第 15 章

Siv3D でゲームを作る

75 回生　neko

15.1　はじめに

75 回生の neko です。Siv3D でゲームを作っていこうと思います。

15.2　Siv3D とは

Siv3D とは、C++ のライブラリで、簡単に言えばゲームやツールを作りやすくするものです。

Siv3D は 2012 年に初めて公開された比較的新しいライブラリで、グラフィカルでインタラクティブなアプリケーションを作ることができます。

しかし、最近、より実用的でモダンな設計にするために、Siv3D の開発を終了し、OpenSiv3D の開発がされているのですが、まだ開発途中で、リファレンスなどが乏しい状況なので、今回は Siv3D で開発していくことにします。

C++ の知識が多少必要ですが、公式 Wiki にチュートリアルが載っていますし、C++ を学びながらゲームを作ることができます。

15.3　Siv3D の良い所

短いコードで簡単に書けること。特に当たり判定が非常に楽に実装できることです。

どれだけ短いコードで書けるかというのを公式サイトから引用したものを紹介します。

List 15.1: oekaki.cpp

```
#include <Siv3D.hpp>

void Main()
{
    Image image(Window::Size(), Palette::White);

    DynamicTexture texture(image);

    while (System::Update())
    {
        if (Input::MouseL.pressed)
        {
            Line(Mouse::PreviousPos(), Mouse::Pos()).overwrite(image, 8, Palette::Orange);

            texture.fill(image);
        }

        texture.draw();
    }
}
```

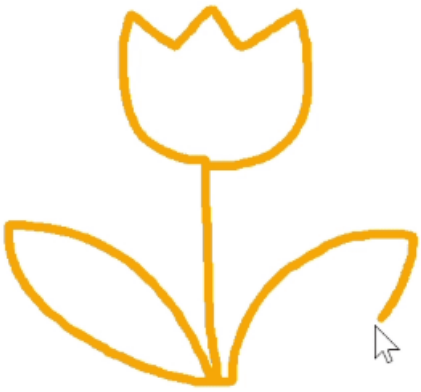

図 15.1: 実際の画面

オレンジ色のペンで絵を描くだけのプログラムですが、これだけで実装できます。
このように短いコードでプログラムを書くことができます。

15.4 環境構築

まず、環境構築ですが、公式 wiki に結構分かりやすく書いてあると思うのでそれを見ればできると思います。ちなみに、この部誌では Visual Studio 2019 で開発していこうと思います。

15.5 Siv3D の基本

基本的な Siv3D の仕様について書いていきます。
プログラムは基本的にこのようなコードで書きます

List 15.2: kihon.cpp

```
#include <Siv3D.hpp>

void Main(){

        while (System::Update()){

        }
}
```

`while(System::Update()){}` の部分が、メインループで、フレーム毎の処理を書きます。フレームは最大 60FPS です。
また、座標についてですが、画面の左上が x=0、y=0 で、右に行くと、+x、下に行くと +y です。
他にも、図形を作る方法だったり色々あるんですが、これからコードを書くときにある程度の解説を交えつつ書いていくので大丈夫です。詳しくは、公式 wiki のチュートリアルを見ればわかります。ただ、C++ の知識はある程度ある (for 文や if 文が分かるくらい) 前提で書きます。

15.6 作るゲームの概要

作るゲームはいたってシンプルで、右から現れる障害物をジャンプして避けるというゲームです。では、ゲームを実装していきます。

15.7 プレイヤーをジャンプさせる

とりあえず、キャラクターのデザインを後回しにして長方形にしておきます。ジャンプの実装ですが、重力を加速度を使って実装するわけではなく、ただの等速運動で実装します。

List 15.3: jamp.cpp

```
#include <Siv3D.hpp>

void Main()
{

        Point player_pos(100, 300);  // プレイヤーの座標の変数

        bool Canfly = 1;// 飛べるかどうか
        while (System::Update()) {

    // spaceキーが押されていてかつ、Canflyがtrueであればプレイヤーのy座標を30下げる(上に30上げる)

                if (Input::KeySpace.pressed && Canfly) {
                        player_pos.y -= 30;

                }
    // プレイヤーのジャンプが上限(この場合100)になった、またはスペースキーが離されたらCanflyをfalseにする。

                if (player_pos.y < 100 || Input::KeySpace.released) {
                        Canfly = 0;
                }
    // 地面についたらCanflyをtrueにする。

                if (player_pos.y >= 300) {
                        Canfly = 1;
                }

    // 重力
                if (player_pos.y < 300) {
                        player_pos.y += 15;
                }

                Rect(player_pos, 100, 50).draw();   // プレイヤー描画

                Line(0, 350, 640, 350).draw();    // 地面描画
        }
}
```

「プレイヤーのジャンプが上限 (この場合 100) になった、またはスペースキーが離されたら Canfly を false にする」というのは、二段ジャンプを防ぐためです。

これで、とりあえず長方形がジャンプするプログラムはできあがりました。

15.8 敵を生成する

敵をまず、Array を使って管理します。そして、画面外右で敵を生成して、画面外左で敵を消します。

先程書いたコードの説明は省きます。

List 15.4: enemy.cpp

```
#include <Siv3D.hpp>

void Main()
{
        Point player_pos(100, 300);
        bool Canfly = 1;

        Array<Rect> enemies;// 敵を管理するもの

        Rect destroy(-500, 100, 10, 1000);// ここに当たったら敵を消す。

        int32 frame_record = 1;

        while (System::Update()) {

                // 60~120のランダムな間隔で敵を生成

                if (frame_record  == System::FrameCount()) {
                        enemies.emplace_back(1200, 300,50,50);
                        frame_record += Random(60,120);

                }

                // 敵を動かす

                for (auto it = enemies.begin(); it != enemies.end(); it++) {
                        it->moveBy(-20,0);
                }



                // プレイヤーの処理
```

```
                if (Input::KeySpace.pressed && Canfly) {
                        player_pos.y -= 30;

                }
                if (player_pos.y < 100 || Input::KeySpace.released) {
                        Canfly = 0;
                }

                if (player_pos.y >= 300) {
                        Canfly = 1;
                }
                if (player_pos.y < 300) {
                        player_pos.y += 15;
                }
    // 画面外左に行ったら敵を削除
                for (auto it = enemies.begin(); it != enemies.end(); it++) {
                        if (it->intersects(destroy)) {
                                enemies.erase(it);
        break;// ここないとバグる
                        }
                }

                // 描画

                Rect(player_pos, 100, 50).draw();
                Line(0, 350, 640, 350).draw();

    // 敵描画
                for (auto& x : enemies) {
                        x.draw();
                }
        }
}
```

15.9 最後

あとは、プレイヤーと敵のデザインと当たり判定を実装するだけです。これらは、この部誌で解説する必要がないと思う (自分で作ってみてほしい) ので説明しません。

15.10 最後に

どんな人が、この文章を読んでいるかわかりませんが、Siv3D おもしろそうだなぁと思ったり、ちょっとゲームを作ってみたくなったと思ったら是非自分で Siv3D を入れて、ゲームを作ってみてください。そして、ここまで読んでくれてありがとうございました。