

1. Unity による 3D ゲームの作成

角舘大, 菊池舜

指導教員 石舘勝好

1. はじめに

ゲームアプリに興味があり, iPad のようなタッチパネルに対応したゲームアプリを作成したいと考えた. また, Unity を使用したゲーム作成が過去の卒業研究でいくつかあるが, それらとは志向を変えて, 自分たちのゲームを作ろうと思った.

私たちが作成するゲームは, 3D ダンジョン RPG というジャンルになる. 有名なゲームとして, ドラゴンクエスト等がある. これは 3D 視点で, ダンジョンに潜りつつ戦闘を繰り返し, 探索を行い, 武器や防具を整え, より深く潜りボスを倒すタイプのゲームである.

2. 研究概要

2.1 ゲームの仕様

ゲームの名称は「Curatenia Poveste」で, 主人公(以下, キャラクター)と敵(以下, エネミー)が存在する. 地下にある寺院をモチーフにしたダンジョン内でエネミーを倒しながら奥へ進みボスを倒すことでクリアとなる. ステージは 1 つで, 特に戦闘面に特化し作成する. プラットフォームは WindowsPC, iOS である.

2.2 開発環境

Windows10 Pro 搭載の PC と Mac OS Mojave を搭載した MacbookPro を使用する. 基本的に, 表 1 の環境で開発を進め, iOS に移植する際のビルドに表 2 の環境を使用する.

表 1 Windows 用開発環境

OS	Windows10 Pro
IDE	Unity2. 12f
使用言語	C#
使用機材	ThinkCentre

表 2 iOS の開発, 実装環境

OS	macOS Mojave
IDE	X-code
使用言語	C#
使用機材	MacbookPro, iPad Pro
SDK	iOS SDK

2.3 Unity について

Unity とは, 統合開発環境を内蔵し, 複数の機材に対応するゲームエンジンである. ウェブプラグイン, デスクトッププラットフォーム, ゲーム機, 携帯機器向けのコンピューターゲームを開発するための言語として, C#, UnityScript(JavaScript), Boo の 3 種類のプログラミング言語に対応している. 本研究では主に C#を用いて開発を行う.

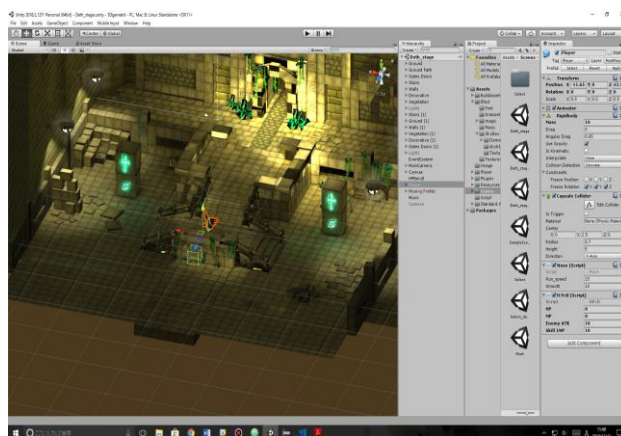


図 1 Unity の開発画面

2.4 Xcode について

ソフトウェアを開発するための Apple が提供する統合開発環境であり, MacOS 上でのみ稼働する. 今回は作成したゲームを iOS 用にビルドするために使用する.

2.5 AssetStore

AssetStore とは、Unity で使用できる 3D モデルの素材や画像、開発効率を上げるプラグイン等を購入できるショップである。ここでダウンロードできるものは有料のものが多く、無料で使える素材も数多く扱われている。本研究では有料と無料、両方の Asset を使用して開発をする。

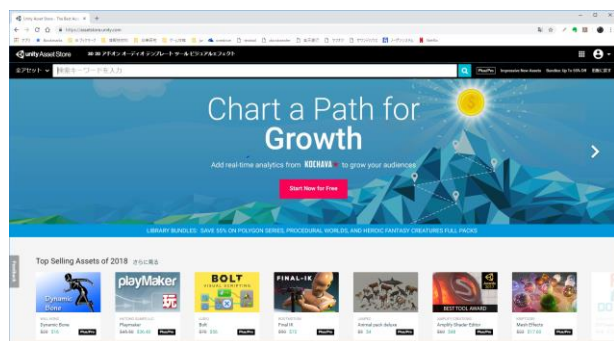


図 2 AssetStore

3. 作業工程

3.1 役割分担

本研究では二人での共同作業になる。そのうえで、主な作業を分担することにした。

角館：UI, Music, 背景などの絵

菊池：ゲームシステムの基幹部分, AI

共同：プロット, ゲームシステム, ステージ

必ずしもこの通りにではないが、この分担で責任を負う。

3.2 キャラクター、ステージの作成

3.2.1 キャラクターの作成方法

当初オリジナルの 3D モデルの作成も考えたが、現実的ではなかったため、AssetStore で公開されている物の中から選び使用することとした。

キャラクターとして StoneSnail 氏が作成した DreamWorld シリーズの Death(\$20)という 3D モデルを購入した、また、エネミーの 3D モデルとして、Maksim Bugrimov 氏の作成した Characters Creature(\$5)を購入し使用した。

ステージに使用する素材は Cartoon Temple Building(Free)という Asset を使用する。

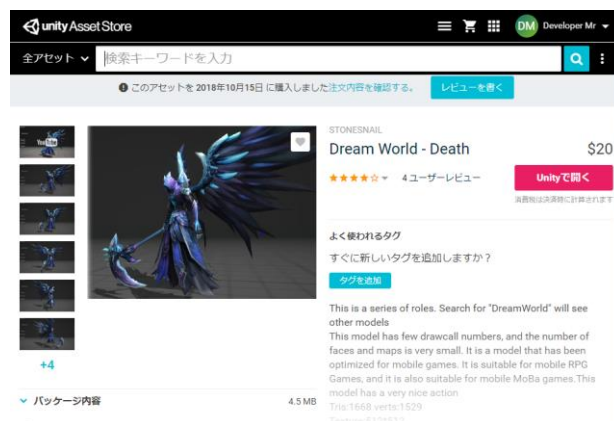


図 3 DreamWorld-Death (キャラクター)

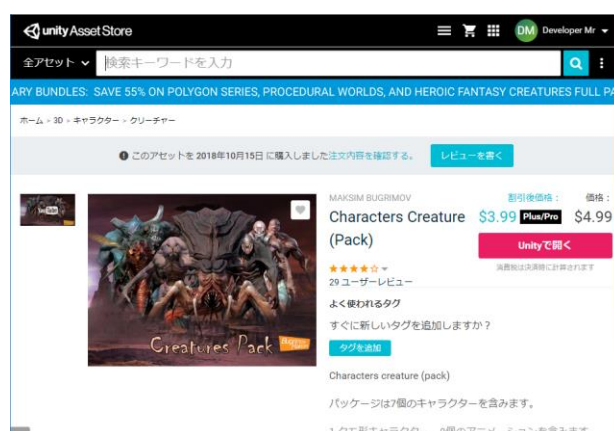


図 4 Characters Creature (エネミー)

3.2.2 ステージの作成方法

AssetStore から図 5 の Cartoon Temple Build をダウンロードする。

Cartoon Temple Build は、地下にある寺院をモチーフにした Asset で、図 6 に示すような素材や図 7、図 8 に示すような既に prefab 化されているものが入っている。

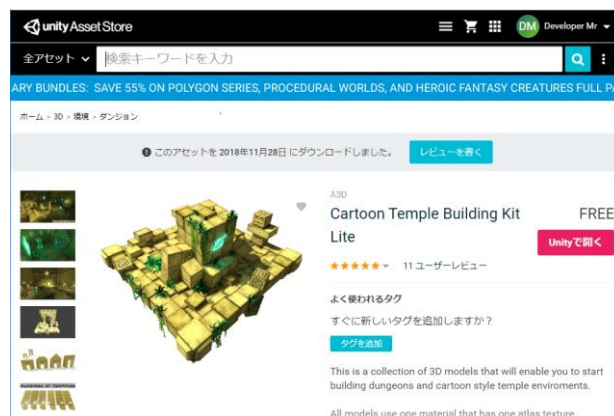


図 5 Cartoon Temple Build (ステージ)

この Asset の中に含まれる素材を組み合わせることで図 9 のようなステージを作成する。この時、一つ一つ Prefab 化されている素材を Scene 上に配置、位置調整する作業を繰り返している。

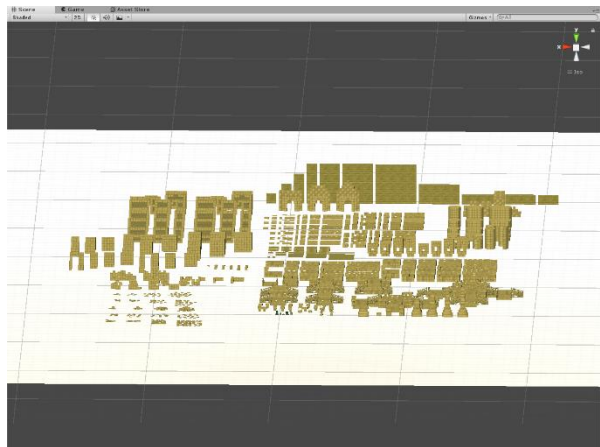


図 6 全ての素材

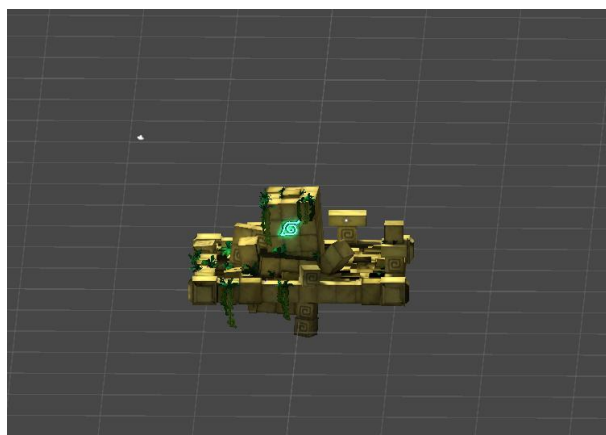


図 7 素材例(1)



図 8 素材例(2)

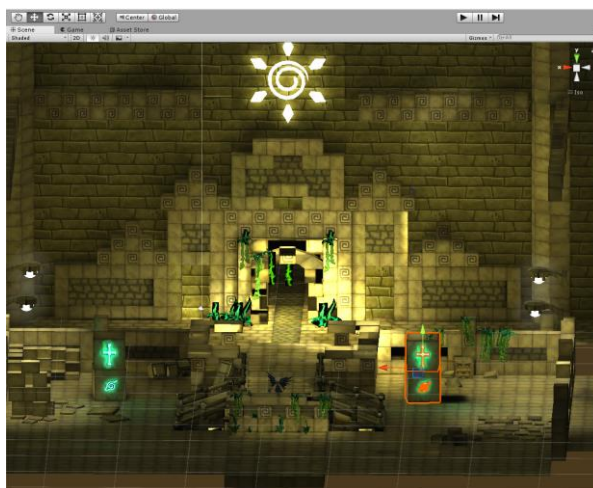


図 9 ステージ作成過程

前述の素材のほかに幾つか用意されている素材の例として 図 11, 図 12 を上げる。今回は素材を一から作成することはせず、素材を利用してステージの作成を行った。

このほかに SkyBox での空の変更や Fog の設定をする。これらの設定をすることで単なる陰影ではなく、よりリアルな見た目にすることができる。この設定に関して詳しくは後述する。

3.3 キャラクターの選定

まず初めに、Cartoon Temple Build の世界観に合い、かつイメージに合うキャラクターを探し、いくつかの候補の中からアニメーションがあらかじめ用意されていて、二人のイメージに合致する図 3 の Death を購入した。

Asset の購入にはアカウントの登録とクレジットカードを登録。単位が円ではなくドルなので購入の際には注意が必要になる。

3.4 アニメーションの作成, 変更

付属しているアニメーションは(○○や○○といった)一連の流れを自動でループするようあらかじめ設定されている。それを一度解除し、図 10 のとおり、任意のタイミングでアニメーションを起動するようにリレーションの設定をやり直す。

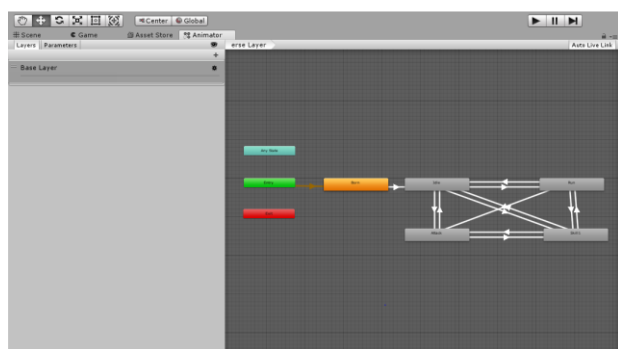


図 10 アニメーションの再設定

現在は付属のアニメーションのみを使用しているが進捗状況によっては、オリジナルあるいは既存のアニメーションを AssetStore から入手し変更をする。

3.5 UI 作成

3.5.1 UI の作成,

GUI の Canvas を設定し、その上に Button や SeekBar などを配置する。シーン'Start'では Image のみを配置し、Canvas にシーン遷移と Fadeout のスクリプトをアタッチした。シーン'Select'では背景を Image ではなく Scene をそのまま背景とした。Button とキャラクター(Death)の住処をイメージした家を配置し、Camera で撮影する形にした。それらのシーン遷移を経て、ゲーム本編へ入る。

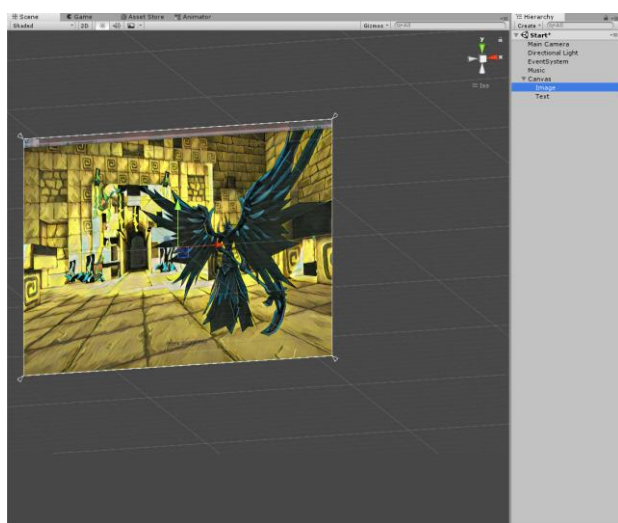


図 11 UI の作成過程



図 12 実行画面

3.5.2 UI の遷移

Scene から Scene に遷移(移動)するには、図 13 のように BuildSetting でシーンを追加しなければならない。

こうすることで、キーボードからのキー入力や Canvas 上に配置した Button からの入力を検知し、任意のタイミングでのシーン遷移が可能になる。

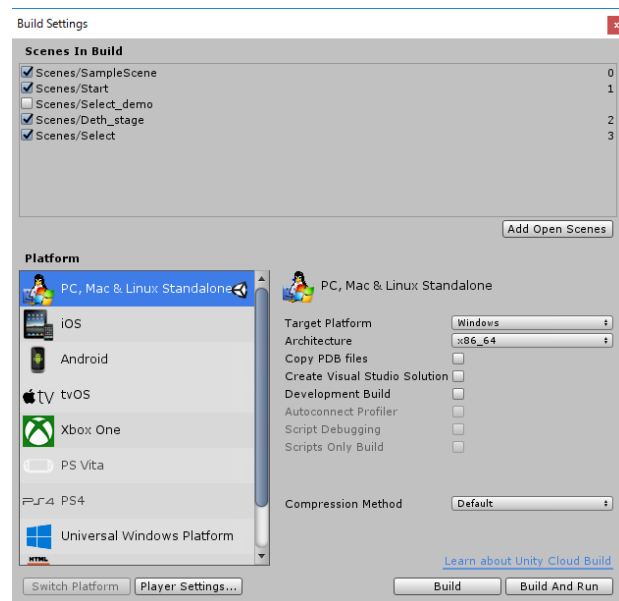


図 13 BuildSetting

しかし、Unity で作成したシーン遷移ではシーンが一時停止してから遷移してしまう。一般的なゲームではシーン遷移の時 FadeOut をする。そこで私たちのゲームでも FadeOut を実装することにした。Canvas 上に Panel を用意し透明度を可変することで、急激なシーンの移り変わりではなく、なめらかな移り変わりになる。これを FadeManager として用意し、図 14 のように他の Script から簡単に参照できるようにした。これにより、FadeOut したのち、シーン遷移が実行される。

```
void Update () {
    //Space キーでシーン遷移
    if (Input.GetKeyDown("space")) {
        //FadeManager から FadeOut を呼び出し
        FadeManager.FadeOut(1);
    }
}
```

図 14 FadeOut 使用例

3.5.3 BGM

BGM は魔王魂⁽³⁾からフリー音楽をダウンロードし、Start, Select, 本編用の 3 つの音楽に使用している。音楽の終わりに近づくと違和感の無いタイミングでループするように図 15 の Script により制御している。

```
void Start () {
    audioSource = gameObject.
    GetComponent<AudioSource>();
}
void update(){
    if (Input.GetMouseButtonDown (0)) {
        //左クリックを検知したら audioClip を流す
        AudioSource.PlayOneShot(audioClip);
    }
}
```

図 15 BGM 制御 Script

3.6 プログラミング

3.6.1 キャラクターの動き

キャラクターを動かす方法として transform を使う方法, Rigidbody や CharactorController, そして Vector を使う方法の 4 つがあげられる。今回は主に transform を使用してキャラクターを動かしている。簡易的に動かす場合は CharactorController をアタッチした方が早いので、ゲームの流れができるまではこれを使用する。その Script(Move.cs)の一部を図 16 に示す。

```
Var cameraForward =
    Vector3.Scale(Camera.main.transform.forward,
    new Vector3(1, 0, 1)).normalized;
Vector3 direction = cameraForward * Input.GetAxis
("Vertical") + Camera.main.transform.right * Input.
GetAxis ("Horizontal");
//Run
if(direction.magnitude > 0.1){
    //体の向きを変更
    Quaternion rotation = Quaternion.
    LookRotation(direction);
    transform.rotation = Quaternion.Lerp(transform.
    rotation, rotation, Time.deltaTime * smooth);
    //前方へ移動
    transform.Translate(Vector3.forward * Time.
    deltaTime * run_speed);
    //アニメーション用
    motion.SetBool("Running", true);
}
```

図 16 任意の方向に走る Script (一部)

3.6.2 エネミーAIの動き

エネミーがキャラクターを感知し、近くにいるときのみ近寄ってきてキャラクターを攻撃する。キャラクターが検知できない間は、その場で待機する。

3.7 ステージ効果の追加

先に述べた通り、Fog の追加や照明、空を設定することでゲームの中は息づき、よりリアルなものを目指すことができる。

3.8 小物の追加

ランタンや丸太などの小物を AssetStore より無料のものを入手し、配置することでクオリティが上がる。今回も家やランタン丸太や木などを各 Scene に配置している。

4. 研究を通して

プログラミングや Unity を使用することで様々な問題があったが、参考文献が多かったのもあり、滞ることなく進めることができた。また、ゲームだけではないと思うがいろいろなアイデアをだすことがとても重要だと思い知った。手段はあっても何を作っているかが決まらない故に進まないことが多々あった。

問題点としては、iOS デバイスへの移植を実現するには必須になってくる全体の処理の軽減化だ。空の Scene では動くプログラムも、本ゲームに移すと正確に動かないなどの不具合も出ることが確認できた。そのためには、オブジェクトの統合や、オクルージョンカリングなどの方法による改善が必要である。

5. 終わりに

目標である、“遊べるゲーム”を作ることができたと思う。iOS デバイスへの移植は断念したが、スタートからエンディングまで納得したものを作ることができた。

普段遊んでいるゲームは私たちが作ったものとは比べるまでもないくらいの差だが、それに近づけるよう試行錯誤したのは、とても楽しくいい経験だった。

6. 参考文献

(1) 「Unity で神になる」

著者：本廣 鉄夫

出版社：オーム社

(2) 「Unity ゲームプログラミングバイブル」

著者：吉谷 幹人 (著)， 布留川 英一 (著)，

出版社：株式会社 ボーンデジタル

(3) 魔王魂

<https://maoudamashii.jokersounds.com/>