シューティングゲームを作る手順

Siv3D

# 最初のコード

Unicodeで自機を描画し、キーボードのカーソルキーで移動できるようにします。

キーボードの矢印キーで自機の移動が

# include <Siv3D.hpp> // Siv3D v0.6.15

void Main()

{

Scene::SetBackground(ColorF{ 0.1, 0.2, 0.7 });

//const Font font{ FontMethod::MSDF, 48 };

// 自機テクスチャ // 🧘🤖⚜

const Texture playerTexture{ U"🧘"\_emoji };

// 自機（初期位置）.

Vec2 playerPos{ 400, 500 };

while (System::Update())

{

// 自機のスピード.

constexpr double PlayerSpeed = 550.0;

// 1フレームの経過時間.

const double deltaTime = Scene::DeltaTime();

// 自機の移動ベクトル.

// 入力方向→正規化→スピード係数をかけて調整→シフト入力で減速する仕組み.

const Vec2 move = Vec2{

(KeyRight.pressed() - KeyLeft.pressed()),

(KeyDown.pressed() - KeyUp.pressed())

}.setLength(deltaTime \* PlayerSpeed \* (KeyShift.pressed() ? 0.5 : 1.0));

// 自機の移動.

playerPos.moveBy(move).clamp(Scene::Rect());

// 自機を描画.

//playerTexture.resized(80).flipped().drawAt(playerPos);

playerTexture.resized(80).drawAt(playerPos);

}

}

//

// - Debug ビルド: プログラムの最適化を減らす代わりに、エラーやクラッシュ時に詳細な情報を得られます。

//

// - Release ビルド: 最大限の最適化でビルドします。

//

// - [デバッグ] メニュー → [デバッグの開始] でプログラムを実行すると、[出力] ウィンドウに詳細なログが表示され、エラーの原因を探せます。

//

// - Visual Studio を更新した直後は、プログラムのリビルド（[ビルド]メニュー → [ソリューションのリビルド]）が必要な場合があります。

# 自機弾を自動で発射

図形

自動的に生成された説明

弾の位置を配列で管理

Array<Vec2> playerBullets;

自機弾を描画.

for (const auto& playerBullet : playerBullets)

{

Circle{ playerBullet, 8 }.draw(Palette::Orange);

}