

ポートフォリオ

- GitHub: <https://github.com/Furuki0324>
- Wantedly: https://www.wantedly.com/id/taiga_furuki

目次

1. 「百面夜行」～チームでのゲーム制作～

P3～P6

制作時期：2021年9月～2021年12月

使用した技術、言語：Unity、C#、GitHub

GitHub: https://github.com/Furuki0324/Hyakumenyakou_ZemiA

2. ボディトラッキングで操作するゲームの開発

P7

制作時期：2022年1月～2022年2月

使用した技術、言語：Direct2D、[MediaPipe](#)、ソケット通信(UDP)、C++、Python

GitHub(ゲーム部分): <https://github.com/Furuki0324/SimpleUDP>

GitHub(MediaPipe部分): https://github.com/Furuki0324/MediaPipe_py

3. RPG開発に使用するツールの開発

P8～P13

制作時期：2022年4月～

使用した技術、言語：VBA、C#

GitHub(作成中のゲーム): <https://github.com/Furuki0324/PCHLearn>

GitHub(ツール): <https://github.com/Furuki0324/AppForMapchip>

「百面夜行」 ～チームでのゲーム制作～

GitHub: https://github.com/Furuki0324/Hyakumenyakou_ZemiA

2021/09 ~ 2021/12

- 概要

友達の1人がゲームのアイデアを提案し、それに賛同した6人のチームでゲームを開発しました。

タワーディフェンスがベースとなっており、鼻を破壊しようとする敵を撃破しつつ、福笑いのように好きな位置に目や耳、口を配置し、ゲーム終了時のそれぞれのパーツの位置によってスコアが変わります。

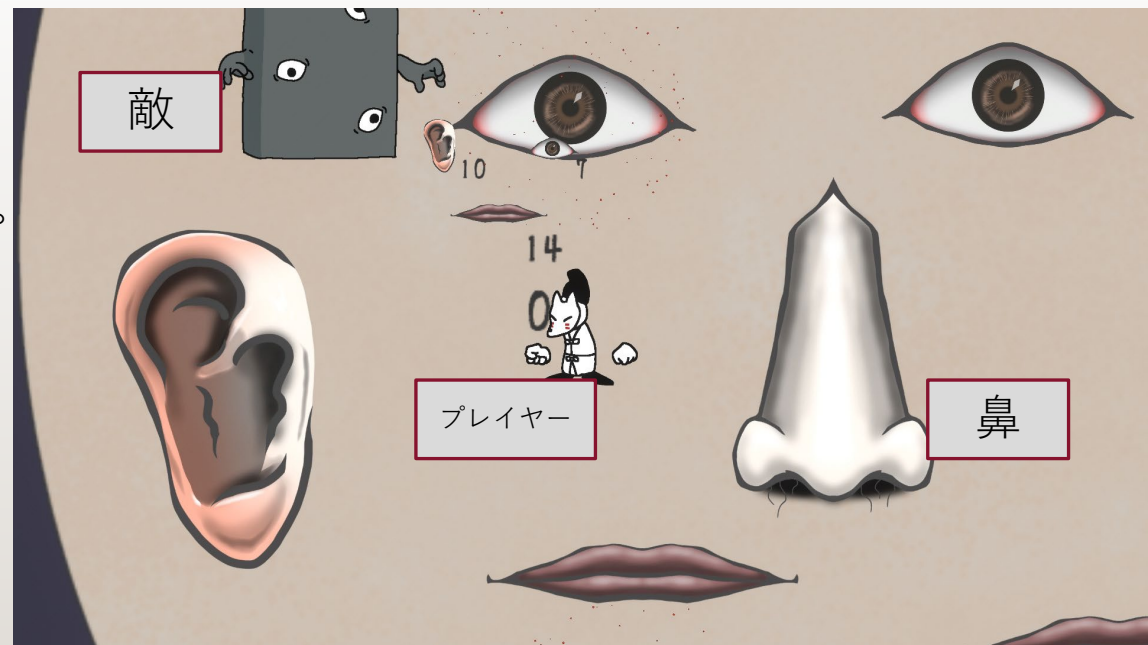
- メンバー構成

プランナー 3 人、プログラマー（私含め） 2 人、デザイナー 1 人

- 参考リンク

プランナーの人たちが作成したPV

参考プレイ動画



↑ プレイ中の画面

「百面夜行」 ～チームでのゲーム制作～

- 制作スタート時の問題点

メンバーが集まって制作を始めることになった際に以下の重大な問題が発覚しました。

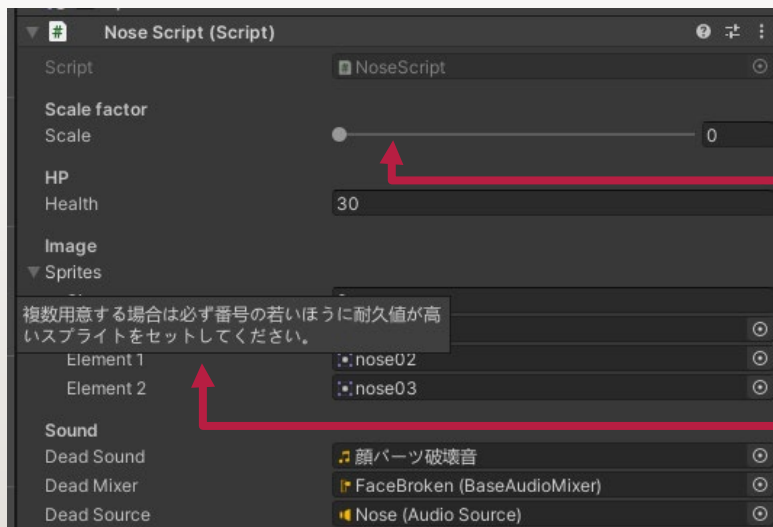
1. ゲームエンジンに**Unity**を使用する話だったが、**Unity**の使い方を知っているのは私ともう 1 人のプログラマーだけだった
2. 共同開発であるため**GitHub**を使ってプロジェクトデータの共有をしようと提案したが、こちらも使い方を知っていたのは私ともう 1 人のプログラマーだけだった

この問題を解決するために、ゲームには直接関係しない部分で工夫をしていました。

「百面夜行」 ～チームでのゲーム制作～

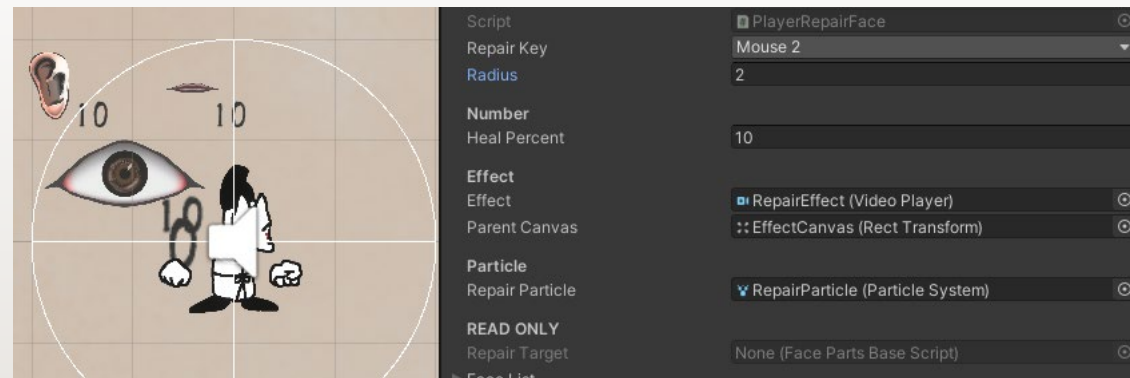
- Unityの使い方に関する問題の解決方法

ゲームの細かな調整はプランナーの担当だったため、Unityの詳しい使い方を知らなくても作業を簡単に出来る環境を目指しました。



入力できる数値の許容範囲があるものはスライダーを使用して入力ミスを防いでいます。

変数の説明を表示することで情報伝達の効率化を図っています。



プレイヤーの周囲のパーツを修理する機能があるため、範囲を可視化することで調整を容易にしています。

この制作活動の以前は個人で制作していたため「動けばそれでよし」と考えている部分がありましたが、チームで活動するためには何に注意するべきか考えるきっかけになりました。

「百面夜行」 ～チームでのゲーム制作～

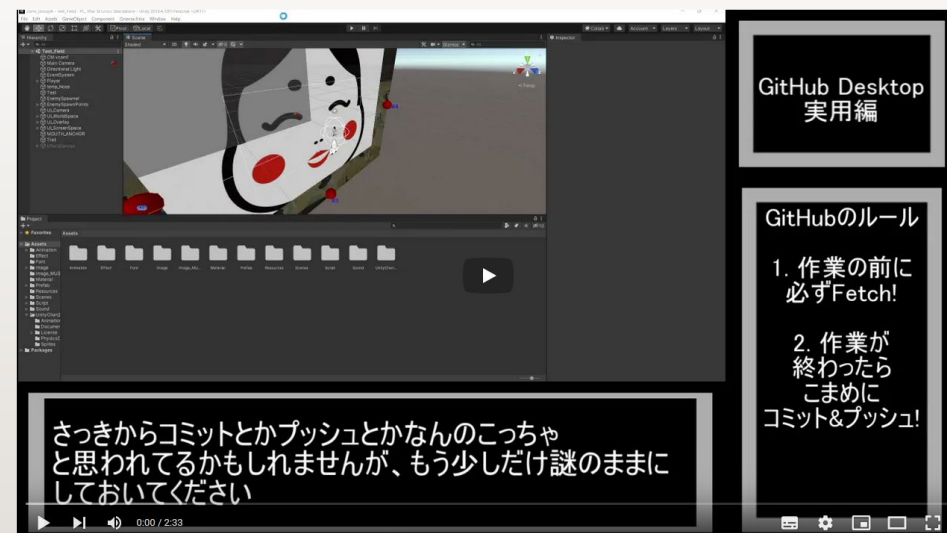
- **GitHub**の使い方に関する問題の解決方法

GitHubの使い方は動画の方が伝わりやすいと考えたため、動画を作成してチームに共有しました。



アカウント作成～インストールまで

Ctrlキーを押しながら画像クリックでYouTubeに移動します。



実際にプロジェクトで使用する際の説明

ボディトラッキングで操作するゲーム

GitHub(ゲーム部分): <https://github.com/Furuki0324/SimpleUDP>

GitHub(MediaPipe): https://github.com/Furuki0324/MediaPipe_py

2022/01 ~ 2022/02

- 概要

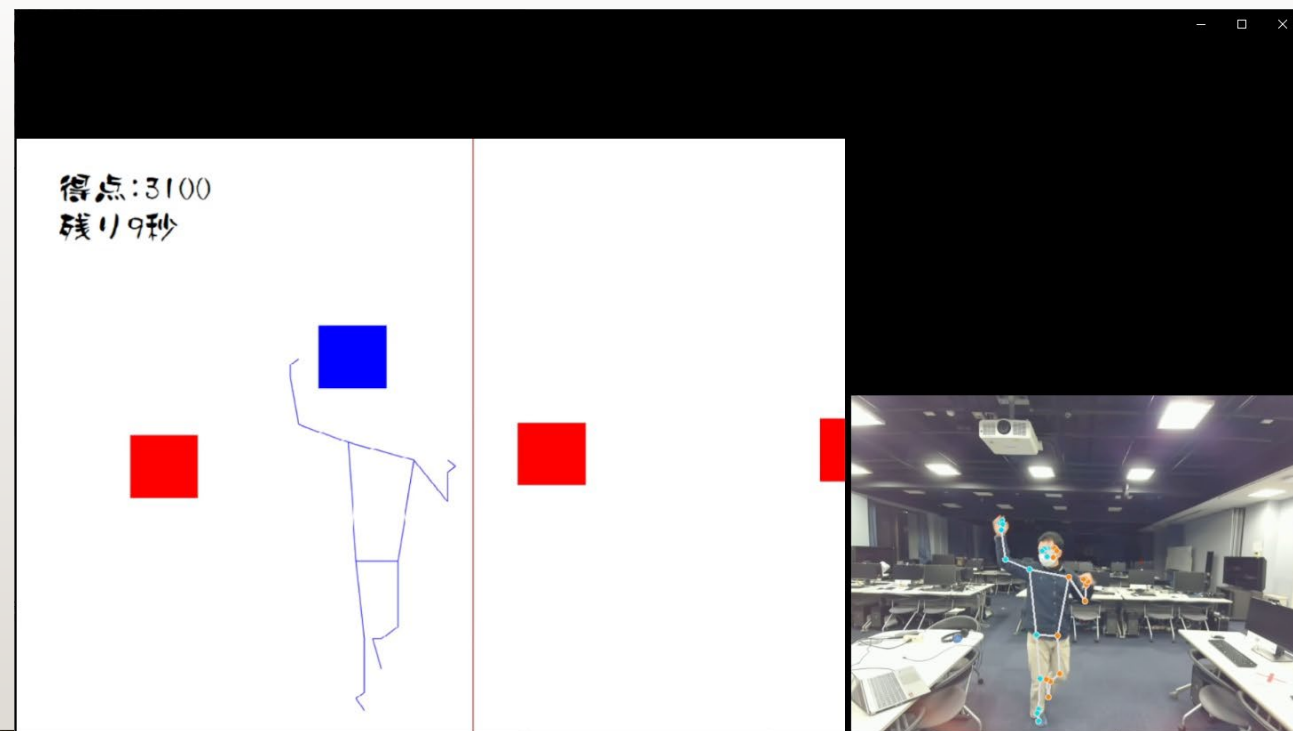
YouTubeでMediaPipeというフレームワークの解説動画を見たことをきっかけに、この技術を使用したゲームを作ることになりました。またゲームエンジンを使わないゲーム開発にも興味があったので、そちらにも挑戦した作品です。

- 使用した技術

Direct2D

MediaPipe <https://google.github.io/mediapipe/>
ソケット通信(UDP)

<https://youtu.be/JlCmLMhPYVY>



RPG開発に使用するツール開発 1

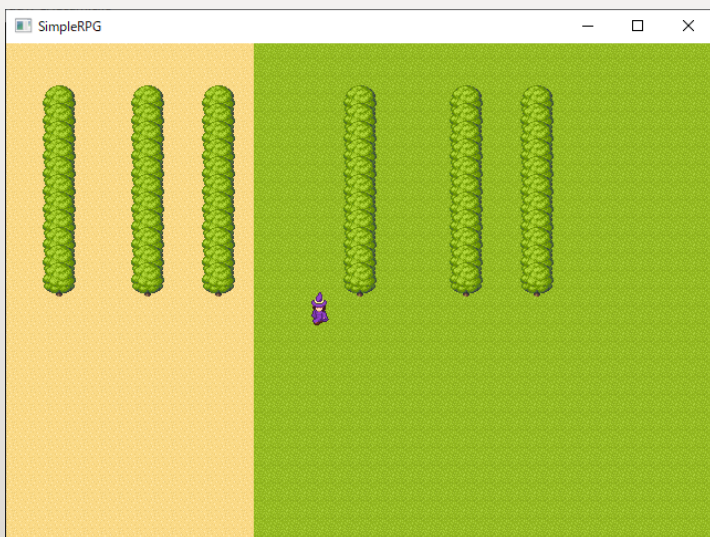
GitHub(作成中のゲーム): <https://github.com/Furuki0324/PCHLearn>

GitHub(ツール): <https://github.com/Furuki0324/AppForMapchip>

2022/04 ~

- 概要

プログラミングの勉強の為に**C++**で開発していた**2D**ゲームの支援ツールです。マップを**CSV**ファイルで管理していましたが、**CSV**ファイルの番号指定で時間がかかっていたため時間短縮のためのツールを作ることになりました。



【左の画像】
ゲームを実行できる
段階まで開発した際
の画面です。

【右の画像】
使用したマップチップのセットの一部です。
16x16ピクセルのチップがまとめられて
128x4672ピクセルになっています。



RPG開発に使用するツール開発 2 - 1

- 初期段階

VBAを使ってエクセルにマクロを作成しました。

必要な項目を設定して実行
(GO) するとアクティブセルに番号が入力されます。

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0					0
49	0					0
65	139					0
65	0					0
65	0					0
65	0					0
65	0					0
65	0					0
65	0					0
65	0	0	0	64	65	0
65	0	0	0	64	65	0

QuerySpriteNumber

×

オブジェクトの種類
崖

その中での行番号(0から)
2

その中での列番号(0から7)
3

☐ フォームを閉じる

GO

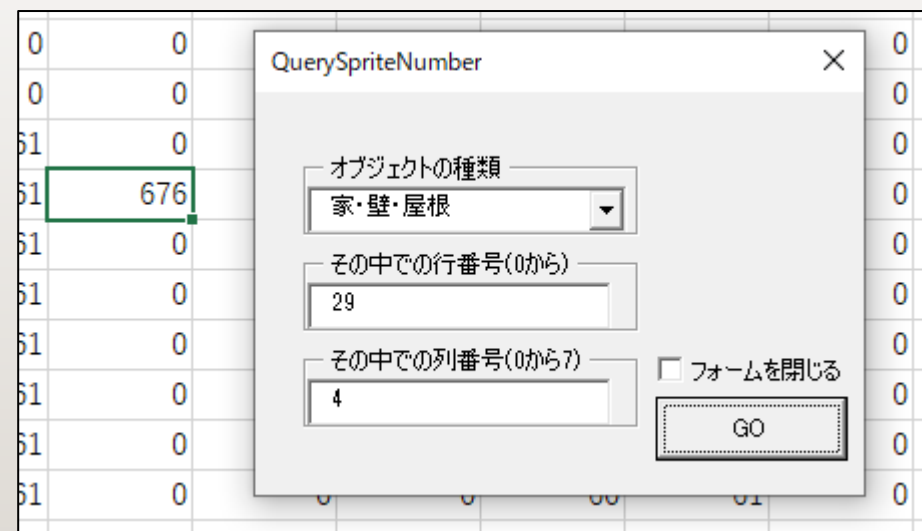
RPG開発に使用するツール開発 2-2

- 問題点

直接計算して入力するよりは簡単に入力できるものの、画像2のように

「家・壁・屋根」グループ内に多くのチップが含まれているときに行番号が大きくなってしまう、時間がかかったり数え間違いの原因になったりしてしまう。

↓ 画像1



→ 画像2



RPG開発に使用するツール開発 3

- VBAのマクロから改善

先ほどの問題を解決するために、**C#**を利用してアプリケーションを作成しました。

下部にチップのサイズを入力してから画像の使用したい部分をクリックすると、

クリックした部分の番号が右下に表示されます。



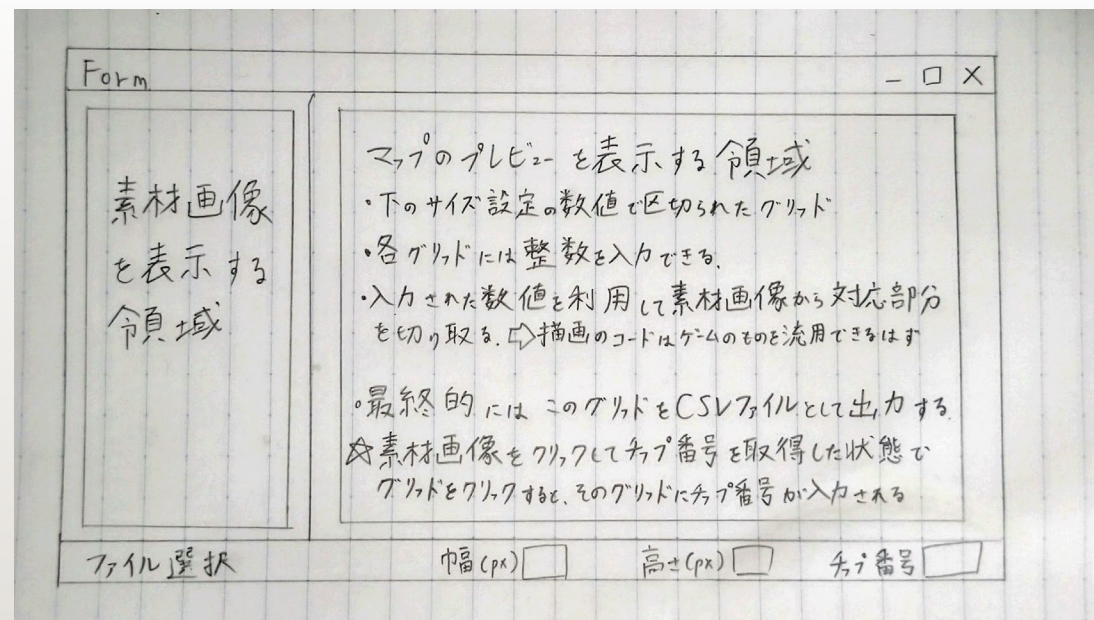
RPG開発に使用するツール開発 4

- 更なる改善案

C#アプリにしたことで数え間違いはなくなったが、エクセルを開いて数値を入力する手間が増えてしまいました。

また今までは編集したマップを確認するためにゲームを起動する必要もあったため、チェックに時間がかかっていました。

これらを解決するために先ほどのC#アプリにCSVファイルの出力機能やプレビュー機能を持たせたツールを考え、開発しています。



1 1 ページのコード

```
private void OnImageClick(object sender, EventArgs e)
{
    //マウスと画像の原点のスクリーン上の座標を取得
    Point mousePoint = MousePosition;
    Point pictureOrigin = PointToScreen(pictureBox1.Location);

    //取得した座標からマウスが画像上のどの位置にいるのか求める
    int x = mousePoint.X - pictureOrigin.X;
    int y = mousePoint.Y - pictureOrigin.Y;

    int chipWidth;
    int chipHeight;
    int spritePerRow;

    //テキストボックスに後の計算が正常に実行されない値が入力されていた場合は中断する
    if (WidthPixel.TextLength == 0
        || WidthPixel.Text == "0"
        || !int.TryParse(WidthPixel.Text, out chipWidth)
        || HeightPixel.TextLength == 0
        || HeightPixel.Text == "0"
        || !int.TryParse(HeightPixel.Text, out chipHeight))
    {
        SpriteNumber.Text = "NaN";
        return;
    }

    //画像の1行の中に何枚のチップがあるか求める
    spritePerRow = pictureBox1.Width / chipWidth;

    //クリックしたチップの番号を求める
    int spriteNum = (y / chipHeight) * spritePerRow + x / chipWidth;
    SpriteNumber.Text = spriteNum.ToString();
}
```