時をかけるプログラマー

名前：芦塚 大樹

学校：福岡情報ITクリエイター専門学校

趣味：映画鑑賞,読書,バイク

出身：長崎県諫早市

スキル

- C/C++（３年）・DxLib（３年）
- C#/Unity（２年）
- Git/GitHub（２年）・Photoshop（２年）
- Illustrator（１年）・Maya/Blender（２年）

自己紹介

長崎！

芦塚大樹

はまっている食べ物

インドカレー

好きなもの

バイク

最終
ゲーム作品

最終作品

オトシキング

- ジャンル　３D対戦アクションゲーム
- 開発環境　C++,DxLib
- 制作人数　1人
- 制作期間　約2ヶ月半

相手をオトセ！

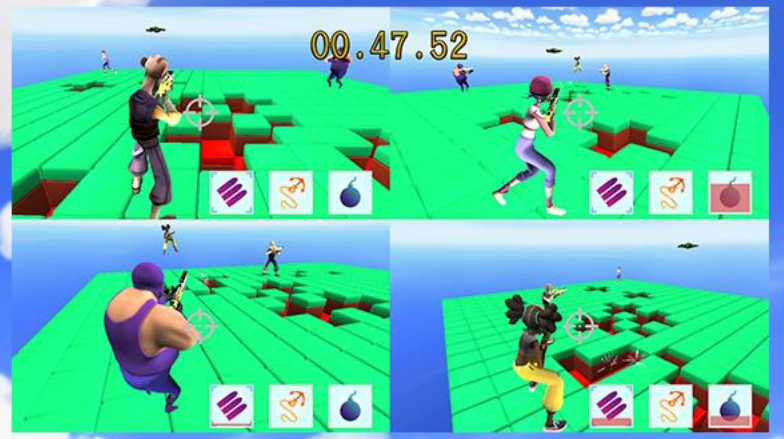
ファイターを選べ！

GitHub 作品リポジトリ URL
https://github.com/aluuuuuuuuuu/OtoshiKing.git

GoogleDrive 作品動画 URL
https://drive.google.com/file/d/1UrEHFcUrJToYCFWc6mxTHRSNTrf_pk3o/view?usp=drive_link

オトシキング

4体のファイターから好みのキャラを選ぼう！



最大4人のマルチプレイ！

友人と共に競い合え！

オトシキング

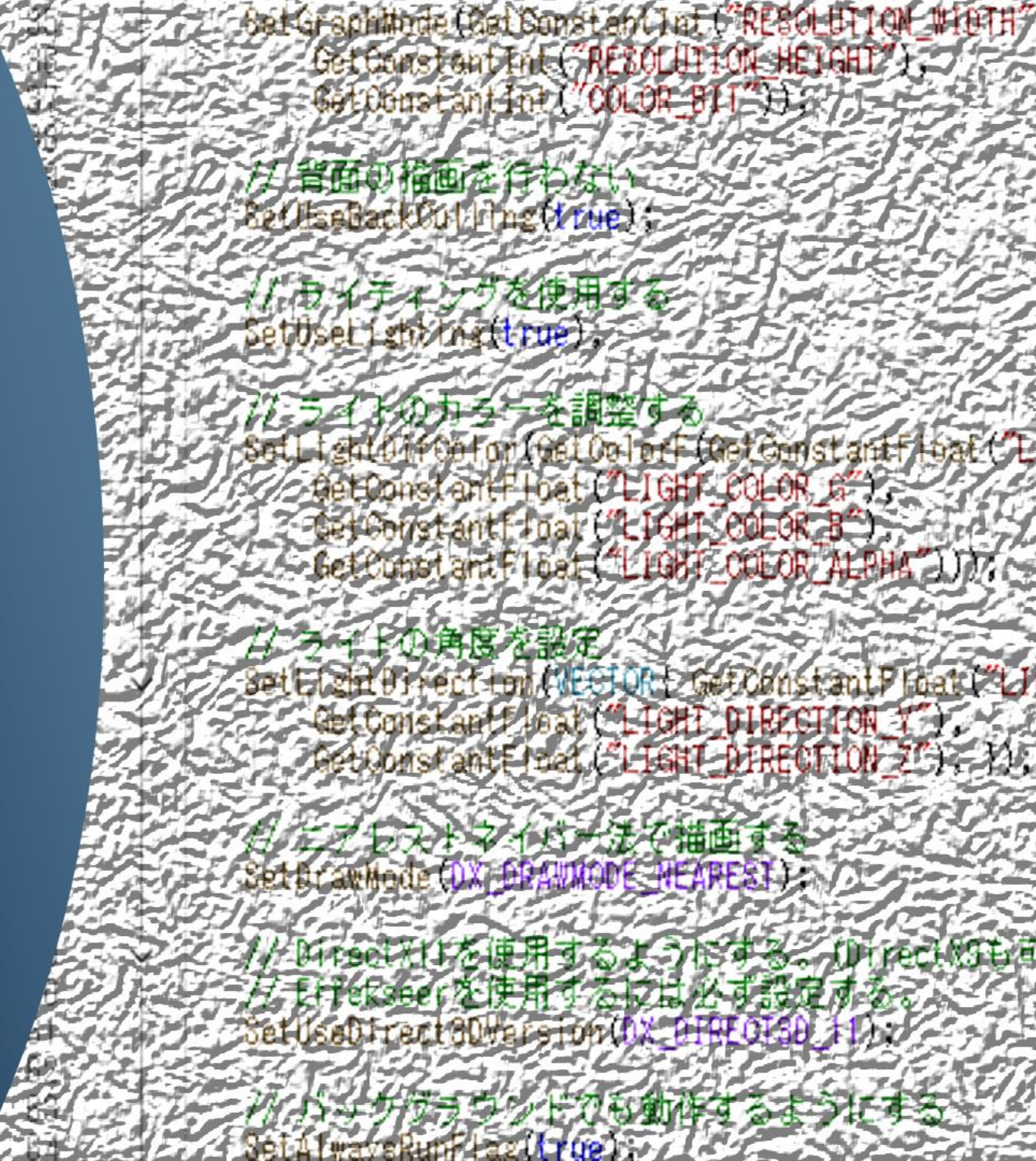
３つの弾を使い分けてステージを破壊！
縦横無尽に駆け回ろう！

高速連射で制圧せよ！

広い範囲を破壊！

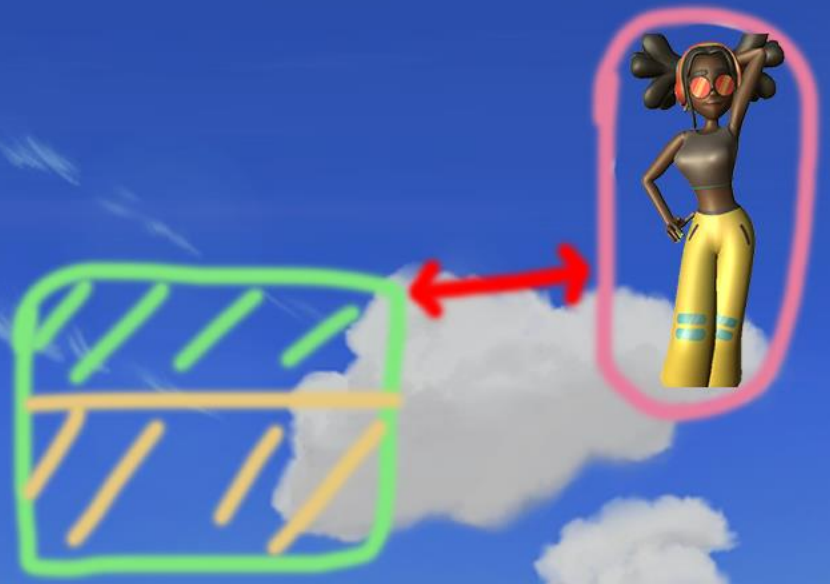
どこでも駆け抜ける！

オトシキング
技術紹介



技術紹介① ステージとプレイヤーの当たり判定

①ボックスとカプセルの線分の最近接点を求める



AABB上の最近接点を求める関数

```
Vec3 CollisionManager::ClosetPointBox(Vec3 max, Vec3 min, Vec3 point)
{
    // 最近接点
    Vec3 closestPoint;

    // xyz軸について判定
    closestPoint.x = (std::max)(min.x, (std::min)(point.x, max.x));
    closestPoint.y = (std::max)(min.y, (std::min)(point.y, max.y));
    closestPoint.z = (std::max)(min.z, (std::min)(point.z, max.z));

    return closestPoint;
}
```

②最近接点同士の距離を求める

③距離とカプセルの半径を比較する

技術紹介② 定数の外部ファイル化

定数を CSV ファイルに記述する

	A	B	C	D	E
1	No.	定数名	型名	定数の値	説明
2	1	SCREEN_WIDTH	_INT	1920	画面の幅の値
3	2	SCREEN_HEIGHT	_INT	1080	画面の高さの値
4	3	COLOR_BIT	_INT	32	カラービット値
5	4	RESOLUTION_WIDTH	_INT	1920	解像度の幅の値
6	5	RESOLUTION_HEIGHT	_INT	1080	解像度の高さの値

定数ファイルを読み込む

```
// 外部ファイルから定数を取得する
ReadCSV("data/constant/Application.csv");
```

読み込んだ定数を使用する

```
// 画面サイズの設定
SetWindowSize(GetConstantInt("SCREEN_WIDTH"),
    GetConstantInt("SCREEN_HEIGHT"));
```

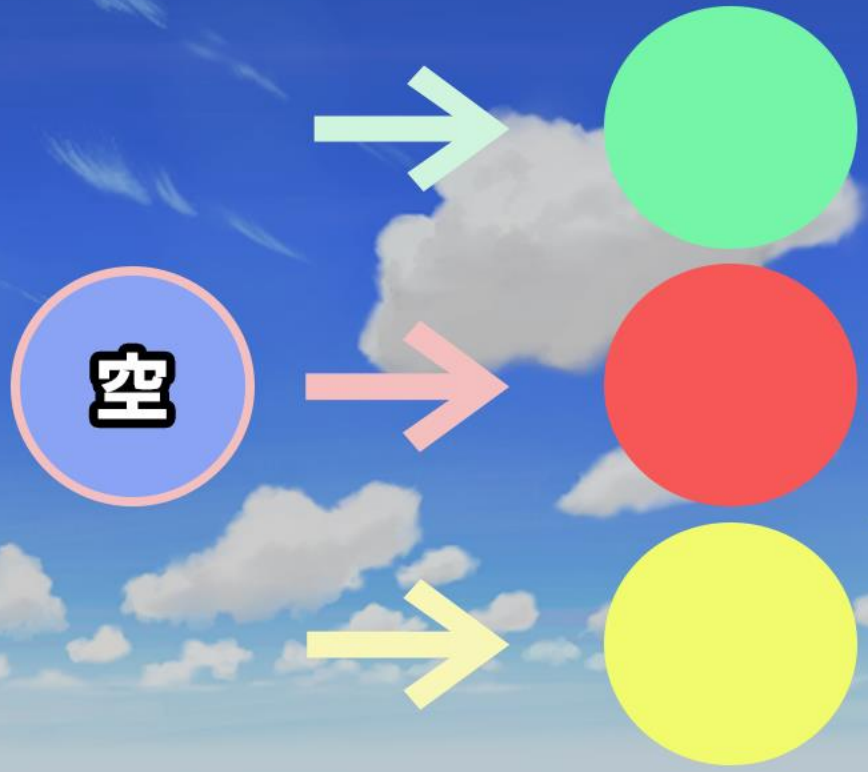
定数を管理するクラス

↓

```
// 定数管理クラス
class Constant
{
    // 定数を格納するための型
    using ConstantVariant = std::variant<int, float, bool, std::string>;

public:

    /// <summary>
    /// 定数mapをそのまま返す
    /// </summary>
    /// <returns>定数map配列</returns>
    std::map<std::string, ConstantVariant> GetConstants();

    /// <summary>
    /// int型の定数を返す
    /// </summary>
    /// <param name="name">定数名</param>
    /// <returns>定数の値</returns>
    int GetConstantInt(std::string name) const;

    /// <summary>
    /// float型の定数を返す
    /// </summary>
    /// <param name="name">定数名</param>
    /// <returns>定数の値</returns>
    float GetConstantFloat(std::string name) const;

    /// <summary>
    /// bool型の定数を返す
    /// </summary>
    /// <param name="name">定数名</param>
    /// <returns>定数の値</returns>
    bool GetConstatBool(std::string name) const;
```

技術紹介③ 弾のベースクラスの作成と管理

・空の弾丸クラスを作成することで
弾の量産とマネージャークラスでの管理を容易にした



すべての弾丸は
弾のベースクラスを
継承する

オトシキングの振り返り

オトシキング

反省

- ゲーム内容がシンプルすぎる
- 演出が弱い

これからやりたいこと

- キャラクターそれぞれに能力など、キャラ差をつける
- アイテムやステージ拡張、オンラインマルチプレイの実装
- 勝利演出やゲームスタート時の演出の強化

ここからは

過去の制作物

制作実績

- 作品名：アルティメットニンジャ
- ジャンル：ワイヤーアクションゲーム
- 開発環境：C++/DxLib
- 制作期間：２～３ヶ月

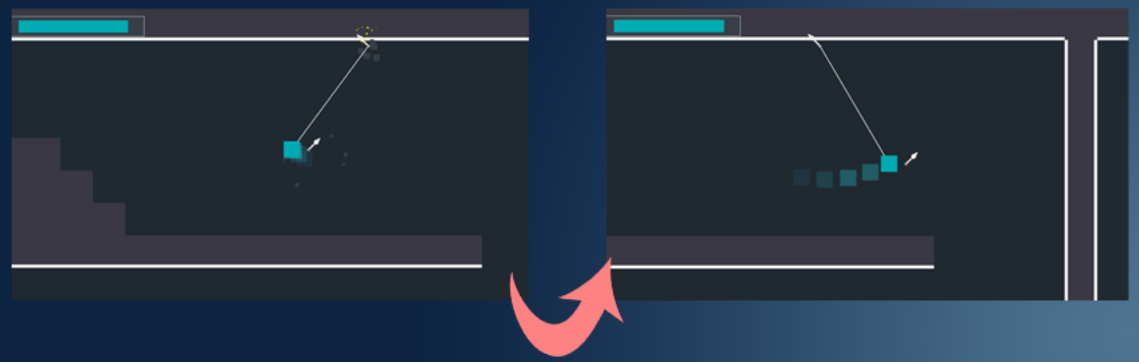
頑張ったところ

- 振り子運動の物理挙動

次ページ→

アルティメットニンジャ

技術紹介：振り子運動



ワイヤーを固定した座標を中心に

滑らかに振り子運動する。

定数の値を変えることで

運動の速度などを簡単に

変更することができる。

```
// 振り子の状態を表すクラス
class Pendulum {
public:
    int length;             // ワイヤーの長さ
    double angle;           // 現在の角度
    double angularVelocity; // 角速度
    double centerX;         // 中心のX座標
    double centerY;         // 中心のY座標
    int circleCenterX;
    int circleCenterY;

    // 振り子の運動を開始する関数
    void Set(double initialAngle, double x, double y) {
        angle = initialAngle;
        angularVelocity = 0.0;
        centerX = x;
        centerY = y;
    }

    // 振り子の運動方程式をもとに座標を更新する
    void update(double dt) {
        const double grabity = 9.8;     // 重力加速度
        const double damping = 0.99;    // 減衰率

        double acceleration = -grabity / length * sin(angle);   // 速度=重力/円周の半径*sinθ
        angularVelocity += acceleration * dt;
        angle += angularVelocity * dt;
        angularVelocity *= damping;

        // 角度が0.001以下になれば0を代入する
        if (abs(angle) <= 0.001) {
            angle = 0;
        }
    }
};
```

制作実績

- 作品名：猛将伝説 百戦錬磨
- ジャンル：３D 無双ゲーム
- 開発環境：C++/DxLib
- 制作期間：２ヶ月

頑張ったところ

- エネミーの簡単な AI
- 外部ファイルによるランキングの読み書き

今後の目標

「新時代の架け橋となる

スペシャリスト」

DirectXを直接触れる力を身に着け、

周囲の技術力を底上げするエンジニアになる。