Portfolio

福岡デザイン&テクノロジー専門学校
3年ゲームプログラマー専攻
服部 怜央

目次

1. プロフィール

2. ゲーム提出作品について

3. 制作実績

4. 今後の目標

プロフィール

名前	服部 怜央	
趣味	ゲーム、アニメ、マンガ、 映画・音楽鑑賞	
特技	バドミントン、山岳	
メール	leohattori@icloud.com	
GitHub	https://github.com/reo1316hw	
スキル	C++	★★★★★
	C#/Unity	★★★
	Git	★★★★
	DXライブラリ	★★★
	PhotoShop	★★

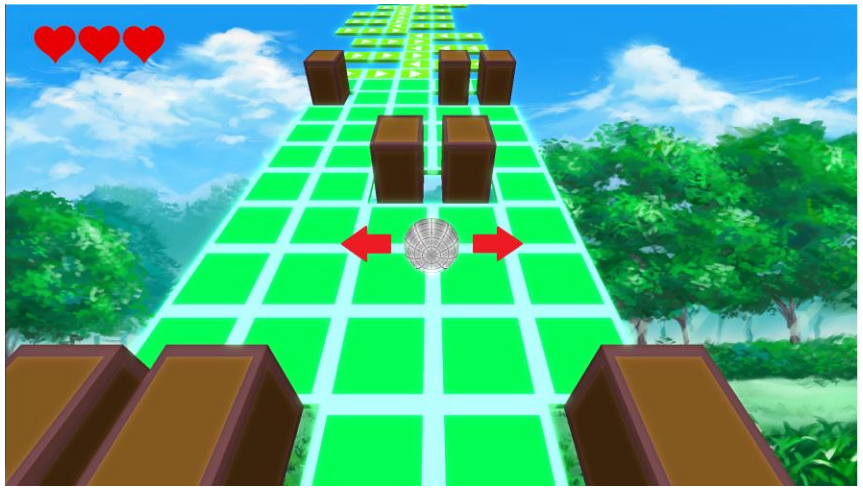
ゲーム提出作品　概要

作品名	BALLRUN
ジャンル	ハイパーカジュアルゲーム
開発環境	OpenGL、C++
対応機種	PC
制作人数	1人
制作期間	5ヶ月
制作時期	2020年9月～2020年1月
担当	背景イラスト、Bloom以外すべて

ゲーム提出作品　企画

ハイパーカジュアルゲーム
最終目標はゴールを目指し、
３つのステージをクリアすること

ボール(プレイヤー)は前に自動で進み、
横移動の操作で障害物を避けていく。



ゲーム提出作品　プログラム(基本システム)

基本となるGameObjectクラスとComponentシステム

全てのオブジェクトの基底となるGameObjectクラス

オブジェクトの状態　タグ　座標　速度　サイズ　回転　メッシュ　矩形当たり判定

プレイヤー　床　障害物　カメラ　背景　UI　エフェクト

全てのGameObjectに付け外しできるGameObjectの補助クラスComponent

ColliderComponent　AnimationComponent　ParticleComponent

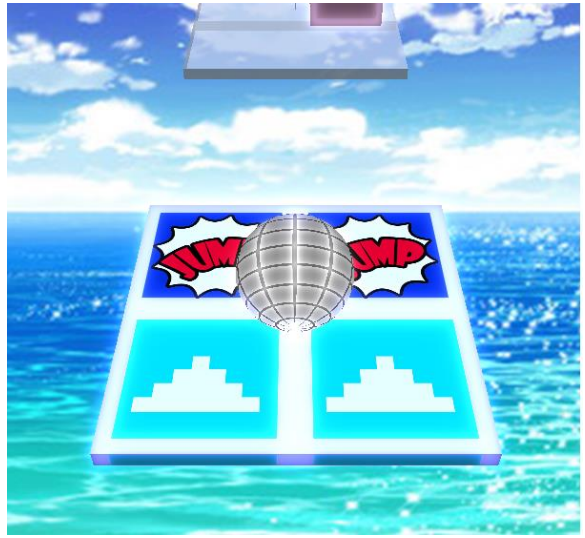
SpriteComponent　MeshComponent

当たり判定やメッシュなど、多くのGameObjectに搭載せざるを得ない機能をそれぞれのComponentに持たせ、GameObjectクラス内で生成するだけでその機能を利用することができる。　　New MeshComponent(this);

ゲーム提出作品　プログラム(頻繁に使用した便利クラス)

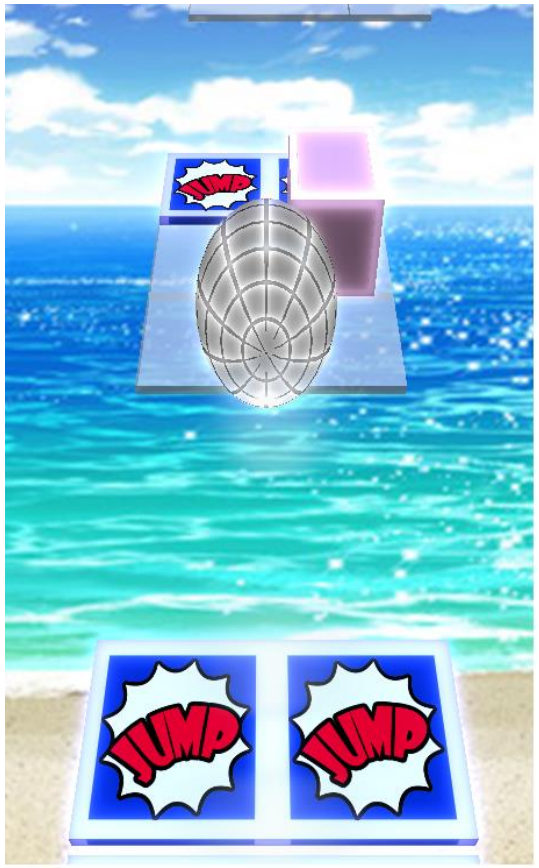
ColliderComponent、BoxCollider、SphereCollider
衝突を行うオブジェクトに搭載するComponent

親GameObjectと衝突している相手との接触状態を管理
接触したら、GameObjectのリアクション関数を呼び出す



Playerが　接触した　→
JumpGround

↓リアクション関数を呼び出し、ジャンプ



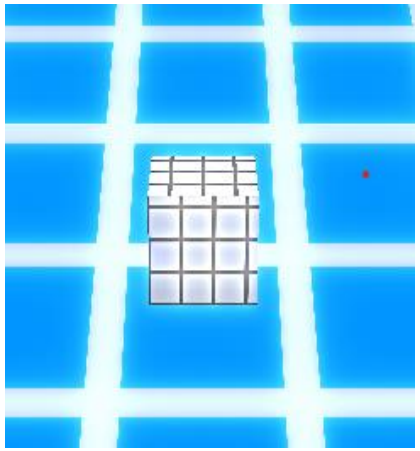
ゲーム提出作品　プログラム(頻繁に使用した便利クラス続)

MeshComponent

メッシュデータの管理と描画を行うComponent

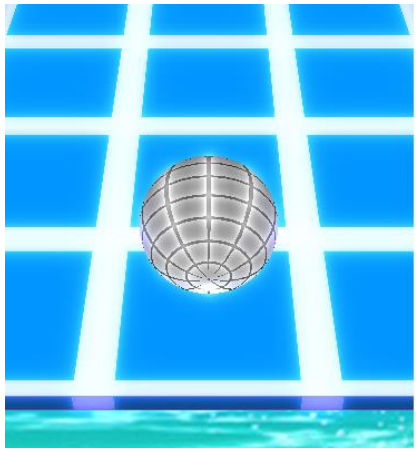
親GameObject内でメッシュデータを管理するクラスを生成し、描画管理クラス内の
メッシュデータ読み込み関数を利用してメッシュデータを設定(.gpmesh)

メッシュデータ(box.gpmesh)



メッシュデータを　→
球状のものに変更

メッシュデータ(sphere.gpmesh)



ゲーム提出作品　プログラム(設計してよかったクラス)

全てのエフェクトの基底となるParticleEffectBaseクラス

生存時間　透明度　加速度　パーティクルの状態　パーティクルコンポーネント

ゴール演出　ゲームオーバー演出　チェックポイント演出　ボールの足元の砂埃

GameObject(基底クラス)から派生したParticleEffectBaseを用意

各パーティクルオブジェクトでParticleEffectBaseを継承することによって再利用性を高めた

GameObjectクラス　→　ParticleEffectBaseクラス　→　GameOverParticleObjectクラス

(親クラス)　派生　(子クラス)　派生　(孫クラス)

GoalParticleObjectクラス

↗　(孫クラス)

派生

↘

CheckPointParticleObjectクラス

(孫クラス)

ゲーム提出作品　プログラム(設計してよかったクラス続)

SceneBase

各シーンの共通の関数や変数を持った、シーンの基底クラス

下記の2点を学習し、応用して容易にシーンの遷移を行うことができた　　↓シーン遷移処理

・派生クラスのポインタを基底クラスのポインタに(暗黙的に)型変換ができるアップキャスト

・派生クラスで挙動を変更できるメンバ関数の仮想関数とオーバーライド

派生クラスのポインタを返還

```
SceneBase* TutorialScene::Update()
{
    if (mPlayer->GetClearFlag())
    {
        mNextSceneCount++;

        if (mNextSceneCount >= 80)
        {
            return new Stage01Scene(stage01);
        }
    }
```

```
SceneBase* tmpScene;

// 実行中のシーンを更新処理
tmpScene = mNowScene->Update();

// シーンの切り替えが発生した？
if (tmpScene != mNowScene)
{
    // 現在のシーンの解放
    delete mNowScene;

    // 現在実行中のシーンの切り替え
    mNowScene = tmpScene;
    continue;
}
```

ゲーム提出作品　プログラム(効率化)

プリコンパイル済みヘッダー

ビルドに2分ほどかかっていたので、短縮できる方法がないか模索し、

Visual Studioにコンパイルを高速化する機能があることを発見

メリット

・ビルド時間が約2分から約5秒に変化したのでデバッグが快適に行えるようになった

・毎回、追加したソースファイルの中で使いたいクラスを何行もインクルードしていたが
　pch.hをインクルードするだけで良いので、1行で済んだ

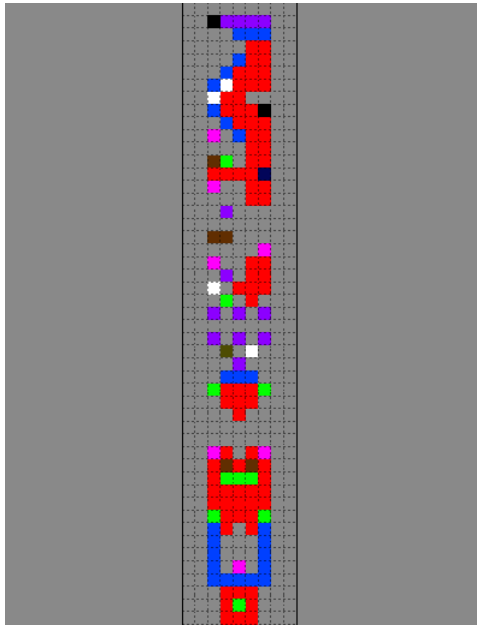
デメリット

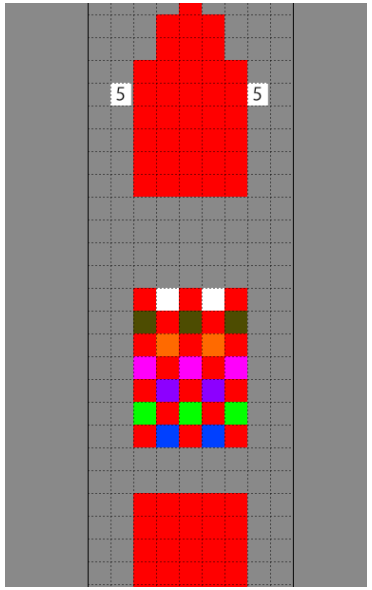
・循環参照を起こす温床になる(ヘッダーファイルを慎重に扱わないと危険)

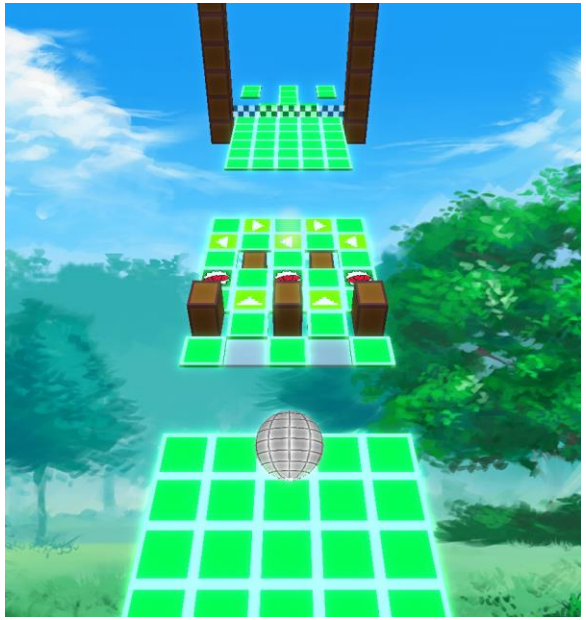
ゲーム提出作品　プログラム(外部ファイルから読み込み)

外部アプリケーション「Tiled」を使用し、
作成したJsonファイルを読み込みステージの作成を高速化

実際に使用したデータ



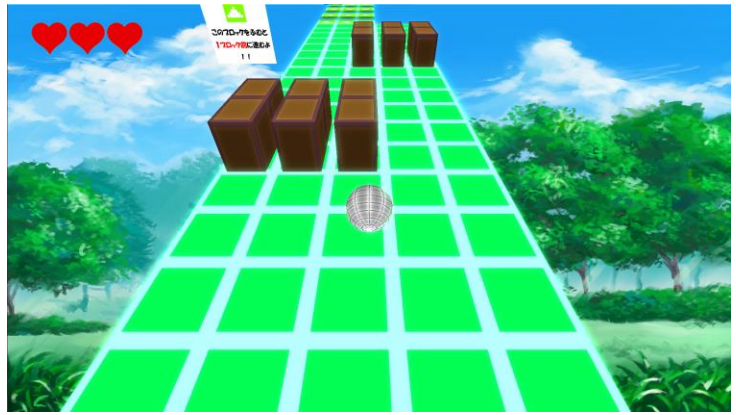




ゲーム提出作品　デザイン(ステージ)

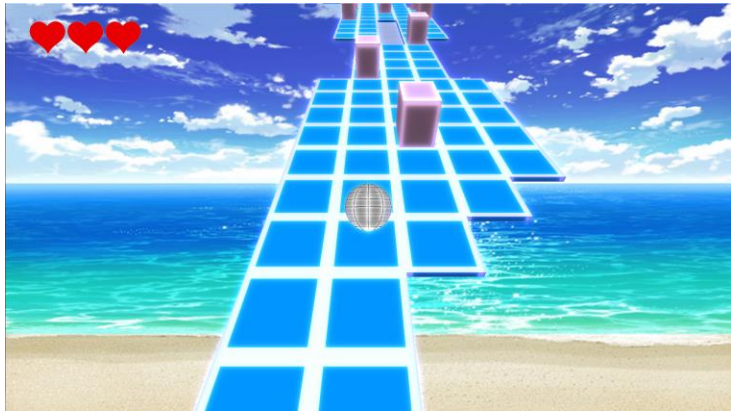
今回制作したゲームは、レベルデザインでかなり難易度が左右されるので
マップの構成はかなりこだわった

1ステージ目



全体的に緑色の配色
簡単なイメージを印象付ける

2ステージ目



全体的に青色の配色
ステージを長くしてチェックポイントを追加

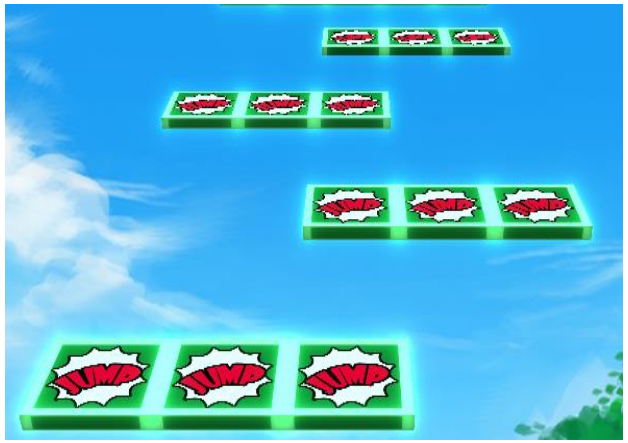
3ステージ目

全体的に紫色の配色
移動する障害物を用意し、難易度が格段にUP

ゲーム提出作品　デザイン(特殊な床ブロック)

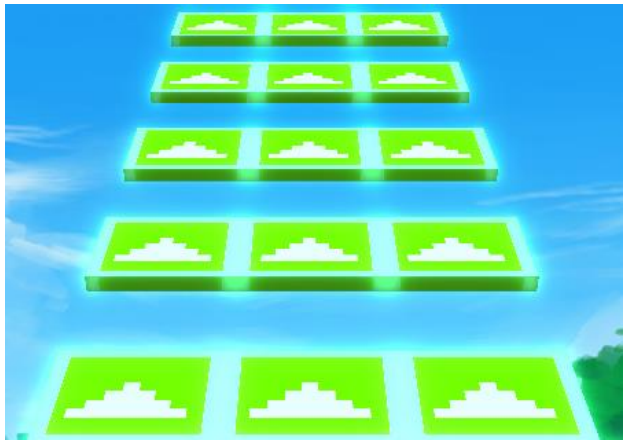
このゲームはスピーディーかつ反射神経が求められるので、プレイヤーがどういった動きをするオブジェクトなのか、瞬時に理解できるデザインにした

ジャンプ床



「ジャンプができる」と理解させるために
弾けるようなデザインに

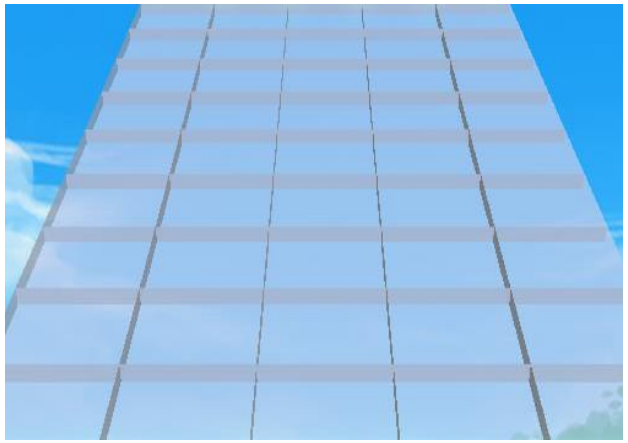
移動床



「前方に進む床」と理解させるために
矢印三角デザインに

他にも、何が起こるか分からないオブジェクトを混同させ、難易度にアレンジを多少加える↓

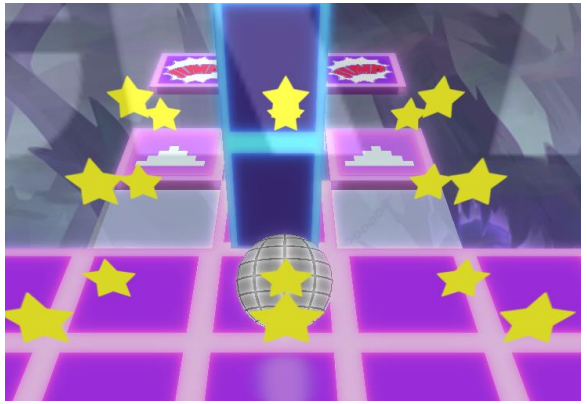
ガラス床



ゲーム提出作品　デザイン(エフェクト)

手触り感を良くしてプレイヤーが飽きないよう、目を引くポップな演出を実装

チェックポイントエフェクト

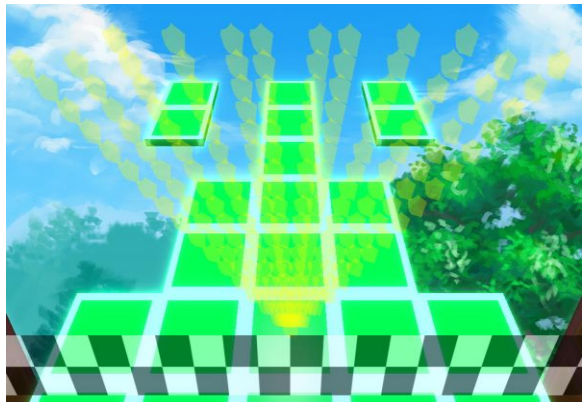


チェックポイントを通過して残機を1つ失うと
通過したチェックポイントからリスタートします
※チェックポイントは3つ

ゲームオーバーエフェクト

ボールが弾けるようなエフェクト

ゴールエフェクト

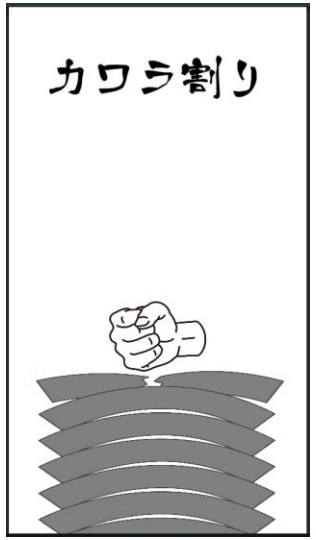


クラッカーをイメージ

制作実績

作品名　カワラ割り
ジャンル　2Dミニゲーム
開発環境　Unity、C#
対応機種　スマホ全般
制作人数　1人
制作期間　2週間
制作時期　2021年3月
担当　イラスト以外全て

学習内容
・スケジュール管理
・プルリクエストによるコメントや設計を意識したコーディング
・複数の解像度によるUI設計
・タッチパネル操作



作品名　ぶらぶらのおぼえてぶらっと福笑い
ジャンル　2Dミニゲーム
開発環境　Unity、C#
対応機種　PC
制作人数　5 人
制作期間　2ヶ月
制作時期　2020年12月～2021年1月
担当　フェードインフェードアウト、スコア処理、プログラム全般サポート

学習内容
・ チーム制作経験
・チームでのGit利用
・Unity制作の基礎知識

制作実績

作品名　BAWA
ジャンル　2Dアクションシューティングゲーム
開発環境　DXライブラリ、C++
対応機種　PC
制作人数　1人
制作期間　3ヶ月
制作時期　2019年12月～2020年2月

作品名　ドットドットドットシューティング
ジャンル　2Dシューティングゲーム
開発環境　DXライブラリ、C言語
対応機種　PC
制作人数　1人
制作期間　3ヶ月
制作時期　2019年7月～2019年9月

作品名　なし(制作中)
ジャンル　3Dアクションゲーム
開発環境　OpenGL、C++
対応機種　PC
制作人数　1人
制作期間　1ヶ月
制作時期　2021年4月

今後の目標

- 現在制作中の3Dアクションゲームを完成させる
- Unity、Unreal Engineなどを用いて1つの3Dゲームを制作する
- シャドウマップ、法線マップ、Bloomなどグラフィック面に関する技術を勉強する
- コードを他人に見せることを意識し、コメントや変数名を丁寧に書く
- 技術ブログやゲーム記事を読む
- たくさんゲームを遊んだり、アニメ、漫画、映画を見る
  (純粋に楽しみ、一方でゲームに組み込むための知識を増やす)

ご覧いただきありがとうございました