンビが歩ける場所が青く表示されます．

なお，試しに障害物として画像では Cube を置いていますがインスペクタから「static」にチェックを入れるのを忘れないようにしてください．

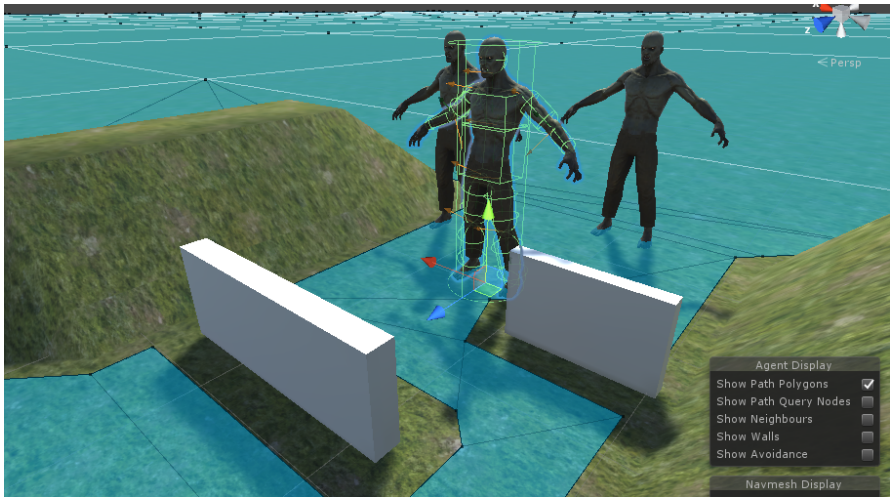

図 2.7: 経路となる領域を Bake した様子

アニメーション遷移の実装

移動中には歩くアニメーション，止まるときには待機のアニメーションを適用するため Animation の設定をします.

※ちゃんと書く

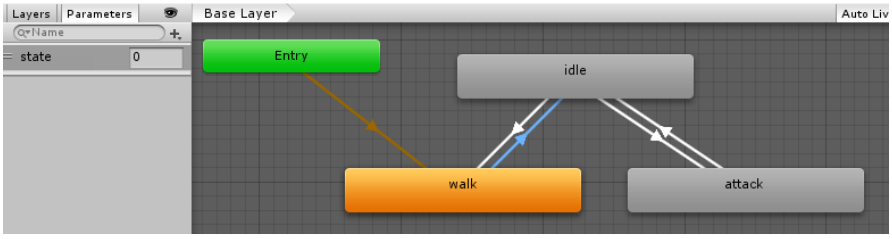

図 2.8: 経路となる領域を Bake した様子

スクリプトを書き換える

経路探索のために AI 機能をインポートします

Zombie.cs

```
using UnityEngine.AI;
```

そして以下のようにガラッと書き換えていきます

Zombie.cs

```
public class Zombie : MonoBehaviour {
private new GameObject camera;
private NavMeshAgent agent;
private bool stop;
private enum state { walk,idle,atack }  //アニメーションの状態
private Animator animator;
void Start() {
  camera = GameObject.Find("Main Camera").gameObject;
  agent = GetComponent<NavMeshAgent>();
  agent.SetDestination(camera.transform.position);  //目標座標を設定
  stop = false;
  animator = GetComponent<Animator>();
}
```

```
  void Update() {
    //ゾンビが目標点まで 2m 近づいたら立ち止まる
    if (!stop && Vector3.Distance(camera.transform.position, this.transform.position) < 2f) {
      animator.SetInteger("state", (int)state.idle);
      Vector3 p = camera.transform.position;
      p.y = this.transform.position.y;
      transform.LookAt(p);
      agent.isStopped = stop = true;
    }
    else {
      //animator.SetInteger("state", (int)state.walk);
    }
  }
  //死ぬ処理
  public void death() {
    GetComponent<Animator>().enabled = false; //アニメーション無効
    Invoke("destroyObject", 5f);  //5 秒後に消滅させる
    SetKinematic(false);  //物理演算を付ける
    agent.enabled = false;
  }
  void destroyObject() {
    Destroy(gameObject);  //オブジェクトを消す
  }

  public void SetKinematic(bool newValue) {
    Component[] components = GetComponentsInChildren(typeof(Rigidbody));
    foreach (Component c in components) {
      (c as Rigidbody).isKinematic = newValue;
    }
  }
}
```

■コラム: Unity で使われるクラス

Vector3 は 3D 空間のベクトルを扱うための関数です．x,y,z の要素を持っています.

Unity と Wii リモコンで簡単アーケードゲーム制作

2019 年 1 月 26 日　初版第 1 刷　発行

著　者　　トミー (@Tomy_0331)