TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

,

ĐỒ ÁN MÔN HỌC
KĨ THUẬT LẬP TRÌNH
ĐỀ TÀI: XÂY DỰNG GAME CỜ CARO

Nhóm 11
22127401 - Nguyễn Quang Thông
22127298 - Hoàng Trung Nguyên
22127339 - Võ Nhật Phước
22127463 - Nguyễn Anh Vũ

Giáo Viên Hướng Dẫn: Trương Toàn Thịnh

Thành Phố Hồ Chí Minh, Ngày 03 tháng 04 năm 2023

Lời nói đầu

Lời cảm ơn

Mục lục

Lời nói đầu 2
Lời cảm ơn 3
Mục lục 4
Mục lục hình 6
1. Hướng dẫn sử dụng 7
1.1. Hiển thị hình ảnh 7
1.2. Code có highlight (Paste code ít thôi, lấy cái nào quan trọng á) 7
2. Table 7
3. Tổng quan về trò chơi 9
3.1. Giới thiệu về trò chơi 9
3.1.1. Gomoku 9
3.1.2. Mục tiêu đề ra 9
3.1.3. Thông tin chung về trò chơi 9
3.2. Mô tả về các tính năng của game 9
3.2.1. Đa ngôn ngữ 9
3.2.2. Thay đổi Theme(Chủ đề) 9
3.2.3. Lưu thiết lập của người chơi 9
3.2.4. Save/Load game đang chơi, replay game đã chơi xong 9
3.2.5. Chế độ chơi Thường 9
3.2.6. Chế độ chơi Rush 9
3.2.7. Đánh với máy 9
3.2.8. Đánh với người 9
3.2.9. Các hỗ trợ trong lúc chơi game 9
3.2.9.1. Gợi ý 9
3.2.9.2. Nổi bật nước mới đi 9
3.2.9.3. Cảnh báo nước 4 9
3.2.9.4. Hoàn tác nước đi 10
3.2.9.5. Đi nháp 10
3.3. Sơ dồ di chuyển 10
4. Chi tiết các chức năng 10
4.1. Logic 10
4.1.1. Chơi hiệu ứng, nhạc nền 10
4.1.1.1. Hàm PlayAndForget 10
4.1.1.2. Class AudioPlayer 11
4.1.1.3. Static class BackgroundAudioService 12
4.1.2. Đọc, ghi, tìm file 13
4.1.2.1. Các hàm hỗ trợ mở file 13

4.1.2.2. Hàm Ensure 13
4.1.2.3. Hàm Delete 14
4.1.2.4. Hàm GetAllTextFileInDir 14
4.1.3. Ngôn ngữ 15
4.1.3.1. Static class Language 15
4.1.4. Cài đặt 17
4.1.5. Chủ đề 17
4.1.6. Điều hướng trong ứng dụng 17
4.1.7. Đồng hồ 17
4.1.8. Hàm trung gian hỗ trợ vẽ giao diện 17
4.1.9. Nhận biết thắng thua 17
4.1.10. Các tương tác với bàn cờ 17
4.1.11. AI 17
4.2. Giao diện 17
4.2.1. Cài đặt 17
4.2.2. Các màn hình lưu, tải game và replay 17
4.2.3. Màn hình trò chơi chính 17
4.2.4. Các màn hình khác 17
5. Đánh giá thành viên 17
6. Kết luận 17
6.1. Kết quả đạt được 18
6.1.1. Ưu điểm của trò chơi 18
6.1.2. Khuyết điểm của trò chơi 18
6.2. Những gì đã học được 18
6.3. Các kinh nghiệm rút ra 18
6.4. Lí do hoàn thành mục tiêu 18
6.5. Hướng phát triển ứng dụng 18

Mục lục hình

Hình 1. Test fig.....7

1. Hướng dẫn sử dụng

1.1. Hiển thị hình ảnh

```
#figure(
    image("HCMUS_Logo.png", width: 40%),
    caption: [Test fig]
)
```

Nhớ bỏ caption vô không là cái mục lục hình bị lỗi

Figure 1: Test fig

1.2. Code có highlight (Paste code ít thôi, lấy cái nào quan trọng á)

```
    while (1) {
        auto tmp = InputHandle::Get();
        if (tmp == L"b" || tmp == L"B") {
            return NavHost.Back();
        }
        if (tmp == L"\r") {
            system(
                "rundll32 url.dll,FileProtocolHandler "
                "https://github.com/thng292/CaroGame"
            );
        }
    }
```

2. Table

Thành viên	Đánh giá

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do.	Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do.

3. Tổng quan về trò chơi

3.1. Giới thiệu về trò chơi

3.1.1. Gomoku

Nguyên

3.1.2. Mục tiêu đề ra

- Game có nhiều ngôn ngữ, người dùng có thể thêm được ngôn ngữ mới
- Có thể load được các theme(chủ đề) bên ngoài
- Lưu được các thiết lập của người chơi
- Có thể save, load game đang chơi
- Có thể lưu và phát lại các game đã hoàn thành
- Có nhiều chế độ chơi
- Có thể chơi với máy, máy có nhiều mức độ
- Game có thể phát nhạc nền, hiệu ứng. Có thể bật tắt được

3.1.3. Thông tin chung về trò chơi

Nguyên Link souce code, chạy trên nền tảng nào, ...

3.2. Mô tả về các tính năng của game

3.2.1. Đa ngôn ngữ

3.2.2. Thay đổi Theme(Chủ đề)

3.2.3. Lưu thiết lập của người chơi

3.2.4. Save/Load game đang chơi, replay game đã chơi xong

3.2.5. Chế độ chơi Thường

3.2.6. Chế độ chơi Rush

3.2.7. Đánh với máy

3.2.8. Đánh với người

3.2.9. Các hỗ trợ trong lúc chơi game

3.2.9.1. Gợi ý

3.2.9.2. Nổi bật nước mới đi

3.2.9.3. Cảnh báo nước 4

3.2.9.4. Hoàn tác nước đi

3.2.9.5. Đi nháp

3.3. Sơ đồ di chuyển

Phước

4. Chi tiết các chức năng

4.1. Logic

4.1.1. Chơi hiệu ứng, nhạc nền

Các file âm thanh được đặt trong thư mục asset/audio và có thể truy cập bằng các `enum`. Các `enum` được map sang một mảng chứa tên các file âm thanh

```
enum class Sound : char {
    NoSound = 0,
    OnKey,
    Draw,
    Win,
    Lose,
    MenuBGM,
    MenuMove,
    MenuSelect,
    GameBGM,
    WinSound,
    GamePlace,
    GameStart,
    Pause,
    WarningSound
};
```

```
constexpr std::array SoundName{
    L"",
    L"Key.wav",
    L"Draw.mp3",
    L"Win.mp3",
    L"Lose.mp3",
    L"MenuBGM.mp3",
    L"MenuMove.wav",
    L"MenuSelect.wav",
    L"GameBGM.mp3",
    L"WinSound.mp3",
    L"GamePlaceMove.mp3",
    L"GameStart.wav",
    L"Pause.wav",
    L"Warning.mp3"
};
```

4.1.1.1. Hàm PlayAndForget

Hàm này sử dụng API PlaySound để chơi nhạc. Được dùng để chơi những âm thanh ngắn, dung lượng nhỏ dưới 100kb. Khi gọi hàm sẽ tự load file vào memory, chơi và đóng file. Do phải load cả file vào bộ nhớ nên khi chơi có độ delay cao và chỉ có thể mở được file `wav`. Được ứng dụng để chơi các âm thanh liên quan tới giao diện, các âm thanh không quan tâm tới độ trễ. Hàm nằm trong `namespace Audio`, file `Audio.h`, `Audio.cpp`

Interface:

```
    bool PlayAndForget(Sound sound, bool wait)
```

Parameters:

- `Sound` : âm thanh cần chơi
- `wait` :
  - `true` => phát âm thanh một cách đồng bộ (synchronous)
  - `false` => phát âm thanh một cách bất đồng bộ (asynchronous)

Usage:

```
    Audio::PlayAndForget(Audio::Sound::MenuSelect);
```

4.1.1.2. Class AudioPlayer

Class này sử dụng **`Media Control Interface`** (`MCI`) để chơi nhạc, chơi được các file âm thanh định dạng `mp3` và `wav`. Chơi được các file lớn, ít delay do không cần load hết file vào bộ nhớ. Được dùng để chơi nhạc nền hoặc file `mp3`.

Interface:

```
class AudioPlayer {
    // Ngăn copy hay move class
    AudioPlayer(AudioPlayer&&) = delete;
    AudioPlayer(const AudioPlayer&) = delete;
    AudioPlayer& operator=(AudioPlayer&&) = delete;
    AudioPlayer& operator=(const AudioPlayer&) = delete;

    AudioPlayer();
    AudioPlayer(Sound song); // Khởi tạo và mở file

    // Mở file. Có thể dùng để đổi file cần chơi
    int Open(Sound song);
    Sound getCurrentSong() const;

    int Play(bool fromStart = 1, bool repeat = 0) const; // Chơi
    int Pause() const;      // Tạm dừng
    int Resume() const;     // Tiếp tục
    int Stop() const;       // Dừng chơi và trả con trỏ về đầu
    int Close();            // Đóng file đang mở

    ~AudioPlayer();         // Đóng file đang mở
}
```

Parameters:

- `song` : âm thanh cần chơi
- `fromStart` :
  - `true` => chơi từ đầu
  - `false` => chơi tiếp tại vị trí con trỏ

- `repeat`:
  - `true` => lặp lại khi kết thúc

Return:

- Các phương thức sẽ trả về `MCI code` của lệnh MCI tương ứng

Usage:

```
{
    Audio::AudioPlayer player(Audio::Sound::Draw);
    player.play(true, true);
    player.pause();
    player.resume();
    player.close();
}
```

4.1.1.3. Static class BackgroundAudioService

Class này được dùng để chơi nhạc nền, sử dụng class AudioPlayer để chơi nhạc.

Interface:

```
class BackgroundAudioService {
    BackgroundAudioService() = delete;
    static Audio::Sound GetCurrentSong();

    static int ChangeSong(Audio::Sound song) // Đổi nhạc

    static int Play(bool fromStart = 0, bool repeat = 1); // Chơi
    static int Pause();                                   // Tạm dừng
    static int Resume();                                  // Tiếp tục
    // Dừng chơi, trả con trỏ về đầu
    static int Stop();
};
```

Parameters:

- `song`: âm thanh cần chơi
- `fromStart`:
  - `true` => chơi từ đầu
  - `false` => chơi tiếp tại vị trí con trỏ
- `repeat`:
  - `true` => lặp lại khi kết thúc

Usage:

```
{
    BackgroundAudioService::ChangeSong(Audio::Sound::MenuBGM);
    BackgroundAudioService::Play(true, true);
}
```

4.1.2. Đọc, ghi, tìm file

Các hàm nằm trong `namespace FileHandle`, file `FileHandle.h`, `FileHandle.cpp`

4.1.2.1. Các hàm hỗ trợ mở file

Hỗ trợ mở các file văn bản `utf-8`

Interface:

```
    typedef std::filesystem::path fsPath;
    std::wofstream OpenOutFile(const fsPath& filePath);
    std::wifstream OpenInFile (const fsPath& filePath);
```

Parameters:

- `filePath`: đường dẫn đến file cần mở

Usage:

```
#include <string>
{
    std::wstring str = L"Tiếng Việt";
    auto outFile = FileHandle::OpenOutFile("test.txt");
    outFile <<  str;
    outFile.close();
    auto inFile  = FileHandle::OpenInFile ("test.txt");
    infile  >> str;
}
```

4.1.2.2. Hàm Ensure

Dùng để đảm bảo đường dẫn đến file muốn mở có tồn tại, nếu không tồn tại, nếu không tồn tại thì tạo đường dẫn đó.

Interface:

```
    void Ensure(const std::filesystem::path& Dir);
```

Parameters:

- `Dir`: đường dẫn muốn kiểm tra/tạo

Return:

- Các fstream tương ứng với thao tác In/Out

Usage:

```
{
    // Đảm bảo đường dẫn tương đối "asset/language" tồn tại
    FileHandle::Ensure("asset/language");
}
```

4.1.2.3. Hàm Delete

Dùng để xóa file

Interface:

```
    bool Delete(const std::filesystem::path& target)
```

Paramterers:

- target: đường dẫn tới file cần xóa

Return:

- Trả về `true` nếu xóa thành công, `false` khi lỗi

Usage:

```
{
    // Xóa file tmp.cpp
    bool res = FileHandle::Delete("tmp.cpp");
    if (res) {
        std::cout << "Success";
    } else {
        std::cout << "Failed";
    }
}
```

4.1.2.4. Hàm GetAllTextFileInDir

Tìm các file văn bản thuần trong thư mục

Interface:

```
struct FileDetail {
    std::filesystem::path               filePath;
    std::filesystem::file_time_type     lastModified;
};

std::vector<FileHandle::FileDetail>
FileHandle::GetAllTextFileInDir(
    const std::filesystem::path& Dir
);
```

Parameters:

- `Dir`: đường dẫn đến thư mục muốn tìm

Return:

- Trả về một `vector` chứa các thông tin của các file đã tìm được

Usage:

```
{
    // Tìm các file văn bản trong đường dẫn
    // tương đối "asset/language"
    auto files = FileHandle::GetAllTextFileInDir(
        "asset/language"
    );
    for (auto& file:files) {
        std::cout << file.filePath.filename() << '\n';
    }
}
```

4.1.3. Ngôn ngữ

Các văn bản trong trò chơi sẽ được load từ một file riêng, điều này kiến cho phần ngôn ngữ trong game dễ tùy biến và thêm các ngôn ngữ mới. File ngôn ngữ là một file văn bản thuần chứa các nhãn và phần văn bản ngăn cách bởi dấu "=", các nhãn có nằm bên trong cặp ngoặc `[]` là `meta` được dùng để chứa thông tin về file ngôn ngữ

Ví dụ file ngôn ngữ:

```
[LANGUAGE]                  =    English
[LANG_SELECT]               =    Language
ABOUT_DESC                  =    LIST OF GROUP MEMBERS AND SOURCE
CODE LINK
ABOUT_SHORTCUT              =    A
ABOUT_TITLE                 =    About us
ABOUT_US_TITLE              =    About us
```

Các phần văn bản sẽ được truy xuất thông qua nhãn tương ứng. Các phần liên quan tới ngôn ngữ nằm trong file `Language.h` và `Language.cpp`

4.1.3.1. Static class Language

Hàm chứa các phương thức và các văn bản ngôn ngữ

Interface:

```
typedef std::unordered_map<std::wstring, std::wstring> Dict;
typedef std::filesystem::path fsPath;

struct LanguageOption {
  Dict meta;
  fsPath path;
};

class Language {
  static Dict languageDict;

public:
  Language() = delete;

  // Chỉ đọc các phần thông tin về ngôn ngữ
  static Dict ExtractMetaFromFile(const fsPath& filePath);

  // Load file ngôn ngữ
  static void LoadLanguageFromFile(const fsPath& filePath);

  // Tìm các file ngôn ngữ
  static std::vector<LanguageOption>
  DiscoverLanguageFile(const fsPath& dirPath);

  // Truy xuất văn bản bằng nhãn
  static const std::wstring&
  GetString(const std::wstring& Label);

  static const std::wstring&
  GetMeta(const std::wstring& Label);
};
```

Parameters:

- `filePath`: đường dẫn tới file cần mở
- `dirPath`: đường dẫn tới thư mục cần tìm
- `Label`: nhãn của văn bản cần lấy

Usage:

```
{
    Language::LoadLanguageFromFile("asset/language/en.txt");
    std::cout << Language::GetMeta(L"[LANGUAGE]");
    std::cout << Language::GetMeta(L"ABOUT_TITLE");
}
```

4.1.4. Cài đặt
Thông

4.1.5. Chủ đề
Thông

4.1.6. Điều hướng trong ứng dụng
Thông

4.1.7. Đồng hồ
Thông

4.1.8. Hàm trung gian hỗ trợ vẽ giao diện
Thông

4.1.9. Nhận biết thắng thua
Vũ

4.1.10. Các tương tác với bàn cờ
Vũ

4.1.11. AI
Vũ

sdhfiuashksj

4.2. Giao diện

4.2.1. Cài đặt
Thông

4.2.2. Các màn hình lưu, tải game và replay
Thông

4.2.3. Màn hình trò chơi chính
Vũ

4.2.4. Các màn hình khác

5. Đánh giá thành viên

6. Kết luận

6.1. Kết quả đạt được

6.1.1. Ưu điểm của trò chơi

- Có thể thêm nhiều ngôn ngữ và theme vào trò chơi
- Có nhạc hay, hiệu ứng sống động
- Có chế độ tính thời gian
- AI chạy tương đối tốt, đánh nhanh
- Lối chơi đa dạng
- Có nhiều nhân vật ngộ nghĩnh
- Có nhiều tính năng hỗ trợ khi chơi game
- Có thể xem lại trận đấu đã chơi
- Màn hình save/load có khả năng tương tác tốt
- Hướng dẫn dễ hiểu, có gợi ý ở mỗi màn hình

6.1.2. Khuyết điểm của trò chơi

Phước

6.2. Những gì đã học được

- Cách làm việc nhóm với git và GitHub
- Cách sử dụng các tính năng mới của C++
- Cách sử dụng các tính năng liên quan tới đo hiệu năng, format code và debug trong Visual Studio
- Cách làm việc nhóm hiệu quả
- Cách lên kế hoạch, phân chia công việc
- Học được cách vẽ biểu đồ di chuyển cho ứng dụng

6.3. Các kinh nghiệm rút ra

- Không nên viết code mà không thiết kế trước
- Nên viết code theo một quy chuẩn nhất định và đồng bộ trong 1 dự án

6.4. Lí do hoàn thành mục tiêu

Nguyên

6.5. Hướng phát triển ứng dụng

- Có thể chơi 2 người qua mạng lan
- Thêm nhiều ngôn ngữ mới
- Thêm nhiều chủ đề hơn
- Hiện lợi thế của 2 bên
- Đưa game lên nhiều nền tảng khác