**本ゲーム「鬼ハ追イカケ君ハ逃ゲル」で使ったUnityのバージョンは2019.4.1f1である。**

**2020/08/31 Google Playに申請**

**09/02 1:00時点でGoogle Play Consoleで「公開待ち」になっていた**

**09/02 3:00 App Storeに申請**

**09/03 2:04 App Storeで”In Review”の状態になった**

**09/03 2:22 App Storeで審査合格した**

**09/03 14:00 すでにApp Storeで公開されていた**

**総作業時間 516時間31分(2020/06/22～2020/09/02)**

**2020/06/22**

【Blender】

Ctrl+Jで選択中のオブジェクトを結合できる。

Ctrl+Alt+0でカメラを現在の視点にする

**2020/06/23**

【Unity】

ライティングの設定をするときに参考にしたサイト:

「UnityでLightMapをもっと綺麗にBakeする！」<https://minafrancesca.fanbox.cc/posts/407026>

「ライトマップのベイク時間を短時間で終わらせる」<http://tsubakit1.hateblo.jp/entry/2015/11/19/004627>

とりあえず”~Rendering”や”~Samples”という名前がついている数値を小さくしたら速くなった。

ベイクにかかる時間を短縮するにはWindow→Rendering→Lighting SettingsにあるLightmapping Settings→Lightmapperを”Progressive GPU(Preview)”にする。(参考: <http://tsubakit1.hateblo.jp/entry/2018/09/28/231808>)

**2020/06/24**

【C#】

実機とエディタの両方でタッチ処理ができるようにする方法: <https://techblog.gracetory.co.jp/entry/2018/06/04/000000>

if (Application.isEditor) {

エディタでやる処理

}

else {

実機でやる処理

}

という感じ。

フリック操作を実装した。画面から手を離した瞬間にChangeDirectionを呼ぶのがポイント。

private Vector2 touchStartPos, touchEndPos; //タッチを始めたときの座標、タッチが終わった時の座標

private void GetFlickInfo() {

/\*エディタ\*/

if (Application.isEditor) {

/\*クリックした瞬間\*/

if (Input.GetMouseButtonDown(0)) {

touchStartPos = new Vector2(Input.mousePosition.x, Input.mousePosition.y);

}

/\*クリックが終わった瞬間\*/

if (Input.GetMouseButtonUp(0)) {

touchEndPos = new Vector2(Input.mousePosition.x, Input.mousePosition.y);

ChangeDirection();

}

}

/\*実機\*/

else {

if (Input.touchCount == 1) { //タッチしている箇所は1箇所とする

Touch touch = Input.GetTouch(0);

/\*画面を押した瞬間\*/

if (touch.phase == TouchPhase.Began) {

touchStartPos = new Vector2(Input.mousePosition.x, Input.mousePosition.y);

}

/\*画面から指を離した瞬間\*/

if (touch.phase == TouchPhase.Ended) {

touchEndPos = new Vector2(Input.mousePosition.x, Input.mousePosition.y);

ChangeDirection();

}

}

}

}

private void ChangeDirection() {

float movingAmountX = touchEndPos.x - touchStartPos.x; // x軸方向の移動量

float movingAmountY = touchEndPos.y - touchStartPos.y; // y軸方向の移動量

if (Mathf.Abs(movingAmountX) >= Mathf.Abs(movingAmountY)) {

if (movingAmountX >= 0) {

**右を向かせる処理**

}

else {

**左を向かせる処理**

}

}

else if (Mathf.Abs(movingAmountX) < Mathf.Abs(movingAmountY)) {

if (movingAmountY >= 0) {

**前に進む処理**

}

else {

**後ろを向かせる処理**

}

}

}

}

switch文の使い方

switch (e) {

case e1:

e == e1のときに行う処理

break;

case e2:

e == e2のときに行う処理

break;

}

列挙型の使い方

private enum Direction {PositiveX, NegativeX, PositiveZ, NegativeZ}

private Direction currentDirection;

switch (currentDirection) {

case Direction.PositiveX:

currentDirection = Direction.NegativeZ;

break;

case Direction.NegativeX:

currentDirection = Direction.PositiveZ;

break;

case Direction.PositiveZ:

currentDirection = Direction.PositiveX;

break;

case Direction.NegativeZ:

currentDirection = Direction.NegativeX;

break;

}

という感じに使う。

構造体の使い方

private struct Point //現在プレイヤーがいるマスの座標

{

public int x, y, z; //この座標はUnityの座標系に準拠する

}

private Point point = new Point(); //インスタンスを生成

if (0 <= point.x && point.x <= 13) {

/\*例外が出ないようにif分を入れ子にした\*/

if (s.squares[point.x + 1, point.y, point.z]) {

point.x++;

StartCoroutine(Proceed(new Vector3(1f, 0f, 0f)));

}

}

カメラを回転させるのにはtransform.Rotate(new Vector3(○, ○, ○))を使った

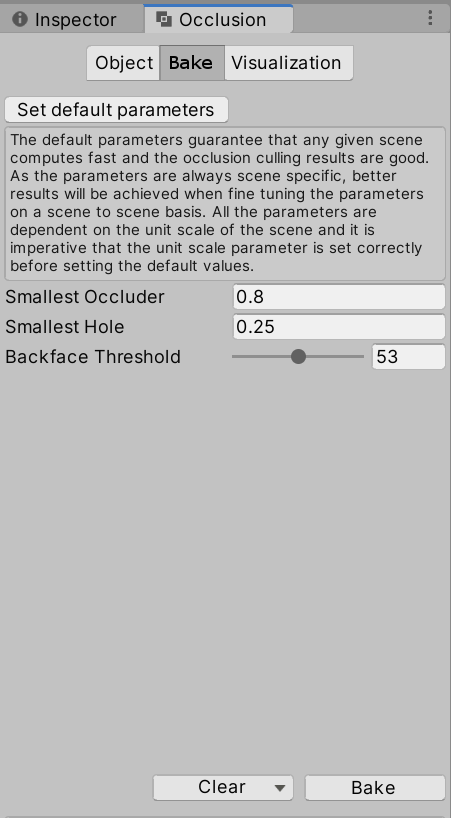
2020/06/25

【C#】

transform.Rotate()でオブジェクトを回転させるときは第2引数でSpace.Worldと入れると、ワールド座標系で回転させられる。

2020/06/26

【Unity】



Staticなオブジェクトに関して、カメラに映らないオブジェクトを描画しない、さらに遮蔽物の後ろに隠れて見えないオブジェクトを描画しないようにするには、Occulusion Cullingという機能を使う。Occulusion Cullingを適用させたいオブジェクトをヒエラルキーで選択した状態でWindow→Rendering→Occulusion Cullingを押す。そうすると、Inspectorの右側に”Occulusion”のタブが出現し、その中に”Object”, “Bake”, “Visualization”という3つのタブがある。”Bake”を押し、Smallest Occuluder, Smallest Hole, Backface Threshouldの値を設定し、下の方にある”Bake”ボタンを押してベイクする。そして、”Visualization”タブを押した状態でゲームを実行すると、Occulusion Cullingが有効になった状態でプレイできる。

二重継承にしなくても、スーパークラスがMonoBehaviourを継承していたら、サブクラスもMonoBehaviourを継承していることになるので問題ないことに気づいた。

CreatureController.csというスーパークラスを作り、そのサブクラスとしてPlayerController, OgreControllerを実装しようと思ったが、継承の実装が難しいので後回しにすることにした。もっと簡単な例で実装してから立ち返ってみるのがいいかもしれない。

今日やりたかった実装

Walk, ClimbUpLadder, ClimbDownLadderというメソッドとProceed, Rotate, ClimbUp, ClimbDownというコルーチンをスーパークラスであるCreatureController.csのみに入れて、サブクラスであるPlayerController.csにはGetFlickInfo, ChangeDirectionというメソッドのみを残すつもりでいた。

**2020/06/27**

**【**C#**】**

メソッド内メソッドを使うと便利な場面があった。Aというメソッドの中でhogeという処理を何度も行うが、Aメソッド以外ではhogeという処理を一切行わないという場合に、下のように書ける。

private void A () {

hoge();

…

hoge();

…

void hoge() {

処理;

}

}

このとき、

* hogeにはアクセス修飾子をつけることができない

ということに注意するべし。

**2020/06/28**

【Photoshop】

画面の端に小さなマップを配置し、そこにプレイヤーの現在位置を表示する。小さなマップをタッチすると画面全体にマップ全体が表示されるように実装したい。そこで、Photoshopでマップの素材を作っていきたい。

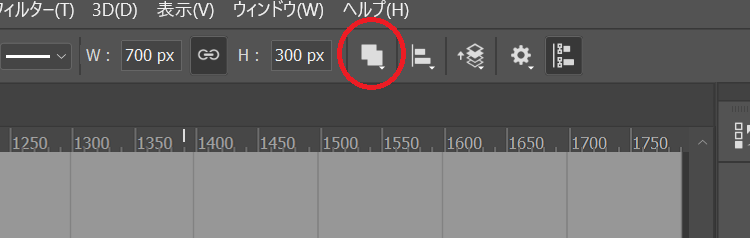
グリッドを表示するには、”表示”→”表示・非表示”→”グリッド”を選択する。

座標を指定して長方形を配置するのは長方形の作成時にもできる。作成後に長方形の辺の長さや中心座標を変更したいのなら、”編集”→”パスの自動変形”を選択すればできる。

グリッドの間隔は、“編集”→”環境設定”→”ガイド・グリッド・スライス”から設定できる。

グリッドを表示している状態で長方形を作成すると、頂点がグリッドの交差点に来るように自動でポインタを位置調整してくれる。

1. 結合させたい長方形が含まれているレイヤーをすべて選択した状態で、”レイヤー”→”シェイプを結合”を選択する。
2. 結合されたシェイプが含まれているレイヤーを選択した状態で、”シェイプコンポーネントを結合”を押す。(下図の赤丸を選択すると出てくる)



**2020/06/29**

【Photoshop】

“選択範囲”→”色域指定”からスポイトで指定した色の部分を選択できる。その選択範囲をすべて消すためには、”編集”→”塗りつぶし”で”内容”を”描画色”、”描画モード”を”消去”に設定すればいい。

選択範囲の色相・彩度・明度を変化させるには、”イメージ”→”色調補正”→”色調・彩度”から調整する。

【Unity】

ミニマップを作るときに参考にした動画: <https://www.youtube.com/watch?v=kWhOMJMihC0>

WholemapCameraで上空からステージを見下ろしたときに、透視投影だとプレイヤーと鬼のアイコンがカメラの視界の中央にあるときしか見えなかったが、平行投影にしたら見えるようになった。

**2020/07/02**

【C#】

列挙型の宣言はクラス外で行うといいかもしれない。

public enum Direction { PositiveX, PositiveZ, NegativeX, NegativeZ }

構造体の宣言もクラス外で行うといいかもしれない。

public struct Point {

public int x, y, z;

public Point (int x, int y, int z) { //コンストラクタ

this.x = x;

this.y = y;

this.z = z;

}

}

**2020/07/03**

【C#】

あるクラスで宣言した変数を他のクラスから変更できないようにしつつ、かつ読み取りはできるようにする方法

private Point goal = new Point(5, 6, 7);

public Point GOAL {

get { return goal; }

}

プロパティのsetを省略することで外部のクラスからgoalの値を変更できないようにする。

<https://qiita.com/ShirakawaMaru/items/5f8bd065b1a4d2fdc8e0>

オブジェクトのオイラー角を取得するには、

int x = gameObject.transform.eulerAngles.x;

などとする。

オブジェクトの名前を取得するには

string name = gameObject.name;

とする。

2020/07/04

【Unity】

シーンを遷移させるときのフェードアウト、フェードインには以下のサイトからダウンロードした素材を使った。

<https://github.com/naichilab/Unity-FadeManager/blob/master/README.ja.md>

**2020/07/05**

【Unity】

実機向けにビルドする際に、実機の画面でゲームを縦向きか横向きで表示させるのかを決めるにはProject Settings→Player→Resolution and Presentationで設定を行う。

**Allowed Orientation for Auto Rotation**

* Portrait: スマホの上部を上にする
* Portrait Upside Down: スマホの下部を上にする
* Landscape Right: スマホのフロントカメラが右側に来る
* Landscape Left: スマホのフロントカメラが左側に来る

また**Default Orientation**では実機での画面をどの向きに固定するかを指定できる。ここをAuto Rotationにしておけば、Allowed Orientation for Auto Rotationでチェックを入れた向きに自動で画面の向きが変わるようになる。Auto Rotation以外に設定しておくと、画面を傾けても表示される向きが変わらなくなる。

実機のアプリのアイコンは

**2020/07/08**

【Unity】

BlenderからUnityにエクスポートした3Dモデルの面がなくなっているのは、その面の法線の向きが逆になっているからである。

ベイクしたときに変な影が落ちていたが、UVマップで面同士が重ならないように面を配置しなおしたら影が消えた。

【Photoshop】

色が白黒になってしまったときは、イメージ→モードで”RGBカラー”を選択すればいいと思う。

一度作った長方形の中心座標を変更するには、編集→パスの自動変形でx, y座標の値を変更すればよい。

**2020/07/09**

【Unity】

Unityから消したオブジェクトはWindowsのごみ箱に入っているので、間違えて消したときはごみ箱から復元しよう。

【C#】

Instantiateで作ったインスタンスに個別にアクセスするには、例えば

var bridges = new List<GameObject>();

bridges.Add(Instantiate(…));

bridges.Add(Instantiate(…));

bridges.Add(Instantiate(…));

として、bridges[1]などとインデックスを指定するといった方法がある。リストの代わりに配列を使ってもできる。

橋が置いてある座標からその橋にアクセスする方法を考えた。まず、橋の座標とゲームオブジェクトを格納する辞書を作って初期化する。

var bridges = new Distionary<Point, GameObject>()

{

{ new Point(4, 3, 6), Instantiate(bridge, new Vector3(4f, 3f, 6f), Quaternion.identity))},

{…},

{…}

};

そのあと座標指定してオブジェクトにアクセスする。

GameObject bridge = bridges[new Point(…)];

(参考にしたサイト: <https://qiita.com/otochan/items/28c3ecf7377ba56187c5>)

テキストや画像の色を変更したいときは

GetComponent<Text>().text.color = new Color(1f, 1f, 1f, 0f);

GetComponent<Image>().color = new Color(0f, 0f, 0f, 0.3f);

とすればよい。(参考にしたサイト: <https://qiita.com/pilkul/items/eb9b4bbbb519772c3397>)

**2020/07/10**

【Unity】

Imageに正しいコードをアタッチしてもOnDragなどのイベントが発生しないときは、そのImageの上に何らかのuGUIが重なっていることがあるので、uGUI同士が重ならないように配置し直すとよいだろう。それか、uGUI同士が重なっても下に埋まっているuGUIを取得できる方法をとる(要調査)。

→ヒエラルキーの順番が大事！ヒエラルキーの下にある方が上のレイヤーに描画される。

【Photoshop】

選択範囲を拡大・縮小させるには、編集→パスの自由変形を選ぶ。

**2020/07/11**

【C#】

OnDropメソッドはドラッグされたものが落とされる側のオブジェクトにつけるメソッドである。

**2020/07/16**

【Unity】

ボタンをグリッド状に並べてさらにスクロールできるようにする方法:

まずスクロールできる部分を作るにはCanvasの子にUI→Scrollbarを新たに作る。するとScroll View/Viewport/Contentが作成されるので、そこにGrid Raycast GroupとContent Size Fitterをアタッチする。あとはパラメータを適当に設定してやればOK。

Grid Layout Group: ”Padding”では上下左右の余白の大きさを決める。Cell Sizeでは並べる対象の大きさを設定する。Spacingでは並べる対象を入れる枠にどれほどの余白を設けるかを決める。Start Axisでは並べる対象の数が増えていったときに水平方向と垂直方向のどちらを優先に対称を配置していくのかを決める。

Content Size Fitter: Preffered Sizeにしておけば、並べる対象の数が増えていったときに自動でContentのサイズを変えてくれる。

並べる対象の数が決まっていないのなら、スクリプトで対象をInstantiateで生成していくとよいだろう。

(参考にしたサイト: <https://odekakeneko.hatenablog.com/entry/2020/01/19/013125>)

**2020/07/18**

【Blender】

選択したオブジェクトを中心にカメラを回転させるには、オブジェクトを選択した状態でShift+S→Cursor to Activeを押して選択しているオブジェクトに3Dカーソルを持って行ったあとにN→View→Lock to 3D Cursorにチェックを入れればよい。

モバイルゲームで頂点数の多いオブジェクトは作らないほうがいい。

**2020/07/19**

【Blender】

BumpかDisplacementで凸凹を表現するのはCycles Renderじゃないとできない。

面を扇状に分割するには面を選択した状態でFace→Poke Facesとする。

Array ModifierでFit TypeをFit LengthにしConstant Offsetにチェックを入れた状態にしてもオブジェクト間の距離がおかしかったが、そのオブジェクトのスケールをApplyしたら解決した。

Planeを分割して別々のPlaneにするにはPlaneをCtrl+Rでいくつかに分割したあと切断する辺を選択してVを押す。

【Unity】

HDR画像をskyboxとして使うときは”Texture Type”をCubeに設定しておく。(参考にしたサイト: <http://unitygeek.hatenablog.com/entry/2017/11/02/171535>)

2020/07/20

【Blender】

ウェイトを自動で設定するのではなく、各ボーンに割り当てるオブジェクトを一つ一つ設定していく方法

1. Object Modeでメッシュを選択
2. Object Modeでボーンを選択
3. Pose Modeで割り当てたいボーンを選択
4. Ctrl+P→Bone

位置的に離れているボーン同士で親子関係を作るには、Edit Modeで子のボーン→親のボーンの順番に選択したあとCtrl+P→Keep Offsetとする。

ボーンを初期位置に戻すにはボーンを選択した状態でAlt+R

**2020/07/21**

【Blender】

左右対象にボーンを配置しているときに、片側のボーンのTransformの情報を左右対象にしてもう片側のボーンに移す方法

1. Pose Modeでボーンを選択してCtrl+C
2. Pose Modeでもう片方のボーンを選択してShift+Ctrl+V(Pose→Paste Pose FlippedでもOK)

ボーンの根本を回転中心にしたいときは、Pose Modeでボーンを選択した状態でShift+S→Cursor to Selectedとする。

メッシュをボーンに割り当てた後に割り当てを変えたくなったときは、Object Modeでメッシュを選択しObject Properties→Relations→Parentを消しておくとよい。

Action Editorで青いバーを特定のフレームに合わせ、さらにAで全キーフレームを選択した状態でSで拡大、縮小すると全キーフレームが青いバーがあるフレームを中心に拡大、縮小される。

アニメーションの再生・停止はSpace

Blenderで設定したオブジェクトのスケールの値がUnityでも同じ値になるようにするには、fbxファイルを出力するときにTransform→Apply Scalingを”FBX ALL”、Forwardを”-Z Forward”、Upを”Y Up”にし、!EXPERIMENTAL! Apply Transformにチェックを入れておく。

(参考にした動画: <https://www.youtube.com/watch?v=LxeMgT4Ox7U&t=454s>)

BlenderからUnityにエクスポートしたオブジェクトの原点が足元になるようにするには、BlenderでオブジェクトのOriginを足元にしておく。

**2020/07/22**

【C#】

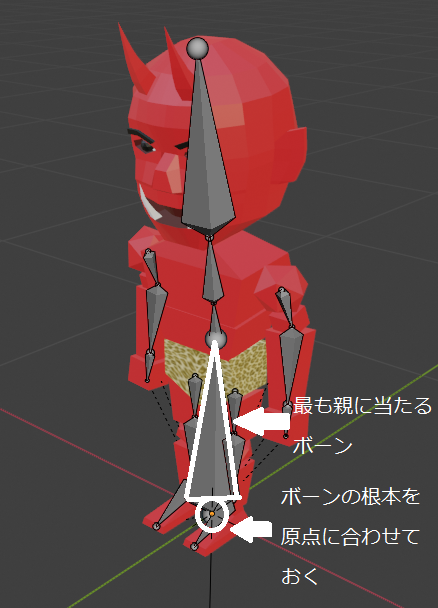
無限に自分自身を呼び続ける関数があったために、StackOverflowExceptionが発生した。

ランダムなbool値を返すスクリプト

bool flag = UnityEngine.Random.Range(0, 2) == 0;

【Blender】

BlenderからUnityにボーンを用いたアニメーション付きのオブジェクトをFBXファイルとしてエクスポートすると、どうやらUnity側でのアニメーションの原点は、最も親にあたるボーンの根本部分になるようである。これを利用して、Blenderで最も親にあたるボーンの根本の位置を、あらかじめUnity側で原点にしたい場所に合わせておけばよいと思う。



ちなみにこの動画(<https://www.youtube.com/watch?v=nNxH2X8-BXY>)で紹介されている方法(空のオブジェクトの位置を原点に合わせておき、そのオブジェクトの子に使いたいオブジェクトを配置するという方法)でオブジェクトの原点の位置を調整しようとしたが、うまくいかなかった。

Unityにエクスポートするためのアニメーションを作る際に、複数のアニメーションを作るには×マークを押して新たなアニメーションを追加していく。(参考にしたサイト: <http://ankohouse.sblo.jp/article/100867645.html>)

【Unity】

Occulusion Cullingを有効にするとき、AnimatorのCulling Modeは基本的に”Cull Completely”でいいと思う。

(参考にしたサイト: <https://docs.unity3d.com/ja/current/ScriptReference/AnimatorCullingMode.html>)

Animatorで特定のアニメーションをある時に一度だけ実行したいなら、Triggerを使うとよい。アニメーション同士をつなぐ矢印の発生条件を入力する欄でTrigger型の変数を指定する。C#のスクリプトで実行するときは、

animator.SetTrigger(“Trigger型の変数の名前”);

とする。(参考にしたサイト: <https://www.lovehatetubaki.work/entry/Animation-Mecanim1>)

あるアニメーションを最後まで実行させてから次のアニメーションに移したいのなら、HasExitTimeにチェックを入れておく。

**2020/07/23**

【C#】

継承を使ったプログラミングで気を付けること

public class OgreController : MonoBehaviour {

//子クラスで使う変数やメソッド、コルーチンにはすべてprotectedというアクセス修飾子をつける

protected bool isLeft; //子クラスで使う変数は宣言だけしておく

void Start () { //子クラスで使う変数の初期化をここでしておく必要はない

}

protected void Initialize(){ //子クラスで使うメソッドやコルーチンの中身はここで書いておく

…

}

}

public class GreenOgreController : OgreController {

void Start () {

isLeft = false; //親クラスで宣言した変数をここで初期化する

}

Initialize(); //親クラスで宣言したメソッドやコルーチンはここでそのまま使える

}

Unityで” The same field name is serialized multiple times in the class or its parent class.”というエラーが出たのは、親クラスと子クラスで同名の変数が宣言されていたためだ。(参考にしたサイト: <https://qiita.com/kyoro_1192/items/eaafb6b30b24db45fd3f>)

親のオブジェクトからこのオブジェクトにアクセスする方法

transform.GetChild(n).gameObject;

ただしnにはヒエラルキーで上から数えて何番目かが入る(一番上が0番目)。

(参考にしたサイト: <https://qiita.com/Armyporoco/items/391776d4c79d25cfbbfe>)

**2020/07/24**

【C#】

StartメソッドとAwakeメソッドを両方使っているときは初期化ができていないせいで何らかのバグが発生することがあるので、注意せよ。

配列をディープコピーする方法: <https://www.sejuku.net/blog/40212>

**2020/07/25**

【Blender】

BlenderからUnityにfbxファイルを持って行ってもrotation値が(0, 0, 0)になるようにするアドオン: <https://light11.hatenadiary.com/entry/2019/01/06/215851>

メッシュに紐づいているアーマチュアのスケールをapplyしたいときは、一回紐づいているのを解いてからCtrl+AでApplyするとよい。

【Unity】

鬼のアニメーションがだんだんと地面に沈んでいくのは、Apply Root Motionのチェックを外したら解決した。

**2020/07/26**

【Blender】

マテリアルのベイクをする

(参考にしたサイト: <https://xr-hub.com/archives/19305>)

**2020/07/28**

【Blender】

メッシュにベイクできなかった理由は、マテリアルにベイクした結果を描きこむ画像を搭載したImage Textureノードを入れていなかったからだった。

余白の値が大きいと隣のメッシュに色が映りこんでしまいやすい。余白の大きさはRender Properties→Bake→Outputから調節できる。

**2020/07/29**

【Unity】

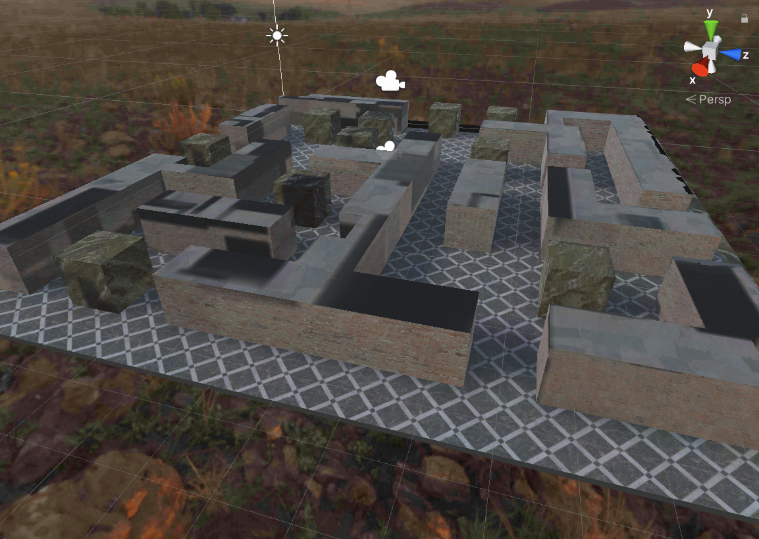
Bakeしてもどうしても影が出ないと思ったら、Directional Lightが地面に直交していたからだった。

Mesh Bakerで作ったマテリアルにベイクしたら変な四角い影がいっぱいできた。

**2020/07/30**

【Unity】

Mesh Bakerを適用すると、もとのステージのLightmapの情報が引き継がれていない。メッシュを統合した後に光の情報をベイクすると奇妙な四角い影がたくさんできる。



→Lightmapの情報を引き継いだままメッシュを統合できないか？

<https://vrcworld.wiki.fc2.com/wiki/Mesh%20Baker>

**2020/08/01**

【C#】

1つのオブジェクトにアタッチされている同一のコンポーネントを区別して取得する方法

private AudioSource[] audioSources;

void Start() {

audioSources = GetComponents<AudioSource>();

}

とすれば、オブジェクトについているAudioSourceのうち一番上にあるものからaudioSource[0], audioSource[1], …で取得できる。

(参考にしたサイト: <https://boccilog.wordpress.com/2015/07/06/%E5%90%8C%E4%B8%80%E3%81%AEcomponent%E3%82%92%E5%88%A5%E3%80%85%E3%81%AB%E5%8F%96%E5%BE%97/>)

**2020/08/02**

【Unity】

音源の距離によって音の大きさを変化させる方法:

AudioSourceコンポーネントのSpatial Blendを1にすると、距離による音量の大きさが最も考慮される。そして、3D Sound Settingを開きMin Distance, Max Distance, 制御点を設定すればOK。

タイトル画面で画面の端で鬼が歩いている様子をループで載せておく

(参考にしたサイト: <https://baba-s.hatenablog.com/entry/2018/03/16/170000>)

**2020/08/04**

【Unity】

Canvas内に動画を入れる方法:

* まずVideo Playerを用意する。Video Clipに再生したい動画を挿入、Render Modeを”Render Texture”、Target Textureに新しく作成したRender Textureを挿入する。
* 次に、Canvasの子としてRaw Imageを作成する。Textureに先ほど作成したRender Textureを挿入すれば動画が再生されるようになる。

再生スピードやループ再生するかどうかはVideo Playerから設定できる。

ボタンの一覧が載っている領域の大きさは、Scroll Viewの大きさを変えるだけでOK。

Grid Layout Groupについて

Cell Size: 一つ一つのセルの大きさを指定する。例えばWidth=100, Height=100のImageを入れようと思ったらx=100, y=100と設定する。

マスクのやり方

1. CanvasにImageを作る
2. 今作ったImageにAdd ComponentからMaskを追加する。
3. Imageの子にマスクされるオブジェクトを配置する。例えばImageやTextなど。
4. ImageのSource Imageにマスク画像を挿入する。ただし、マスク画像の黒い部分だけが見えることになる。
5. MaskのShow Mask Graphicのチェックを外す。

【Clip Studio Paint】

選択範囲の拡大・縮小の中心を選択範囲の中心点にするには、Altを押しながらやればよい。

**2020/08/05**

【Unity】

Grid Layout Groupについて

Constraint: Fixed ~ Countにすると、行数や列数を決まった値にできる。

マスク画像をペイントソフトで作るときは、映したい部分を黒、映したくない部分を透明で塗るべし。

Admob

鬼ハ追イカケ君ハ逃ゲル

**Android**

アプリID: ca-app-pub-5831891803553766~4022954650

以下は広告ユニットIDである。

バナー: ca-app-pub-5831891803553766/5391984444

インタースティシャル: ca-app-pub-5831891803553766/7635004402

リワード: ca-app-pub-5831891803553766/3121044321

admobのバナー広告の位置を見たかったが、どうやらバナー広告はエディタでは見ることができず、実機でないと見られないようである。

参考にしたサイト

YouTubeの動画: <https://www.youtube.com/watch?v=fGMFHN3gXpY>

**2020/08/06**

【Photoshop】

画像の端を古紙のようにギザギザにする方法: <https://liginc.co.jp/designer/archives/2017>

ギザギザの大きさは水晶のセルの大きさで調整できる。

画像に影をつける方法: <https://techacademy.jp/magazine/2924>

【Android】

Androidでビルドするときは

* 端末情報→ビルド番号を7回タッチしてデベロッパーモードにしておく。
* システムと更新→開発者向けオプション→USBデバッグをオンにする。

【Unity】

ボタンの中心ではなく端っこの方を押さないとボタンが反応しないことがあったが、これはボタンよりも上のレイヤーにある何かしらのオブジェクト(Image, Text等)のRaycast Targetがオンになっていたからだと思う。ポインターからのレイを受け取らなくてよいオブジェクトのRaycast Targetはすべてオフにしておいた方がいい。

プレハブにアタッチされているスクリプトにpublic変数を用意しておきインスペクターからシーン内にあるオブジェクトを挿入することはどうやらできないようだ。ドラッグ&ドロップしようとしたが、ドロップできなかった。

1234567890

1234567890

1234567890

1234567890

アイコンはIcooon Monoというサイトがおススメかな。

**2020/08/07**

【Clip Studio Paint】

選択範囲のみを上下反転するには、選択範囲を指定した状態で編集→変形→上下反転とする。

白を色相変化で別の色に変えることはできないようだ。

**2020/08/08**

【Blender】

重なっている頂点を結合するには、Edit Modeで頂点を選択した状態でAlt+M→By Distanceを選ぶ。

【Photoshop】

ステージの一番外側にある柵を作るときに繰り返しのパターンを作る方法が知りたかった。

繰り返しのパターンを作る方法: <https://pronama.jp/2013/02/07/photoshop-pattern-layer/>

【Unity】

Sprite Atlasの使い方

1. ProjectビューでSprite Atlasを新しく作成する。
2. Objects for PackingにまとめたいSpriteを全部詰め込んでいく。(詰め込まれた結果はPack Previewで見られる。)
3. 実際に実行中の画面でSprite Atlasから持ってきたSpriteを表示させるには、以下のコードをImageのオブジェクトにアタッチする。

using UnityEngine;

using UnityEngine.U2D;

using UnityEngine.UI;

public class CircleFrameController : MonoBehaviour

{

[SerializeField]

private SpriteAtlas UISpriteAtlas;

void Start() {

gameObject.GetComponent<Image>().sprite = UISpriteAtlas.GetSprite("frame");

}

}

使用するSpriteAtlasをインスペクターで指定するのと、Sprite名を文字列で指定するのに注意する。

(参考にしたサイト: <https://kan-kikuchi.hatenablog.com/entry/SpriteAtlas>)

Sprite AtlasのPackingに入っているパラメータ

Allow Rotation: それぞれのSpriteを回転してもいいか

Tight Packing: これにチェックが入っていると、環状のSpriteの中に別のSpriteが入り込んでしまうことがある。

Padding: 余白の大きさ

Sprite AtlasのPack Previewが表示されない問題は、このサイトを見て解決した。→<https://stackoverflow.com/questions/48456965/unity-sprite-packer-is-disabled>

【C#】

アプリを閉じてもデータを残すには

PlayerPrefs.SetInt(“Course1”, 1);

などとする。データを読み取るには

PlayerPrefs.GetInt(“Course1”, 0);

とする。キーが存在しない場合は第2引数に指定されている値を返す。

50個のコースのクリア履歴はこのサイト(<https://teratail.com/questions/171451>)を参考にして記録した。

**2020/08/09**

【Unity】

cameraを平行投影にすると”size”という値が出てくるが、これは視界の広さを表している。

アルファ値が0のUIもバッチ数を稼ぐので、表示していないUIはアルファ値を0にするのではなく非アクティブにするのがよい。

Mesh Bakerでただのシーンオブジェクトをベイクする手順

まず初めに、Mesh Bakerを使うにはGameObject→Create Other→Mesh Bakerを選択する。だいたいはTextureBaker and MeshBakerでいいと思う。テクスチャを劣化させたくないときなどはMesh Bakerのみでもいいだろう。

1. (MB3\_Texture\_Baker.csでまず作業)Open Tools For Adding Objectsを押す。
2. Search For Meshes To Add→Exclude meshes with out-of-bounds UVsのチェックを外す。
3. ヒエラルキーで一つにまとめるオブジェクトを全部選択してAdd Selected Meshes To Targetを押す。
4. (次にMB3\_Mesh Baker Grouper.csで作業)Lightmappting UvsをGenerate\_new\_UV2\_layoutに変える。
5. (MB3\_Texture\_Baker.csに戻る)Bake Materials into Combined Materialを押す。しばらく待つ。
6. TextureBakerの子になっているMeshBakerにアタッチされているMB3\_Mesh Baker.csのBakeボタンを押す。
7. そうすると、ベイクされたオブジェクトが生成されるので、最後にライトマップをベイクする。

問題 ベイク後、テクスチャが乱れており、カメラの角度によってテクスチャが流れていくように見える。また、Emissionのマテリアルから出た光がベイクされていない。

解決 Mesh Bakerを適用すると、放射マテリアルから出ている光によるベイクはされないようだ。壁や床のようなマテリアルがタイリングされているオブジェクトにMesh Bakerはうまく適用されないので、Mesh Bakerから外すのが良い。

異なるメッシュのまとまりを作るには一回Mesh Bakerを消して、再度Mesh Bakerをヒエラルキーに入れればうまくいくと分かった。何かが初期化されるのだと思う。

橋をMesh Bakerにかけると断面のメッシュに樹皮のテクスチャが割り当てられていてうまくいっていなかったが、鬼をMesh Bakerにかけるとうまくいった。これは一つのオブジェクトに複数のマテリアルを割り当てているとうまくいかないということが理由なのではないか…？

2020/08/10

【Unity】

Mesh Bakerでプレハブを生成する方法

1. “Texture Baker(0)”に関しては8/9の5番までと同じ。
2. “Mesh Baker”についてからがただのシーンオブジェクトとは異なる。まず、ヒエラルキー内に空のゲームオブジェクトを生成する。
3. 生成した空のゲームオブジェクトをプレハブのファイルに移動する。
4. MB3\_Mesh Baker.cs→Output→Outputを”Bake into Prefab”に変更する。
5. できたプレハブをMB3\_Mesh Baker.cs→Combined Mesh Prefabにドラッグ&ドロップする。
6. 最後にMB3\_Mesh Baker.csの”Bake”ボタンを押す。

(参考にしたサイト: <https://www.youtube.com/watch?annotation_id=annotation_3900969295&feature=iv&src_vid=OZlvOEE2CGY&v=AGChQ7lnMOc>)

Mesh Bakerはアニメーションしているオブジェクトには使えない。メッシュが消失した。

アニメーションによって増えるBatch数を減らす方法については結局よく分からなかった(Skinned Mesh Rendererを搭載しているオブジェクトならうまくいくと思われるが、Blenderで作った鬼にはMesh Rendererが割り当てられていたのでAnimation Instancingは使えなかった。)が、実機でビルドしてProfilerで見てみたら60fpsは安定して出ていたので妥協することにした。

【Android】

Android Debug Bridge(adb)のインストール方法: <https://www.orefolder.net/blog/2017/03/platform-tools/>

※書いてあることを全部やった後、再起動したらうまくいった。

Android実機でUnityのProfilerを見る方法: <https://light11.hatenadiary.com/entry/2019/07/14/235740>

adbトンネルを作るときに使うBundle IdentifierはFile→Build Settings→Player Settings→Player→Other Settings→Identification→Package Nameを引用した。

実機でゲームを動かしてその時の様子をProfilerで見ると、どこがボトルネックになっているかが分かる。

**2020/08/11**

【C#】

ボタンの辺り判定を円形にする: <http://tsubakit1.hateblo.jp/entry/2015/01/30/213000>

このページに書いてあるコードを辺り判定を持たせたいImage等にアタッチすればよい。

AdMobのバナー広告を実装するポイント

バナー広告をリクエストするのは最初の1回でいい。1回リクエストしたら、あとは表示するときに

bannerView.show();

として、非表示にするときに

bannerView.Hide();

とすればいいのではないかと思う。

リワード広告を実装するときに <https://developers.google.com/admob/unity/rewarded-ads?hl=ja> を参考にした。

HandleUserEarnedRewardメソッドの中で通常の通りにメソッドを呼び出したりゲームオブジェクトを操作することができないため、以下のサイトで入手したコードを使ってHandleUserEarnedRewardメソッド(他のイベントハンドラも同様)の中で処理を行った。下のコードのMainThreadEventExecutorはこのページから引用した。

<https://teratail.com/questions/274468>

リワード広告の実装一覧

using GoogleMobileAds.Api;

using System;

using UnityEngine;

using UnityEngine.UI;

public class RewardAdMob : MonoBehaviour

{

private UIManager uiManager;

private AudioSource[] audioSources; //シーン内にあるすべてのAudioSource

private RewardedAd rewardedAd;

private void Awake() {

MainThreadEventExecutor.Initialize();

}

void Start() {

uiManager = GameObject.Find("UIManager").GetComponent<UIManager>();

audioSources = new AudioSource[]

{

GameObject.Find("Player").GetComponent<AudioSource>(),

GameObject.Find("StageManager").GetComponent<AudioSource>(),

GameObject.Find("UIManager").GetComponent<AudioSource>()

};

#if UNITY\_ANDROID

string adUnitId = "ca-app-pub-3940256099942544/5224354917";

#elif UNITY\_IPHONE

string adUnitId = "ca-app-pub-3940256099942544/1712485313";

#else

string adUnitId = "unexpected\_platform";

#endif

this.rewardedAd = new RewardedAd(adUnitId);

// Called when an ad request has successfully loaded.

this.rewardedAd.OnAdLoaded += HandleRewardedAdLoaded;

// Called when an ad request failed to load.

this.rewardedAd.OnAdFailedToLoad += HandleRewardedAdFailedToLoad;

// Called when an ad is shown.

this.rewardedAd.OnAdOpening += HandleRewardedAdOpening;

// Called when an ad request failed to show.

this.rewardedAd.OnAdFailedToShow += HandleRewardedAdFailedToShow;

// Called when the user should be rewarded for interacting with the ad.

this.rewardedAd.OnUserEarnedReward += HandleUserEarnedReward;

// Called when the ad is closed.

this.rewardedAd.OnAdClosed += HandleRewardedAdClosed;

// Create an empty ad request.

AdRequest request = new AdRequest.Builder().Build();

// Load the rewarded ad with the request.

this.rewardedAd.LoadAd(request);

}

public void CreateAndLoadRewardedAd() {

#if UNITY\_ANDROID

string adUnitId = "ca-app-pub-3940256099942544/5224354917";

#elif UNITY\_IPHONE

string adUnitId = "ca-app-pub-3940256099942544/1712485313";

#else

string adUnitId = "unexpected\_platform";

#endif

this.rewardedAd = new RewardedAd(adUnitId);

this.rewardedAd.OnAdLoaded += HandleRewardedAdLoaded;

this.rewardedAd.OnUserEarnedReward += HandleUserEarnedReward;

this.rewardedAd.OnAdClosed += HandleRewardedAdClosed;

// Create an empty ad request.

AdRequest request = new AdRequest.Builder().Build();

// Load the rewarded ad with the request.

this.rewardedAd.LoadAd(request);

}

/\*リワード広告を視聴する\*/

public void UserChoseToWatchAd() {

if (this.rewardedAd.IsLoaded()) {

this.rewardedAd.Show();

}

}

private void HandleRewardedAdFailedToShow(object sender, AdErrorEventArgs e) {

}

/\*リワード広告が開くときに実行される\*/

private void HandleRewardedAdOpening(object sender, EventArgs e) {

MainThreadEventExecutor.Execute(() =>

{

uiManager.IS\_GOING\_ON = false; //ゲームを停止する

/\*リワード広告視聴中はアプリの音が聞こえないようにする\*/

foreach (AudioSource audioSource in audioSources) {

audioSource.mute = true;

}

});

}

private void HandleRewardedAdFailedToLoad(object sender, AdErrorEventArgs e) {

}

private void HandleRewardedAdClosed(object sender, EventArgs e) {

MainThreadEventExecutor.Execute(() =>

{

uiManager.CloseWatchRewardVideoScreen(); //リワード広告を見るかどうか聞くパネルを閉じる

/\*シーン内の音を復活させる\*/

foreach (AudioSource audioSource in audioSources) {

audioSource.mute = false;

}

uiManager.IS\_GOING\_ON = true; //ゲームを再開する

this.CreateAndLoadRewardedAd(); //新たなリワード広告を読み込んでおく

});

}

/\*動画を視聴したユーザーに報酬を付与するときに実行される\*/

private void HandleUserEarnedReward(object sender, Reward e) {

MainThreadEventExecutor.Execute(() =>

{

uiManager.OpenWholeMap(); //全体マップを見せる

});

}

private void HandleRewardedAdLoaded(object sender, EventArgs e) {

}

void Update() {

}

}

ボタンを一度に複数回押されないようにするには、一度ボタンを押したらそのボタンのinteractableをfalseにすればいい。

(参考にしたサイト: <https://tech.pjin.jp/blog/2018/01/10/unity_ugui_one-click-button/>)

【Photoshop】

画像をすでにあるパレットの上に持ってくるにはドラッグ&ドロップする。

【C#】

本ゲームはスマホを横持ちにして遊ぶが、横持ちの場合は

Canvas→Render Mode: Screen Space – Overlay

Canvas Scaler→UI Scale Mode: Scale With Screen Size

Canvas Scaler→Screen Match Mode: Match Width Or Height

Canvas Scaler→Match: Height(1)

に設定したらうまくいった。

**2020/08/17**

【Google Play】

Google Playにリリースする際のアイコンの作り方: <https://developer.android.com/google-play/resources/icon-design-specifications>

参考にする予定のプロモーション動画

<https://play.google.com/store/apps/details?id=com.netease.wxzcglobal&hl=en_US>

<https://play.google.com/store/apps/details?id=com.enteriosoft.extremebalancer3&hl=ja>

<https://play.google.com/store/apps/details?id=com.enteriosoft.extremebalancer3&hl=ja>

<https://play.google.com/store/apps/details?id=jp.gungho.pad&hl=ja>

<https://play.google.com/store/apps/details?id=com.madfingergames.wargames.shooter&hl=ja>

<https://play.google.com/store/apps/details?id=com.netease.idv.googleplay&hl=ja>

**2020/08/20**

【Premiere Pro】

動画のスタート位置の変更はドラッグ&ドロップでできる。

プログラムウィンドウにある「フレームを書きだし」ボタン(カメラのマーク)を押すと、現在プログラムウィンドウに映っているフレームを画像として書き出せる。「プロジェクトに読み込む」にチェックを入れておくと書き出した画像がプロジェクトウィンドウに格納される。その画像をV2トラックに入れるなどする。

(参考にしたサイト: <https://www.youtube.com/watch?v=anBsk9leCXI>)

V2がV1より上層のレイヤーのようだ。

フェードインは、V2に映したい動画、V1にファイル→新規→カラーマットで背景色を投下する。そして、V2を選択した状態でエフェクトコントロール→不透明度でフェードイン開始時に不透明度を0%、フェードイン終了時に100%にしておけばよい。その際、「不透明度」の左隣にあるストップウォッチのマークをアクティブにしておかなければならない。

(参考にしたサイト: <https://helpx.adobe.com/jp/premiere-pro/kb/how-to-fade-in-and-out-jp.html>)

**2020/08/21**

プロモーション動画を作る際に使った音源は以下の通り。

女性の声: びたちー素材館

効果音: 効果音ラボ

唾を飲む音: G-Sound 飲み込む音　ゴクリ

【Clip Studio Paint】

テキストの縁取りをする方法:

テキストを選択してノギスボタン→テキスト→フチから設定する。

(参考にしたサイト: <http://www.clip-studio.com/site/gd/csp/manual/userguide/csp_userguide/710_fukidashi_text/710_fukidashi_text_text_edit_fuchi.html>)

集中線の描き方

図形ツールの中にある。

(参考にしたサイト: tips.clip-studio.com/ja-jp/articles/1342)

マスク画像の設定の仕方

**2020/08/23**

【C#】

動画の再生位置をスクリプトで変更するには

private VideoPlayer videoPlayer;

videoPlayer.**time** = ~f;

とする。

(参考にしたサイト: <https://qiita.com/idempotence/items/90af40866ee4cf5f246d>)

**2020/08/24**

【Unity】

いずれ64bit要件に対応していないアプリはGoogle Playで配信できなくなるので、今からアプリは64bit版で作っておくべきである。Unityでやっておく設定は、Edit→Project Settings→Player→Other Settings→ConfigurationにあるScripting Backendを”IL2CPP”にし、ARM64にチェックを入れる。

(参考にしたサイト: <https://www.petitmonte.com/unity/android_64bit.html>)

AAB（Android App Bundle）ファイル形式で Android用実行ファイルを作成すると、ファイル容量が APK形式に比べて小さくすることができる。Android App Bundleをアップロードするには、Google Playアプリ署名に登録している必要がある。これをUnityで実現するには次の方法をとる。UnityでKeystoreを作成する。Edit→Project Settings→Player→Publishing Settingsにある”Keynote Manager”から新しいKeynoteを作成する。パスワードやkey等を決める。進捗ノートに控えた。また、File→Build Settingsにある”Build App Bundle(Google Play)”にチェックを入れる。

(参考にしたサイト: <http://blog.lab7.biz/archives/20428584.html>)

**2020/08/25**

【Unity, C#】

Unityで作ったアプリをSNSで共有する方法

1. **Social Connector**というプラグインをUnityに入れる。<https://github.com/anchan828/social-connector?fbclid=IwAR1MgI8SZXQhqGLl5FfeQN5cBtbWarMDAht4k_XEgLv0iljp5yLoRkazlNs#fileprovider%E3%81%AB%E5%AF%BE%E5%BF%9C%E3%81%99%E3%82%8B> からzipファイルをダウンロードし、解凍する。
2. UnityでAssets→Import Package→Custom Packageから”**SocialConnector.unitypackage**”を選択する。すると、UnityのAssets下に”Social Connector”というフォルダができる。
3. AndroidManifest.xmlを一部書き換える。例えば「鬼ハ追イカケ君ハ逃ゲル」の場合は、Package Nameが”**com.Mikoshi.Onikimi**”なので”com.kyusyukeigo.socialconnector.fileprovider”を” **com.Mikoshi.Onikimi.fileprovider**”に書き換える。書き換えるのはそこだけでOK。Package NameはEdit→ Project Settings→Player→Other Settings→Identificationから確認できる。(参考にしたサイト: <https://github.com/anchan828/social-connector?fbclid=IwAR1MgI8SZXQhqGLl5FfeQN5cBtbWarMDAht4k_XEgLv0iljp5yLoRkazlNs#fileprovider%E3%81%AB%E5%AF%BE%E5%BF%9C%E3%81%99%E3%82%8B>)
4. ファイルの位置を移動させる。Assets/SocialConnector/Plugins/Androidに入っているすべてのフォルダ、ファイル(AndroidManifest.xml, resフォルダ, support-core-utils-25.3.1.aar)をAssets/Plugins/Android下に移動する。
5. SNSで共有する画像をあらかじめ用意しておく場合は、Assets/StreamingAssetsという名前のフォルダを作成し、StreamingAssets内にファイルを置いていく。例えば、フィーチャー画像など。
6. 下のコードを書く。

アプリからTwitterに共有させる際にTLに表示させる画像

* 横1200px以上(画質が上がる)
* アスペクト比16:9(TLに表示されるときにトリミングされない)
* ファイルサイズは5MB以下

が良い。(参考にしたサイト: <https://gaiax-socialmedialab.jp/post-1111/>)

アプリストアのURLを申請前に確認する方法

Google Play:

**play.google.com/store/apps/details?id=(アプリの ID)**

例えばこのアプリだったらplay.google.com/store/apps/details?id=com.Mikoshi.Onikimiとなる。

アプリのIDは**Package Name**と同じである。

App Store:

**app.apple.com/jp/app/(アプリ名)/(アプリ ID)**

例えばこのアプリだったらapp.apple.com/jp/app/鬼ハ追イカケ君ハ逃ゲル/id1529844082

アプリIDはApp Store Connectの一般→App情報から見られる。

(参考にしたサイト: <https://backapp.co.jp/blog/11866/>)

**2020/08/26**

【Unity】

Minimap Cameraが地面に対して垂直になっていないために、鬼のアイコンが角度によって消えたり、位置がずれて見えていたのだろう。カメラを地面に対して垂直にしたら、鬼のアイコンが正常に見えるようになった。SpriteRendererをCanvasのImageに変更しただけでは見た目に何の変化もなかった。

**2020/08/27**

【C#】

ボタンの当たり判定の形をカスタムする方法

このページ(<https://qiita.com/sune2/items/cf9ef9d197b47b2d7a10>)に載っているようにボタンを作成する。Spriteをファイルから選択してInspectorで見ると、”Sprite Editor”というボタンがあるのでそれを押す。あとはCustom Physics Shapeでコライダーの形を決めればOK。

一方、このページ(<http://tsubakit1.hateblo.jp/entry/2015/01/30/213000>)に載っている方法は試してみたが、うまくいかなかった。

【Git】

Windowsで開発したUnityプロジェクトをGitHubを使ってMac miniに移す方法

1. SourceTreeを開いて、”新規”→”ローカルリポジトリを作成”からパスを指定する。鬼ハ追イカケ君ハ逃ゲルだったら、ここは” C:\Users\Hiroaki Koshimizu\Documents\UnityWorks\Play Tig”となる。
2. GitHubでリモートリポジトリを作成する。鬼ハ追イカケ君ハ逃ゲルの場合は、”Play-Tig”という名前にした。.gitignoreで”**Unity**”を忘れずに選択しよう。
3. SourceTree画面の右上にある”設定”→”リモート”で「追加」ボタンを選択し、”リモートの名前”に”origin”、”URL”に先ほど作成したリモートリポジトリのURLを入れる。
4. SourceTree画面で”プル”を押し、**gitignore**ファイルをダウンロードする(参考にしたサイト: [https://naichilab.hatenablog.com/entry/unity-github)。 gitignore](https://naichilab.hatenablog.com/entry/unity-github)。%20gitignore)ファイルはSourceTreeの画面で右クリックから編集できる。gitignoreファイルに書くテキストは<https://ikep.hatenablog.com/entry/2018/11/09/005207> をコピーした。また、**gitattribute**ファイルは”リポジトリ”→”Git LTS”→”リポジトリを初期化”を選択し、適当に拡張子を1つくらい入れておけば作成できる。そのあとSourceTree画面でgitattributeファイルを右クリックし、中身を<https://ikep.hatenablog.com/entry/2018/11/09/005207> に書き換えることで完成。(ファイルの中身は<https://qiita.com/cs1000/items/07368892a599b2b7b836>でもいいかも。)

(参考にしたサイト: <https://xrdevja.hatenablog.com/entry/2019/03/01/012021>)

1. 次に、ローカルリポジトリのファイルたちをリモートリポジトリにプッシュする。SourceTree画面でターミナルを開き、

$ git add .

$ git commit -m "メッセージ"

$ git remote add origin “ブラウザで作成したリモートリポジトリのurl”

$ git push origin master

を入力する。(参考にしたサイト: <https://ikep.hatenablog.com/entry/2018/11/09/005207>)

1. 続いて、Mac miniにリモートリポジトリにあるファイルをプルする作業だ。まず、Mac miniの好きなところにローカルリポジトリとするファイルを作成する。
2. Mac miniでターミナルを開いて、以下のコマンドを入力する。

$ cd ~ (今作ったファイルに移動する)

$ git init (これは必要なのかよく分からない)

$ git remote add origin “リモートリポジトリのURL”

$ git pull origin master

これで完了！！！

MacのUnityでデフォルトのテキストエディタをVisual Studioにするには、Unity→Preferences→External Tools→External Script Editorから行く。WindowsのUnityとはUIが異なる。

**2020/08/28**

【Unity】

Windowsでのビルド時に出たエラーで

CommandInvokationFailure: Gradle build failed.

C:/Program Files/2019.4.1f1/Editor/Data/PlaybackEngines/AndroidPlayer\OpenJDK\bin\java.exe -classpath "C:\Program Files\2019.4.1f1\Editor\Data\PlaybackEngines\AndroidPlayer\Tools\gradle\lib\gradle-launcher-5.1.1.jar" org.gradle.launcher.GradleMain "-Dorg.gradle.jvmargs=-Xmx4096m" "bundleRelease"

UnityEditor.BuildPlayerWindow+BuildMethodException: 186 errors

at UnityEditor.BuildPlayerWindow+DefaultBuildMethods.BuildPlayer (UnityEditor.BuildPlayerOptions options) [0x002bb] in <a0a9ad37e16b4f7cb955e1101b72d4cc>:0

at UnityEditor.BuildPlayerWindow.CallBuildMethods (System.Boolean askForBuildLocation, UnityEditor.BuildOptions defaultBuildOptions) [0x00080] in <a0a9ad37e16b4f7cb955e1101b72d4cc>:0

UnityEngine.GUIUtility:ProcessEvent(Int32, IntPtr)

というのが出てきたが、自動的にAndroidの依存関係が解消された後は正常にビルドできるようになった。

ローカルリポジトリをリモートリポジトリで強制的に上書きする方法

$ git fetch origin master

$ git reset --hard origin/master

(参考にしたサイト: <http://www-creators.com/archives/1097>)

.gitattributeファイルに.ttcを入れていなかったので、MacのUnityでフォントのエラーが出た。

iOS向けにビルドするとき、Edit→Project Settings→Player→Other Settingsで

Identification

Bundle Identifier: com.Mikoshi.Onikimi

Signing ID: Hiroaki Koshimizu

Automatically Sign: ✓

Configuration

Scripting Backend: IL2CPP

とした。

【Xcode】

AdMobでCocoaPodsを使っている場合は、プロジェクト名.xcodeprojではなく、プロジェクト名**.xcworkspace**からXcodeを開く。(参考にしたサイト: <https://akatana.games/blog/tips/unity/184/>)

Xcodeを開いたら、”Signing & Capabilities”でAutomatically manage siigningにチェックを入れ、TeamにHiroaki Koshimizuをセットし、Bundle Identifierにcom.Mikoshi.Onikimiを入れた。

XcodeからApp Store Connectにビルド情報をアップロードするのに使ったサイトはこちら。<https://i-app-tec.com/ios/app-release.html>

何も躓くことなくスムーズにいった。とても分かりやすい記事だった。

**2020/08/29**

【Unity】

<https://tedenglish.site/unity-modify-anyresolution/> ←このページの通りにスクリプトを貼ったりオブジェクトを作り、うまく画面が表示されなかったので戻そうとしたらTransformのPositionにNaNが入って位置データが吹っ飛んだ。二度とこのサイトを参考にしないと心に誓った。

テキストのSizeを大きくして多くの文字数を配置すると、

Failed to update dynamic font (HGRME) texture; all the needed characters do not fit onto a single texture (max size 4096). Try using less text or a smaller font size.

というエラーが出た。これはSizeを小さくすることで解決できそうだ。画面の解像度を上げれば、テキストのSizeを小さくしてもぼやけて見えないので問題ない。

UIの配置がスマホの画面比によって変わらないようにするコード

このコードをカメラにアタッチする。

using UnityEngine;

public class CameraScaleManager : MonoBehaviour

{

private Camera cam;

// 固定したい表示サイズ

private float width = 1000f; //この値はプロジェクトによって異なるだろう

private float height = 500f; //この値はプロジェクトによって異なるだろう

// 画像のPixel Per Unit

private float pixelPerUnit = 100f;

//カメラのSize設定が height / 2 / picelParUnit である必要がある

//picelParUnitが 200 で height が 1920 なら カメラのサイズは 4.8になる。

void Awake() {

float aspect = (float)Screen.height / (float)Screen.width; //表示画面のアスペクト比

float bgAcpect = height / width; //理想とするアスペクト比

// カメラコンポーネントを取得します

cam = GetComponent<Camera>();

// カメラのorthographicSizeを設定

cam.orthographicSize = (height / 2f / pixelPerUnit);

if (bgAcpect > aspect) {

//画面が横に広いとき

// 倍率

float bgScale = height / Screen.height;

// viewport rectの幅

float camWidth = width / (Screen.width \* bgScale);

// viewportRectを設定

cam.rect = new Rect((1f - camWidth) / 2f, 0f, camWidth, 1f);

}

else {

//画面が縦に長い

//想定しているアスペクト比とどれだけ差があるかを出す

float bgScale = aspect / bgAcpect;

// カメラのorthographicSizeを縦の長さに合わせて設定しなおす

cam.orthographicSize \*= bgScale;

// viewportRectを設定

cam.rect = new Rect(0f, 0f, 1f, 1f);

}

}

}

(参考にしたサイト: <https://pengoya.net/unity/aspect/>)

Canvasの端の方にあるUIはスマホによって画面比が異なる場合に、アンカーを中央に設定しておくと何かと位置がずれそうなので、例えばそのUIがCanvasの右上にあるならアンカーも右上に設定しておくとよいだろう。(参考にしたサイト: <http://www.metalbrage.com/UnityTutorials/uGUI/RectTool.html>)

**2020/08/30**

【Unity】

UIのアンカーを左上や右下に合わせても画面の解像度によってUIの場所が変わってしまう問題について

CourseSelectionシーンでは、階層構造を

Canvas

|--- Title Screen  
 |--- Course Selection Screen

|--- ~

として各スクリーンは空のオブジェクトにしていた。各スクリーンの子に様々なUIを置いていたが、この状態だとUIのアンカーは各スクリーンのRect Transformを基準に定められる。したがって、各スクリーンのRect Transformの値が一番の親であるCanvasのRect Transformの値と一致していなければならない。この部分が一致していなかったために、UIの配置が上手くいっていなかった。

一方、StageシーンではCanvasの子に直接各種UIを設置していたので、UIの配置が上手くいっていた。

CanvasのRectTransformのWidth, Heightを取得、設定する方法

→<https://qiita.com/7of9/items/0eb8ee3dabf5449f6ae2>

iOS14に対応しているUnity向けのAdMobは現時点では公開されていないようなので、iOS14にも対応したUnity向けのAdMobが出てから実装しようかと思っている。

参考にしたサイト:

<https://developers.google.com/admob/unity/start?hl=ja>

<https://support.google.com/admob/answer/9997589?hl=ja>

**2020/08/31**

【Unity】

mmファイルはPyCharmで開けた。編集もできた。

iPadでもSocial Connectorを使えるようにする方法→<https://develop.hateblo.jp/entry/unity-social-connector>

iPhone/iPadのUDIDを確認する方法

XcodeでWindow→Devices and Simulators→Devicesにある”Identifier”に書かれている。

UnityのAndroid API Levelを30にするには、この動画を参考にした(<https://www.youtube.com/watch?v=O3AHZVEDAQY>)。古いバージョンのAPIは削除しても問題ないようだ。

App Storeのストア画面に表示させるAppプレビュー(動画)をスクリーンショットのサイズは

iPhone 6.5インチディスプレイ: 1242x2688

iPhone 5.5インチディスプレイ: 1242x2208

iPad 12.9インチディスプレイ: 2732 x 2048

**2020/09/01**

【Premiere Pro】

途中からフレームサイズを変更するには、上部メニュー→シーケンス→シーケンス設定からフレームサイズの数字を変更すればよい。フレームサイズウィンドウの下の方にあるOKボタンはマウスではどうにも押すことができないようなので、エンターキーで代用した。

【App Store】

入力する際に気を付けたこと

* iOS App
  + バージョン情報
    - Appプレビューとスクリーンショット
      * 解像度が異なるとアップロードできない。
      * スクリーンショットはjpegのほうが無難。
      * Appプレビューとスクリーンショットでは解像度が異なる。

|  |  |  |
| --- | --- | --- |
|  | Appプレビュー | スクリーンショット |
| iPhone 6.5インチ | 1920x886 | 1242x2688 |
| iPhone5.5インチ | 1920x1080 | 1242x2208 |
| iPad | 1600x1200 or 1200x900 | 2732 x 2048 |

ただし、縦横の順番はどちらでも可。

* サポートURLには、鬼ハ追イカケ君ハ逃ゲルに関する内容を本当に軽くだけ書いたはてなブログの記事のURLを入れておいた。
* App Clipには何も入力しなかった。
* App一般情報
  + 著作権の欄には”2020 Hiroaki Koshimizu”と書いた。
* Game Centerには何も入力しなかった。
* App Reviewに関する情報
  + サインイン情報: 何も入力しなかった。
  + 連絡先情報の電話番号は+815037439317とした。
  + メモには500字ほど操作説明を書いておいた。
* 広告ID(IDFA)で広告(AdMob等)を貼るなら「はい、IDFAを使用します」を選択する。
* 一般
  + App情報
    - ローカライズ可能な情報
      * プライバシーポリシーURLには適当にApp Storeで見つけたアプリの分量の少ないプライバシーポリシーの文章をほぼそのままパクって書いた。
    - 一般情報
      * カテゴリと年齢制限を入力。

(参考にしたサイト: <https://itblogdsi.blog.fc2.com/blog-entry-190.html>)

2020/09/05

【レビュー依頼】

https://blog.item-store.net/entry/2017/09/22/193438

Appliv

SQOOL.NETゲーム研究室

スマホゲームCH

アプリナビappnavi

もぐらゲームス

Touch Lab

OCTOBA

iPhoroid

ファミ通App

4Gamer.net

---------------------------------------------------

～様

越水 裕晶と申します。

私の制作したスマートフォンゲームアプリ「鬼ハ追イカケ君ハ逃ゲル」のレビューをしていただきたくご連絡させていただきます。

◆ストアのURL◆

App Store

<https://apps.apple.com/jp/app/%e9%ac%bc%e3%83%8f%e8%bf%bd%e3%82%a4%e3%82%ab%e3%82%b1%e5%90%9b%e3%83%8f%e9%80%83%e3%82%b2%e3%83%ab/id1529844082>

Google Play

<https://play.google.com/store/apps/details?id=com.Mikoshi.Onikimi>

◆本ゲームの概要◆

密閉空間に閉じ込められたプレイヤーが、鬼から逃げながらステージ内のどこかにある脱出口から脱出しようという内容の3Dアクションゲームです。

脱出口の位置は、現在地から脱出口までの距離とミニマップを頼りに探り当てます。

脱出口にたどり着くには鬼から逃げるのみではなく、橋を使って行けない場所に行けるようにしたり、壁を使って鬼の侵入を防ぐことも必要です。

◆面白いポイント◆

・ステージごとに異なる脱出口の位置を探す楽しみがあります。

・一人称視点のため鬼により迫力を感じます。

・プレイヤーの移動はフリック入力のみなので、操作が簡単です。

・鬼が追いかけてくるスリルや、鬼に見つからないように隠れるドキドキ感があります。

・地形や鬼の行動に合わせてうまくアイテムを活用しなければいけません。

◆遊び方◆

コース選択画面からコース番号を指定してから「行く」ボタンを押せば、ゲームプレイ開始です。

鬼から逃げながら、そして橋や壁といったアイテムを駆使しながら、コース内のどこかにある脱出口から脱出できたらクリアです。

脱出口の位置は

・現在地から脱出口までの直線距離(画面左上)

・ミニマップ(画面右上)

・全体マップ(ミニマップをクリックすると表示される)

から特定します。

プレイヤーの移動は上下左右方向のフリック入力のみです。

橋や壁の設置はアイテム欄の木材をドラッグ&ドロップで、橋や壁の回収は「回収する」ボタンから行えます。

私のブログで本ゲームの紹介記事を書いています。こちらのブログに載っている画像はご自由に使っていただいて結構です。

<https://quietmikoshi.hatenablog.com/entry/2020/09/02/015414>

また、本ゲームのプロモーション動画はこちらになります。動画もご自由にお使いください。

<https://www.youtube.com/watch?v=Cum1t6CC-kk>

数多くのレビューをなさっている中でお忙しいとは存じますが、私のアプリのレビューをしていただけましたら幸いです。どうぞよろしくお願いいたします。

2020/09/07

【レビュー依頼】

みこしです

もし良かったらでいいんだけど、またりん君のブログかnoteでワイのスマホゲームのレビュー記事を書いてもらえないかな？(嫌だったら断ってくれて全然構わないです)

紹介画像と動画はこちらで用意してあるので、新たにまたりん君の方で用意する手間はかからないと思う