INSTITUTO FEDERAL DE CIÊNCIA E TECNOLOGIA DE CAMPOS DO JORDÃO

Murilo Ramos Froes de Paula

Jogo de Plataforma

Programação Orientada a Objeto (POO)

CAMPOS DO JORDÃO

2024

RESUMO

Por meio desde Trabalho busco apresentar os conhecimentos adquiridos em sala de aula da disciplina e aplica-los em um Projeto, cujo projeto consiste em construir um jogo utilizando uma biblioteca da linguagem C e C++, além de expandir os conhecimentos de Lógica de Programação e Programação Orientada a Objeto, ao decorrer do trabalho serão descritos os processos para a sua construção, além de outros aspectos técnicos que devemos levar em consideração ao construir um jogo de computador.

PALAVRAS-CHAVE: Jogo; Programação; Disciplina.

ABSTRACT

Through this work I aim to present the knowledge acquired in the classroom and apply it to a project, which consists of building a game using a C and C++ language library, as well as expanding my knowledge of Programming Logic and Object-Oriented Programming. Throughout the work the processes for its construction will be described, as well as other technical aspects that we must take into account when building a computer game.

KEYWORDS: Game; Programming; Discipline.

INTRODUÇÃO

Com o decorrer do tempo, fui adquirindo muitos conhecimentos em diversas áreas da tecnologia, devido a demais oportunidades de colocar em prática, ao iniciar o trabalho, logo comecei a compreender a complexidade do projeto, pois construir um jogo envolve outros fatores além da Programação, em que certas partes obtivem mais facilidade para atribuir ao projeto final, ao decorrer do documento será explicado mais a fundo.

OBJETIVOS

Construir um jogo de computador utilzando os conhecimentos adquiridos em sala de aula.

JUSTIFICATIVA

Elevar o conhecimento e compreensão dos conteúdos abordados no curso de tecnologia.

METODOLOGIA

A abordagem a ser seguida nesse documento, seguirá de forma explicativa e expositiva dos Recursos utilizados para a construção do Projeto de Jogo de Computador.

CONSIDERAÇÕES INICIAIS

Primeiramente, para a construção do projeto foi necessário utilizar a biblioteca Raylib, do qual é disponibilizado de forma gratuita na internet em seu site.

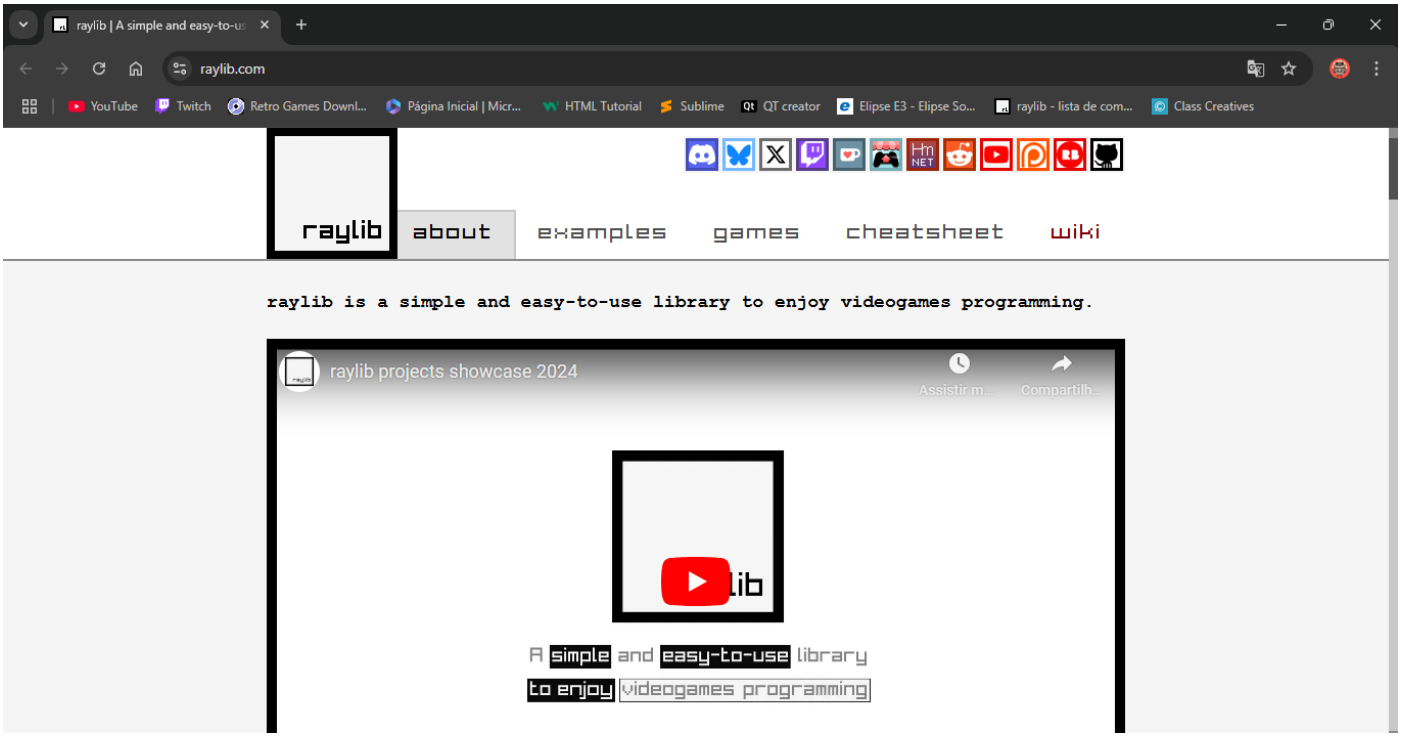

FIGURA 1: Site Raylib

Com a instalação da biblioteca, fora feita a sua configuração no compilador para ser corretamente utilizada, o compilador que foi escolhido fora o NotePad++ do qual está contido na pasta de instalação do Raylib.

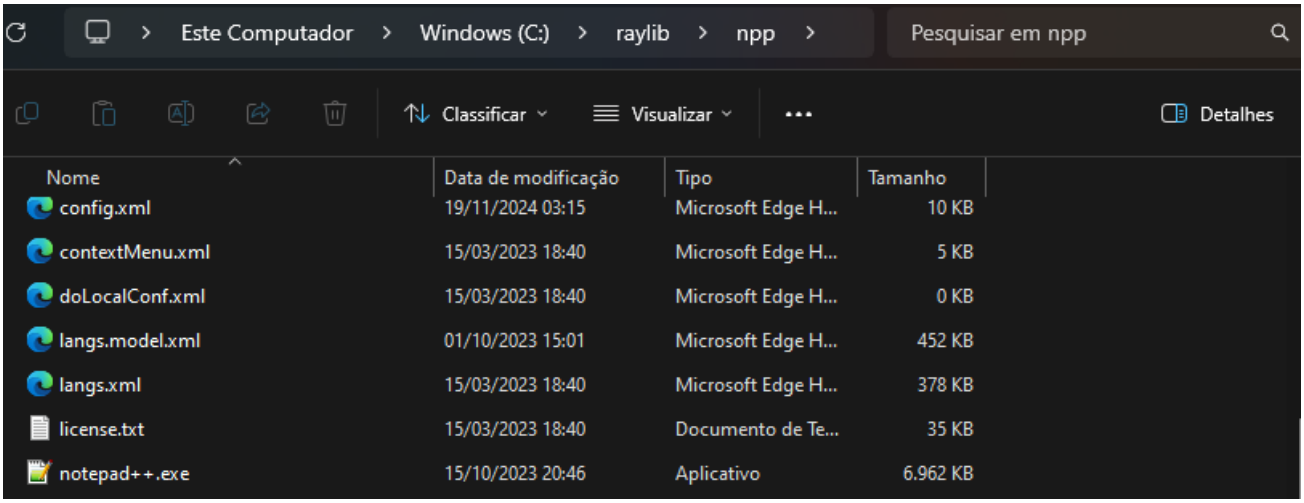

FIGURA 2: Pasta onde se localiza o notepad++

PROJETO

O Jogo construído se chama de Jumper_Joe, um jogo que consiste em um cavalheiro que pula de plataforma em plataforma para não cair no buraco, quanto mais plataformas forem puladas mais pontos o jogador adquire.

Se o jogador cair no buraco é exibido uma tela de Game Over com score adquirido, assim o jogador deve clicar na tela para iniciar uma nova jogada.

Para a utilização dos assets do jogo, fora utilizado um software de criação e edição de fotos e criação de templates personalizados, o programa está disponível de forma gratuita na internet.

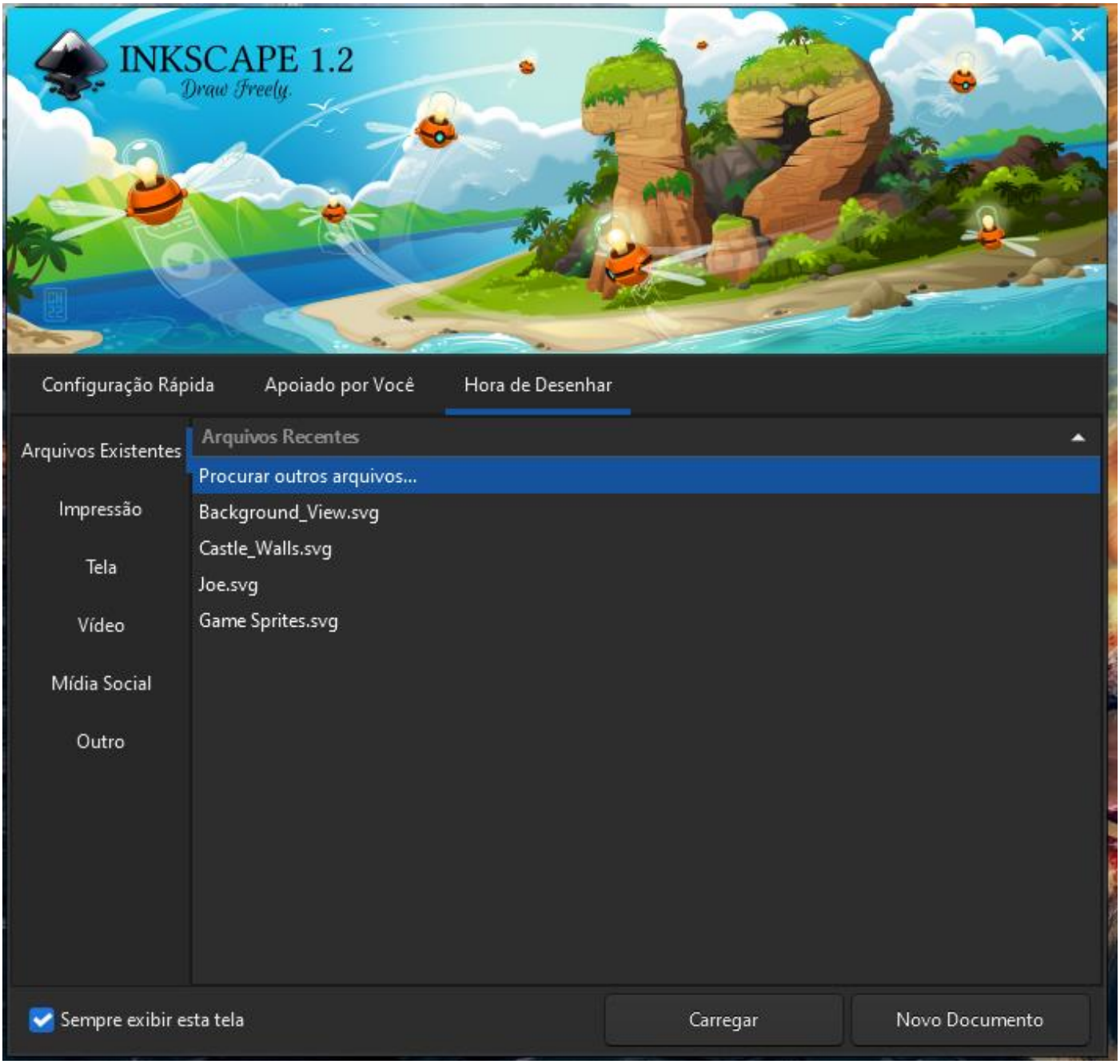

FIGURA 3: Ferramenta Inkscape

RESULTADOS OBTIDOS

Para demonstrar os resultados, devemos compreender o que fora feito em seu código, para as demais etapas, os códigos utilizados serão explicados e suas funcionalidades.

```cpp
#include "raylib.h"
#include <vector>

// Estrutura da plataforma
struct Platform
{
    float x, y;
    bool active;
    Texture2D texture;
    float scale;
};

// Estrutura do jogador
struct Player
{
    float x, y;
    float width, height;
    float speedY;
    Texture2D texture;
    float scale;
};

// Estrutura da parede
struct Wall
{
    float x, y;
    float width, height;
    Texture2D texture;
    float scale;
};
```

FIGURA 4: Código de estruturas.

Estruturas criadas para representar as plataformas, paredes e o jogador para dar sequência a lógica do jogo.

```cpp
int main()
{
    const int screenWidth = 800;
    const int screenHeight = 600;
    InitWindow(screenWidth, screenHeight, "Jumper_Joe");

    // Espaço para alocação de Texturas do jogo
    Texture2D playerTexture = LoadTexture("C:/raylib/raylib/src/Jumper_Joe/Joe.png");
    Texture2D platformTexture = LoadTexture("C:/raylib/raylib/src/Jumper_Joe/Plataform.png");
    Texture2D backgroundTexture = LoadTexture("C:/raylib/raylib/src/Jumper_Joe/Background_View.png");
    Texture2D wallTexture = LoadTexture("C:/raylib/raylib/src/Jumper_Joe/Castle_Walls.png");
    Texture2D startScreenTexture = LoadTexture("C:/raylib/raylib/src/Jumper_Joe/Jumper_Joe_Start.png");
    Texture2D gameOverScreenTexture = LoadTexture("C:/raylib/raylib/src/Jumper_Joe/Game_Over.png");
```

FIGURA 5: Alocação de Texturas

Espaço criado para alocar as texturas para representar os objetos utilizados no jogo.

```
Player player = { screenWidth / 2.0f - 25, screenHeight / 2.0f - 25, 50, 50, 0, playerTexture, 1.0f };
float gravity = 0.2f;
float jumpStrength = 10.0f;
```

FIGURA 6: Geração do Personagem

Gerar Personagem e atribuir a força de gravidade a ser exercida pelo personagem.

```
std::vector<Platform> platforms;
platforms.push_back({ screenWidth / 2.0f - 50, screenHeight - 30, true, platformTexture, 1.0f });

//Colisão das paredes
std::vector<Wall> walls;
walls.push_back({ 0, 0, 50, screenHeight, wallTexture, 1.0f });
walls.push_back({ screenWidth - 50, 0, 50, screenHeight, wallTexture, 1.0f });

int score = 0;
GameState gameState = START_SCREEN;
SetTargetFPS(60);
```

FIGURA 7: Colisão das paredes

Gera as Paredes usadas no jogo além de atribuir colisões.

```
    else if (gameState == GAMEPLAY)
    {
        player.speedY += gravity;
        player.y += player.speedY;

        // Comandos do Personagem
        if (IsKeyDown(KEY_LEFT) && player.x > 50) player.x -= 5;
        if (IsKeyDown(KEY_RIGHT) && player.x < screenWidth - player.width - 50) player.x += 5;
        if (IsKeyDown(KEY_UP)) player.y -= 15;

        // Colisões de plataformas
        for (Platform& platform : platforms)
        {
            if (platform.active &&
                player.speedY > 0 &&
                player.x + player.width > platform.x &&
                player.x < platform.x + platform.texture.width * platform.scale &&
                player.y + player.height > platform.y &&
                player.y + player.height < platform.y + platform.texture.height * platform.scale)
            {
                player.speedY = -jumpStrength;
                platform.active = false;
                score++;
                // Gerar Plataformas após colisões
                platforms.push_back({ (float)(GetRandomValue(50, screenWidth - platform.texture.width * platform.scale - 50
                )),
                                      (float)(GetRandomValue(0, screenHeight / 2)), true, platformTexture, 1.0f });
            }
        }
```

FIGURA 8: Comandos

Quando o jogo está em estado de gameplay, os comandos para jogar com o personagem são aplicados além de gerar colisões com as plataformas e geração de mais plataformas ao serem colididas.

```
// Tela de Game_Over
else if (gameState == GAME_OVER)
{
    if (IsMouseButtonPressed(MOUSE_LEFT_BUTTON))
    {
        gameState = START_SCREEN;
    }

    BeginDrawing();
    ClearBackground(RAYWHITE);
    DrawTexture(gameOverScreenTexture, 0, 0, WHITE);
    DrawText(TextFormat("Final Score: %i", score), 10, 10, 20, DARKGRAY);
    EndDrawing();
}
```

FIGURA 9: Estado de Game Over

Quando o jogo está em estado de “Game Over” ele gera a tela para reiniciar a jogada e exibir o score adquirido pelo personagem, fazendo o jogador voltar a tela inicial e iniciar uma nova jogada.

CONCLUSÃO

Com o jogo finalizado, pude colocar em prática vários dos conceitos apresentados em sala de aula e cheguei a resultados bem próximos do esperado, ao decorrer do projeto O que mais demandou tempo fora a construção dos assets utilizados, em seguida estão as imagens do projeto finalizado.

Sendo um primeiro projeto a ser abordado nesse escopo, considero que aprendi muitas coisas novas durante o seu desenvolvimento, com o tempo pretendo aprimorar os conhecimentos adquiridos e personalizar certos projetos futuros.

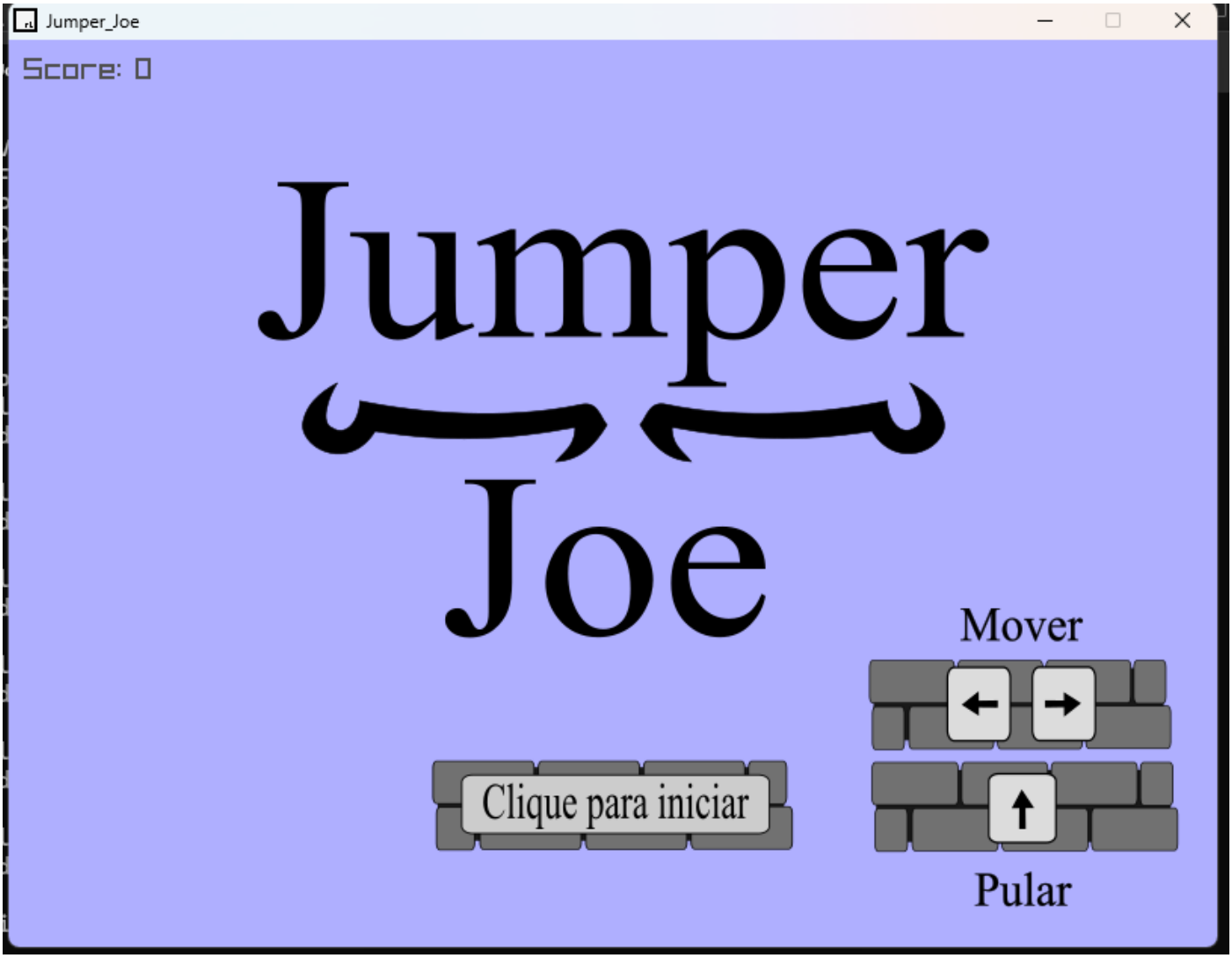

FIGURA 10: Tela Inicial

FIGURA 11: Jogo em Execução

FIGURA 12: Tela de Game_Over

REFERÊNCIAS BIBLIOGRÁFICAS

SARAIVA JÚNIOR, Orlando. Introdução à Orientação a Objetos com C++ e Python: Uma abordagem prática. São Paulo: Novatec Editora Ltda, 2017.

ENG, Lee Zhi. Hands-on Gui Programming with C++ and Qt5. 1st ed. Birmingham: Packt Publishing Ltd. 2018.