

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA HỆ THỐNG THÔNG TIN



BÁO CÁO
NGÔN NGỮ LẬP TRÌNH JAVA
ĐỀ TÀI:
PHẦN MỀM NGHE NHẠC TRÊN MÁY TÍNH

Giảng viên hướng dẫn:

Thầy Huỳnh Tuấn Anh

Lớp:

SE330.J21.PMCL

Sinh viên thực hiện:

HOÀNG DUY PHƯƠNG 16520965

Tp. Hồ Chí Minh, tháng 6/2019

MỤC LỤC

| | |
|--|-----------|
| PHẦN 1: GIỚI THIỆU | 4 |
| I. GIỚI THIỆU ĐỀ TÀI..... | 4 |
| II. Mô tả yêu cầu chương trình..... | 4 |
| PHẦN 2: THIẾT KẾ CHI TIẾT | 5 |
| I. Thiết kế lớp | 5 |
| 1.1 Biểu đồ lớp | 5 |
| 1.2 Thiết kế chi tiết lớp..... | 5 |
| PHẦN 3: CHƯƠNG TRÌNH MINH HOẠ..... | 7 |
| I. Xây dựng chương trình..... | 7 |
| 1.1 Giới thiệu về ngôn ngữ lập trình Java | 7 |
| 1.2 Các ứng dụng Java..... | 8 |
| 1.3 Quá trình lập trình tuân thủ quy ước lập trình :..... | 10 |
| II. Kiểm thử đơn vị..... | 10 |
| III. Kết quả chương trình | 19 |
| IV. Chức năng các button trên Tab Chơi Nhạc : | 23 |
| PHẦN 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN..... | 24 |
| TÀI LIỆU THAM KHẢO..... | 25 |
| PHỤ LỤC | 26 |

LỜI NÓI ĐẦU

Trong thời đại khoa học công nghệ đang phát triển như vũ bão hiện nay, các thiết bị như điện thoại máy tính đã được ứng dụng vào mọi mặt của đời sống, giải trí của con người. Khi máy tính mới ra đời nó chỉ để giúp con người tính toán số học, nhưng đến ngày nay nó còn giúp con người giải trí: chơi game, lướt web đọc báo, xem phim và nghe nhạc. Ta đã biết đến phần mềm chơi nhạc Windows Media Player của Windows – giao diện và tính năng rất phù hợp với người dùng. Để hiểu sâu về các ứng dụng nghe nhạc nói chung em đã chọn đề tài này.

Để hoàn thành được project này, em xin được gửi lời cảm ơn chân thành đến thầy đã hết lòng giúp đỡ, hướng dẫn, chỉ dạy tận tình để em hoàn thành được đề tài này.

PHẦN 1: GIỚI THIỆU

I. GIỚI THIỆU ĐỀ TÀI

- Chương trình nghe nhạc (**HDPPlayer**). Lấy ý tưởng từ chương trình nghe nhạc nổi tiếng Windows Media Player trên HDH Windows.
- Sử dụng ngôn ngữ Java để xây dựng chương trình.

II. Mô tả yêu cầu chương trình

Chương trình nghe nhạc (**HDPPlayer**) như chương trình Window Media Player:

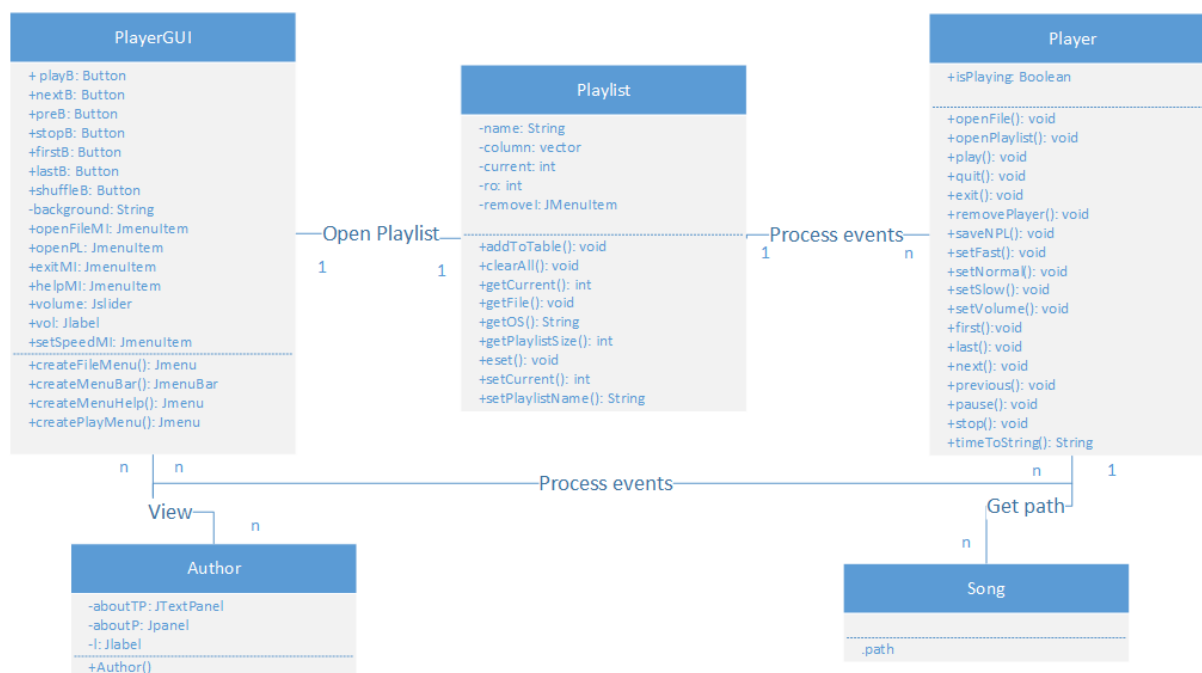
- Tạo ứng dụng cho phép nghe nhạc từ các bài hát được lưu trữ từ các thư mục khác nhau tương tự giao diện media player của window
- Thanh điều hướng, cho phép người sử dụng có thể sử dụng các chức năng : Next, Last, First, Previous, Autoplay.
- Thanh điều chỉnh mức độ âm thanh có thể dùng.

PHẦN 2: THIẾT KẾ CHI TIẾT

I. Thiết kế lớp

1.1 Biểu đồ lớp

Vẽ và giới thiệu về các lớp, mối quan hệ trong Biểu đồ lớp.



Hình 1.1: Biểu đồ lớp

Hình trên là biểu đồ lớp thể hiện các lớp chính và các mối quan hệ của các lớp đó. Biểu đồ trên gồm 3 nhóm lớp chính là: Lớp giao diện : PlayerGUI và Author, Playlist, lớp nghiệp vụ: Player, lớp thực thể là Song. Mỗi liên hệ giữa các lớp: 1 lớp PlayerGUI chỉ có thể open playlist từ 1 lớp playlist. Một lớp Playlist có thể gửi nhiều sự kiện đến lớp Player và Lớp Player có thể xử lý nhiều sự kiện nhận được từ lớp Playlist. Sau đó lớp player sẽ nhận đường dẫn đến thư mục chứa bài hát cần phát trong lớp Song.

1.2 Thiết kế chi tiết lớp

Mô tả chi tiết về các lớp quan trọng:

Thiết kế chi tiết cho các lớp: Các thuộc tính, phương thức, mối quan hệ, ràng buộc.

❖ Lớp giao diện PlayerGUI gồm 4 phương thức chính là:

- createFileMenu(): thuộc lớp JMenuItem trong Java Swing mà lớp PlayerGUI kế thừa.

-
- Lớp này bao gồm thuộc tính là `openFileMI (JMenuItem)` , `openPL(JMenuItem)`, `savePL(JMenuItem)` và `exitMI (JMenuItem)`.
- `createPlayMenu()`: thuộc lớp `Jmenu` trong Java Swing mà lớp `PlayerGUI` kế thừa.
- Lớp này gồm thuộc tính `setSpeedPlay (JMenuItem)`, `changeBackground (JMenuItem)`.
- `createMenuHelp()`: kế thừa từ lớp `Author`.
 - `createButton()`: thuộc lớp `JButton` trong Java Swing mà lớp `PlayerGUI` kế thừa. Bao gồm 7 thuộc tính là: `playB`, `stopB`, `nextB`, `preB`, `firstB`, `lastB`, `shuffleB`.
- ❖ Lớp giao diện `Playlist` gồm 4 phương thức chính là:
- `addToTable()`: kế thừa `Jtable` của Java Swing. Phương thức này sẽ tạo ra các hàng từ danh sách bài hát thành 1 bảng chứa danh sách đó.
 - `getCurrent()`: có thuộc tính là `current`. Trả về giá trị nguyên của vị trí bài hát đang phát hiện tại.
 - `getOS()`: có thuộc tính `name` kiểu trả về là `String`. Hàm sẽ lấy tên hệ điều hành đang chạy chương trình (`windows` hoặc `Linux`).
 - `getPlaylistSize()`: phương thức này trả về kiểu `int` chỉ số lượng bài hát trong danh sách `playlist`.
- ❖ Lớp nghiệp vụ `Player` gồm các phương thức chính sau:
- `openFile()`: trả về đường dẫn chứa file nhạc.
 - `openPlaylist()`: trả về đường dẫn chứa `playlist`.
 - `play()`: dùng để chạy các file nhạc với các đường dẫn trên.
 - `exit()`: thoát khỏi chương trình .
 - `setSlow()`: chế độ chạy nhạc với tốc độ chậm.
 - `setFast()`: chạy nhạc với tốc độ nhanh.
 - `setNormal()`: chạy nhạc với tốc độ bình thường.
 - `setVolume()`: điều chỉnh âm lượng bài hát.
 - `next()`: có thuộc tính `isPlaying` ,chức năng chạy bài kế tiếp.
 - `previous()`: có thuộc tính `isPlaying` ,chức năng chạy bài trước đó.
 - `last()`: có thuộc tính `isPlaying` ,chức năng chạy bài cuối cùng của `playlist`.
 - có thuộc tính `isPlaying` ,chức năng
 - `pause()`: có thuộc tính `isPlaying` ,chức năng tạm dừng bài hát đang phát.
 - `stop()`: có thuộc tính `isPlaying` ,chức năng dừng hẳn bài hát đang phát.
 - `timeToString()`: chuyển thời gian sang kiểu `String`.

PHẦN 3: CHƯƠNG TRÌNH MINH HOẠ

I. Xây dựng chương trình

Chương trình nghe nhạc được xây dựng bằng ngôn ngữ lập trình java.

1.1 Giới thiệu về ngôn ngữ lập trình Java

❖ Java là gì?

Java là ngôn ngữ lập trình hướng đối tượng (tựa C++) do Sun Microsystem đưa ra vào giữa thập niên 90. Chương trình viết bằng ngôn ngữ lập trình java có thể chạy trên bất kỳ hệ thống nào có cài máy ảo java (Java VirtualMachine).

❖ .Lịch sử phát triển của ngôn ngữ lập trình Java

Ngôn ngữ lập trình Java do James Gosling và các công sựcủa Công ty Sun Microsystem phát triển. Đầu thập niên 90, Sun Microsystem tập hợp các nhà nghiên cứu thành lập nên nhóm đặt tên là Green Team. Nhóm Green Team có trách nhiệm xây dựng công nghệ mới cho ngành điện tử tiêu dùng. Để giải quyết vấn đề này nhóm nghiên cứu phát triển đã xây dựng một ngôn ngữ lập trình mới đặt tên là Oak tương tự như C++ nhưng loại bỏ một số tính năng nguy hiểm của C++ và có khả năng chạy trên nhiều nền phần cứng khác nhau. Cùng lúc đó world wide web bắt đầu phát triển và Sun đã thấy được tiềm năng của ngôn ngữ Oak nên đã đầu tư cải tiến và phát triển. Sau đó không lâu ngôn ngữ mới vươt tên gọi là Java ra đời và được giới thiệu năm 1995. Java là tên gọi của một hòn đảo ở Indonexia, Đây là nơi nhóm nghiên cứu phát triển đã chọn để đặt tên cho ngôn ngữ lập trình Java trong một chuyến đi tham quan và làm việc trên hòn đảo này. Hòn đảo Java này là nơi rất nổi tiếng vướnhiều khu vườn trồng cafe, đó chính là lý do chúng ta thường thấy biểu tượng ly cafe trong nhiều sản phẩm phần mềm , công cụ lập trình Java của Sun cũng như một số hãng phần mềm khác đưa ra.

❖ Một số đặc điểm nổi bật của ngôn ngữ lập trình Java

❖ Máy ảo Java (JVM - Java VirtualMachine)

Tất cả các Chương trình muốn thực thi được thì phải được Biên dịch ra mã máy. Mã máy của từng kiến trúc CPU của mỗi máy tính là khác nhau (tập lệnh mã máy của CPU Intel, CPU Solarix, CPU Macintosh ... là khác nhau), vì vậy trước đây một Chương trình sau khi được biên dịch xong chỉ có thể chạy được Trên một kiến trúc CPU cụ thể nào đó. Đối với CPU Intel chúng ta có thể chạy cáchệ đi ều hành như Microsoft Windows, Unix, Linux, OS/2, ... Chương trình thực thi được trên Windows được biên dịch dưới dạng file có đuôi .EXE còn trên Linux thì được biên dịch dưới dạng file có đuôi .ELF, vì vậy trước đây một Chương trình chạy được trên Windows muốn chạy được trên hệ điều hành khác như Linux chẳng hạn thì phải chỉnh sửa và biên dịch lại . Ngôn ngữ lập trình Java ra đời , nhờ vào máy ảo Java mà khó khăn nêu trên đã được khắc phục.Một Chương trình viết bằng ngôn ngữ lập trình Java sẽ được biên dịch ra mã của máy ảo java (mã java bytecode). Sau đó máy ảo Java chịu trách nhiệm chuyển mã java bytecode thành mã máy tương ứng. Sun Microsystem chịu trách nhiệm phát triển các máy ảo Java chạy trên các hệ điều hành trên các kiến trúc CPU khác nhau.

❖ Thông dịch: Java là một ngôn ngữ lập trình vừa biên dịch vừa thông dịch.

Chương trình nguồn viết bằng ngôn ngữ lập trình Java có đuôi *.java đầu tiên

được biên dịch thành tập tin có đuôi *.class và sau đó sẽ được trình thông dịch thông dịch thành mã máy.

❖ **Độc lập nền:**

Một Chương trình viết bằng ngôn ngữ Java có thể chạy trên nhiều máy tính có hệ điều hành khác nhau (Windows, Unix, Linux, ...) miễn sao ở đó có cài đặt máy ảo java (Java VirtualMachine). Viết một lần chạy mọi nơi (write once run anywhere).

❖ **Hướng đối tượng:**

Hướng đối tượng trong Java tương tự như C++ nhưng Java Là một ngôn ngữ lập trình hướng đối tượng hoàn toàn. Tất cả mọi thứ đề cập đến trong Java đều liên quan đến các đối tượng được định nghĩa trước, thậm chí hàm chính của một Chương trình viết bằng Java (đó là hàm main) cũng phải đặt bên trong một lớp. Hướng đối tượng trong Java không có tính đa kế thừa (multi inheritance) như trong C++ mà thay vào đó Java đưa ra khái niệm interface để hỗ trợ tính đa kế thừa. Vấn đề này sẽ được bàn chi tiết trong Chương 3.

❖ **Đa nhiệm - đa luồng (MultiTasking – Multi threading):** Java hỗ trợ lập trình đa nhiệm, đa luồng cho phép nhiều tiến trình, tiểu trình có thể chạy song song cùng một thời điểm và tương tác với nhau.

❖ **Khả chuyển (portable):**

Chương trình ứng dụng viết bằng ngôn ngữ Java chỉ cần chạy được trên máy ảo Java là có thể chạy được trên bất kỳ máy tính, hệ điều hành nào có máy ảo Java. “Viết một lần, chạy mọi nơi” (Write Once, Run Anywhere).

❖ **Hỗ trợ mạnh cho việc phát triển ứng dụng :**

Công nghệ Java phát triển mạnh mẽ nhờ vào “đại gia Sun Microsystem” cung cấp nhiều công cụ, thư viện lập trình phong phú hỗ trợ cho việc phát triển nhiều loại hình ứng dụng khác nhau cụ thể như: J2SE (Java 2 Standard Edition) hỗ trợ phát triển những ứng dụng đơn, ứng dụng client-server; J2EE (Java 2 Enterprise Edition) hỗ trợ phát triển các ứng dụng thương mại, J2ME (Java 2 Micro Edition) hỗ trợ phát triển các ứng dụng trên các thiết bị di động, không dây, ...

1.2 Các ứng dụng Java

❖ **Java và ứng dụng Console**

ứng dụng Console là ứng dụng nhập xuất ở chế độ văn bản tương tự như màn hình Console của hệ điều hành MS-DOS. Loại Chương trình ứng dụng này thích hợp với những ai bước đầu làm quen với ngôn ngữ lập trình java. Các ứng dụng kiểu Console thường được dùng để minh họa các ví dụ cơ bản liên quan đến cú pháp ngôn ngữ, các thuật toán, và các Chương trình ứng dụng không cần thiết đến giao diện người dùng đồ họa.

```
class HelloWorld
{
    public static void main(String[] args)
    {
        System.out.println("\nHello World");
    }
}
```

❖ **Java và ứng dụng Applet**

Java Applet là loại ứng dụng có thể nhúng và chạy trong trang Web của trình duyệt web. Từ khi internet ra đời, Java Applet cung cấp một khả năng lập trình mạnh mẽ cho các trang web. Nhưng gần đây khi các Chương trình duyệt

web đã phát triển với khả năng lập trình bằng VB Script, JavaScript, HTML, DHTML, XML, ... cùng với sự cạnh tranh khốc liệt của Microsoft và Sun đã làm cho Java Applet lu mờ. Và cho đến bây giờ gần như các lập trình viên đều không còn “mặn mà” với Java Applet nữa. (trình duyệt IE đi kèm trong phiên bản Windows 2000 đã không còn mặc nhiên hỗ trợ thực thi một ứng dụng Java Applet). Java và phát triển ứng dụng Desktop dùng AWT và JFC.

Việc phát triển các Chương trình ứng dụng có giao diện người dùng đồ họa trực quan giống như những Chương trình được viết dùng ngôn ngữ lập trình VC++ hay Visual Basic đã được Java giải quyết bằng thư viện AWT và JFC. JFC là thư viện rất phong phú và hỗ trợ mạnh mẽ hơn nhiều so với AWT. JFC giúp cho người lập trình có thể tạo ra một giao diện trực quan của bất kỳ ứng dụng nào.

❖ Java và phát triển ứng dụng Web

Java hỗ trợ mạnh mẽ đối với việc phát triển các ứng dụng Web thông qua công nghệ J2EE (Java 2 Enterprise Edition). Công nghệ J2EE hoàn toàn có thể tạo ra các ứng dụng Web một cách hiệu quả không thua kém công nghệ .NET mà Microsoft đang quảng cáo.

❖ Java và phát triển các ứng dụng nhúng

Java Sun đưa ra công nghệ J2ME (The Java 2 Platform, Micro Edition J2ME) hỗ trợ phát triển các Chương trình, phần mềm nhúng. J2ME cung cấp một môi trường cho những Chương trình ứng dụng có thể chạy được trên các thiết bị cá nhân như: điện thoại di động, máy tính bỏ túi PDA hay Palm, cũng như các thiết bị nhúng khác. Bạn có thể tìm hiểu chi tiết hơn về công nghệ J2ME tại địa chỉ : <http://java.sun.com/j2me/>

❖ Dịch và thực thi một chương trình viết bằng Java

Việc xây dựng, dịch và thực thi một Chương trình viết bằng ngôn ngữ lập trình Java có thể tóm tắt qua các bước sau:

- Viết mã nguồn: dùng một Chương trình soạn thảo nào

đấy (Notepad hay Jcreator chẳng hạn) để viết mã nguồn và lưu lại với tên có đuôi “.java”

- Biên dịch ra mã máy ảo: dùng trình biên dịch javac để biên dịch mã nguồn “.java” thành mã của máy ảo (Java bytecode) có đuôi “.class” và lưu lên đĩa

- Thông dịch và thực thi: ứng dụng được load vào bộ nhớ, thông dịch và thực thi dùng trình thông dịch Java thông qua lệnh “java”.

+ Đưa mã Java bytecode vào bộ nhớ: đây là bước “loading”. Chương trình phải được đặt vào trong bộ nhớ trước khi thực thi. “Loader” sẽ lấy các files chứa mã Java bytecode có đuôi “.class” và nạp chúng vào bộ nhớ.

+ Kiểm tra mã Java bytecode: trước khi trình

Thông dịch chuyển mã bytecode thành mã máy để thực thi thì các mã bytecode phải được kiểm tra tính hợp lệ.

+ Thông dịch & thực thi: cuối cùng dưới sự điều khiển của CPU và trình thông dịch tải mỗi thời điểm sẽ có một mã bytecode được chuyển sang mã máy và thực thi.

Java là ngôn ngữ vừa biên dịch, vừa thông dịch. Đầu tiên, mã chương trình nguồn được biên dịch bằng chương trình javac để chuyển thành dạng Bytecode. Sau đó, được thực thi trên từng máy cụ thể nhờ thông dịch trong máy ảo JVM. Mục tiêu của các nhà thiết kế Java.

1.3 Quá trình lập trình tuân thủ quy ước lập trình :

<1> Đặt tên cho các gói (package) nên bằng chữ in thường toàn bộ:

Ex: gui, main

<2> Tên biến phải bắt đầu bằng một ký tự in thường, các từ tiếp theo được bắt đầu bằng một ký tự in hoa:

Ex: fileM, openMI...

<4> Tên hằng phải đặt toàn bộ là chữ in hoa, các từ tách biệt nhau bởi ký tự gạch dưới “_”.

<5> Tên của các phương thức phải là động từ bắt đầu bằng 1 ký tự in thường và các từ tiếp sau được viết rõ ràng bởi các từ bắt đầu bằng 1 ký tự in hoa:

Ex: getName(), getCurrent()...

<6> Tên các biến cục bộ của lớp nên kết thúc bằng hậu tố “_”

```
class PlayerGUI{  
    public String name_; ...  
}
```

<7> Tất cả các tên nên được viết bằng Tiếng Anh

<8> Từ khóa “set/get” phải được đặt trong các phương thức truy cập trực tiếp đến thuộc tính:

Ex: getName(), setCurrent()...

<9> Tiền tố “is” nên được sử dụng trong các phương thức, hoặc các biến kiểu boolean:

Ex: isPlaying, isOpen...

<10> Các biến JFC (Java Swing) nên được đặt hậu tố là kiểu đối tượng:

Ex: openFile, nameTextField, leftScrollbar, mainPanel, openPlaylist...

II. Kiểm thử đơn vị

- Kịch bản chạy thử chương trình .

Đây là Class Player chứa các hàm xử lý chính của chương trình:

```
public void setVolumn(float volume) {  
    try{  
        player.getGainControl().setLevel(volume);  
    }catch(Exception e){  
  
    }  
}
```

```
public void openFile() {  
    cs = new JFileChooser();  
    File[] list = null;  
    cs.setFileFilter(new FileNameExtensionFilter("Audio File (*.mp3,  
*.wav)", "mp3", "wav"));  
    cs.setMultiSelectionEnabled(true);  
    if (cs.showOpenDialog(null) == JFileChooser.APPROVE_OPTION) {  
        PlayerGUI.playlist.reset();  
        removePlayer();  
        Playlist.fileList = new LinkedList<File>();  
    }  
}
```

```

        list = cs.getSelectedFiles();
        for (int i = 0; i < list.length; i++) {
            PlayList.fileList.add(list[i]);
            PlayerGUI.playList.addToTable(list[i]);
        }
        PlayerGUI.playList.setCurrent(0);
        PlayerGUI.playList.setRowSelectionInterval(0, 0);
    }
    play();
    isPlaying = true;
}

```

```

public void removePlayer() {
    if (player != null) {
        player.stop();
        player.removeController(player);
        player.close();
        player = null;
    }
}

```

```

public void openPlayList() {
    File list = null;
    BufferedReader reader;
    String line;
    File f;
    cs = new JFileChooser();
    cs.setFileFilter(new FileNameExtensionFilter("PlayList File (*.m3u)",
"m3u"));
    cs.setMultiSelectionEnabled(false);
    if (cs.showOpenDialog(null) == JFileChooser.APPROVE_OPTION) {
        removePlayer();
        try {
            PlayerGUI.playList.reset();
            PlayList.fileList = new LinkedList<File>();
            list = cs.getSelectedFile();
            reader = new BufferedReader(new InputStreamReader(new
FileInputStream(list), "UTF-8"));
            reader.readLine();
            while ((line = reader.readLine()) != null) {
                if(line.startsWith("#")){

                }else{
                    f = new File(line);
                    PlayList.fileList.add(f);
                    PlayerGUI.playList.addToTable(f);
                    reader.readLine();
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

```

```

        }
    }

    PlayerGUI.playList.setCurrent(0);
    PlayerGUI.playList.setRowSelectionInterval(0, 0);
    } catch (IOException ex) {
        JOptionPane.showMessageDialog(null, ex.getMessage(), "Message",
JOptionPane.ERROR_MESSAGE, null);
    }
}
play();
isPlaying = true;
}

public void saveNPL() {
    cs = new JFileChooser();
    cs.setFileFilter(new FileNameExtensionFilter("PlayList File (*.m3u)",
"m3u"));
    BufferedWriter writer;
    if (cs.showSaveDialog(null) == JFileChooser.APPROVE_OPTION) {
        try {
            File f = cs.getSelectedFile();
            int r = 5;
            if(f.exists()){
                r = JOptionPane.showConfirmDialog(playerGUI, "File "+
f.getName() + " already exist! Overwrite?", "Question ?",
JOptionPane.YES_NO_OPTION, JOptionPane.QUESTION_MESSAGE);
            }
            if(!f.exists() || r == JOptionPane.YES_OPTION){
                if (f.getName().endsWith("m3u")) {
                    writer = new BufferedWriter(new OutputStreamWriter(new
FileOutputStream(f), "UTF-8"));
                } else {
                    f = new File(f.getPath() + ".m3u");
                    writer = new BufferedWriter(new OutputStreamWriter(new
FileOutputStream(f), "UTF-8"));
                }

                try {
                    writer.write("#EXTM3U");
                    writer.newLine();
                    for (int i = 0; i < PlayList.fileList.size(); i++) {
                        writer.write("#EXTINF:0,"+PlayList.fileList.get(i).getPath());
                        writer.newLine();
                        writer.write(PlayList.fileList.get(i).getPath());
                        writer.newLine();
                        writer.newLine();
                    }
                }
            }
        }
    }
}

```

```

        writer.close();
    } catch (IOException ex) {
        errorMessage(ex.getMessage());
    }
} else {
    JOptionPane.showMessageDialog(playerGUI, "No overwrite. No
file selected.", "Message", JOptionPane.INFORMATION_MESSAGE);
}
} catch (UnsupportedEncodingException ex) {
    errorMessage(ex.getMessage());
} catch (FileNotFoundException ex) {
    errorMessage(ex.getMessage());
}
}
}

```

```

public void play() {
    URL fileToPlay;
    File filePlay = null;
    try {
        int position = PlayerGUI.playList.getCurrent();//vt bh hien tai
        filePlay = PlayerGUI.playList.getFile(position);
        file = new File(filePlay.getPath());//chon file
        if (player == null) {//bat bai hat
            try {
                fileToPlay = new URL("file:/" + file.getPath());
                player = Manager.createPlayer(fileToPlay);
                player.addControllerListener(this);
                PlayerGUI.progress.setString("Realizing...");
                player.realize();
                isPlaying = true;

                } catch (MalformedURLException ex) {
                    errorMessage(ex.getMessage());
                } catch (NoPlayerException ex) {
                    errorMessage(ex.getMessage());
                } catch (IOException ex) {
                    errorMessage(ex.getMessage());
                }
            } else
            if (player != null) {
                if (!isPlaying) {
                    try {
                        player.start();
                        isPlaying = true;
                    } catch (Exception e) {
                        errorMessage(e.getMessage());
                    }
                }
            }
        }
    }
}

```

```

        PlayerGUI.progress.setString("Playing : " + file.getName());
    } else {
    }
}
if(isPlaying){
    playerGUI.playB.setIcon(new ImageIcon("images/pause.jpg"));
    playerGUI.playB.setActionCommand("Pause");

}

    } catch (Exception e) {

    }
}

public void setSlow() {
    player.stop();
    player.setRate(0.8f);
    if(isPlaying){
        player.start();
    }
}

public void setNormal() {
    player.stop();
    player.setRate(1f);
    if(isPlaying){
        player.start();
    }
}

public void setFast() {
    player.stop();
    player.setRate(1.5f);
    if(isPlaying){
        player.start();
    }
}

public void pause() {
    if (isPlaying) {
        try {
            player.stop();
            PlayerGUI.progress.setString("Pause");
            isPlaying = false;
            playerGUI.playB.setIcon(new ImageIcon("images/play.jpg"));

```

```

        playerGUI.playB.setActionCommand("Play");

        } catch (Exception e) {
        }
    } else {
    }
}

public void stop() {
    if (player != null) {
        player.removeControllerListener(this);
        player.stop();
        player.close();
        player = null;
    }
    if (playThread != null) {
        playThread = null;
    }

    playerGUI.playB.setIcon(new ImageIcon("images/play.jpg"));
    playerGUI.playB.setActionCommand("Play");

    PlayerGUI.progress.setValue(0);
    PlayerGUI.progress.setString("");
    PlayerGUI.timeL.setText("00:00:00 / 00:00:00");
}

public void exit() {
    if (player != null) {
        if (player != null) {
            player.removeControllerListener(this);
            player.stop();
            player.close();
            player = null;
        }
        System.exit(0);
    } else {
        removePlayer();
        System.exit(0);
    }
}

public void quit(){
    int i = JOptionPane.showConfirmDialog(null, "Do you want to exit?",
    "JPlayer", JOptionPane.YES_NO_OPTION,
    JOptionPane.QUESTION_MESSAGE);
    if (i == JOptionPane.NO_OPTION) {

```

```
playerGUI.setDefaultCloseOperation(JFrame.DO_NOTHING_ON_CLOSE);
    } else if (i == JOptionPane.YES_OPTION) {
        exit();
    }
}
```

```
public void next() {
    try{
        int position = PlayerGUI.playList.getCurrent();

        if (position == PlayerGUI.playList.getRowCount() - 1) {
            position = 0;
            PlayerGUI.playList.setCurrent(position);
            stop();
        } else if (position < PlayerGUI.playList.getRowCount()) {
            position++;
            stop();
        }
        PlayerGUI.playList.setCurrent(position);
        PlayerGUI.playList.setRowSelectionInterval(position, position);
        if (player != null) {
            stop();
        }
        GUI.PlayerGUI.normal.setSelected(true);

        play();

    }catch (Exception e){

    }

}
```

```
public void previous() {
    try {
        int position = PlayerGUI.playList.getCurrent();
        if (position == 0) {
            position = PlayerGUI.playList.getRowCount() - 1;
            stop();
        } else if (position > 0) {
            position--;
            stop();
        }
        PlayerGUI.playList.setCurrent(position);
        PlayerGUI.playList.setRowSelectionInterval(position, position);
        if (player != null) {
            stop();
        }
    }
}
```

```

        }
        GUI.PlayerGUI.normal.setSelected(true);
        play();
    }catch(Exception e){

    }
}
public void shuffle() {
    try{
        int position = PlayerGUI.playList.getCurrent();

        if (position == PlayerGUI.playList.getRowCount() - 1) {
            position = 0;
            PlayerGUI.playList.setCurrent(position);
            stop();
        } else if (position < PlayerGUI.playList.getRowCount()) {

            position=(int)(Math.random()*(PlayerGUI.playList.getRowCount()-1));

            stop();
        }
        PlayerGUI.playList.setCurrent(position);
        PlayerGUI.playList.setRowSelectionInterval(position, position);
        if (player != null) {
            stop();
        }
        GUI.PlayerGUI.normal.setSelected(true);

        play();

    }catch (Exception e){

    }

}
public void first() {
    try{
        int position = PlayerGUI.playList.getCurrent();

        if (position !=0) {
            position = 0;
            PlayerGUI.playList.setCurrent(position);
            stop();
        }
        PlayerGUI.playList.setCurrent(position);
        PlayerGUI.playList.setRowSelectionInterval(position, position);
        if (player != null) {
            stop();

```

```

    }
    GUI.PlayerGUI.normal.setSelected(true);

    play();

} catch (Exception e){

}

}

public void last() {
    try{
        int position = PlayerGUI.playList.getCurrent();

        if (position != PlayerGUI.playList.getRowCount()-1 ) {
            position = PlayerGUI.playList.getRowCount()-1;
            PlayerGUI.playList.setCurrent(position);
            stop();
        }
        PlayerGUI.playList.setCurrent(position);
        PlayerGUI.playList.setRowSelectionInterval(position, position);
        if (player != null) {
            stop();
        }
        GUI.PlayerGUI.normal.setSelected(true);

        play();

    } catch (Exception e){

    }

}

public String timeToString(double d) {
    int hour = (int) (d / 3600);
    int scR = (int) (d - hour * 3600);
    int minu = (int) (scR / 60);
    scR = (int) (scR - minu * 60);
    String h = "", m = "", s = "";
    if (hour < 10) {
        h = "0" + hour;
    } else {
        h = "" + hour;
    }
    if (minu < 10) {
        m = "0" + minu;
    } else {

```

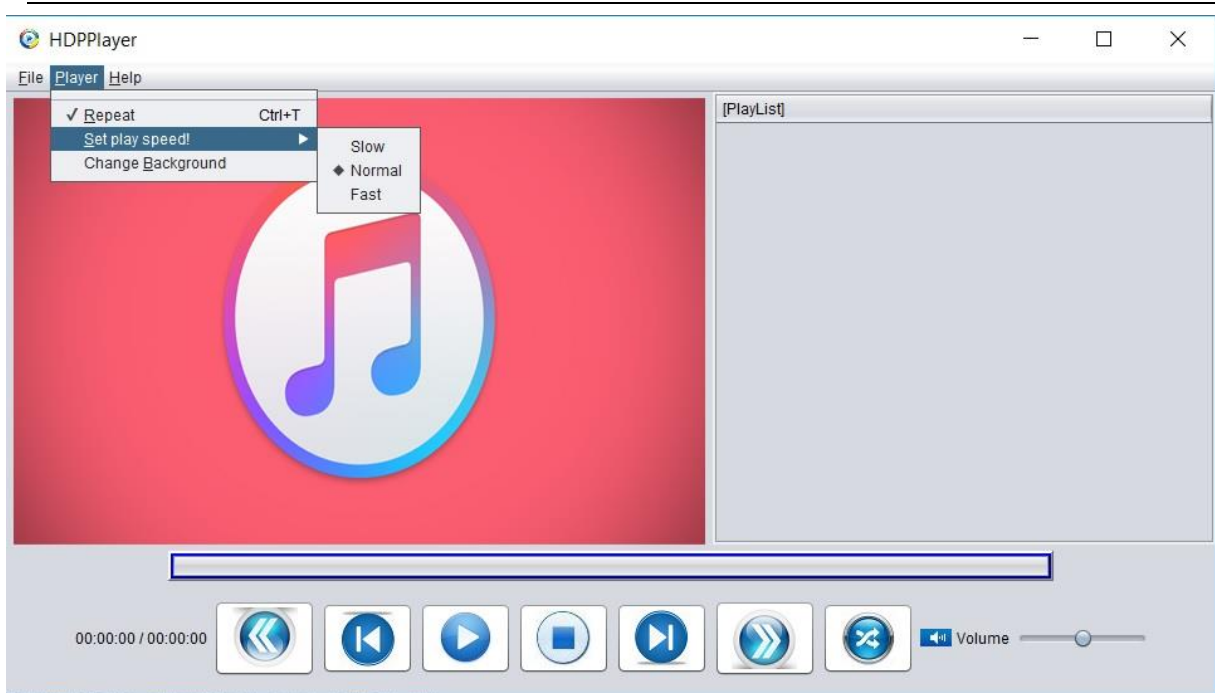
```
        m = "" + minu;
    }
    if (scR < 10) {
        s = "0" + scR;
    } else {
        s = "" + scR;
    }
    return h + ":" + m + ":" + s;
}

    int position = PlayerGUI.playList.getCurrent();
    if(position == (PlayerGUI.playList.getPlayListSize() - 1)){
        if(PlayerGUI.repeatMI.isSelected()){
            next();
        }else{
            stop();
        }
    } else{ next();
    }
}
```

III. Kết quả chương trình

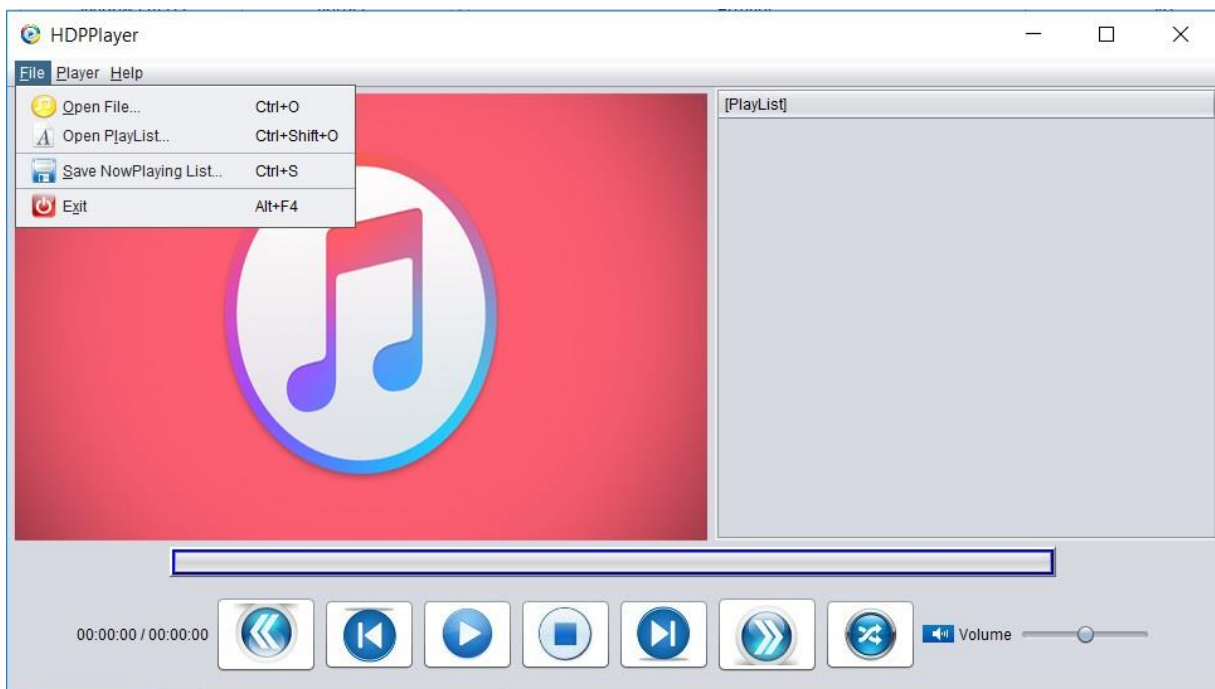
Dưới đây là màn hình chức năng chính của chương trình

- Giao diện chính của chương trình gồm những button Play, Pause, Next, Stop, First, Last, Shuffle, Menu để người dùng có thể click giao tiếp với chương trình.



Hình 3.1: Giao diện chính của chương trình

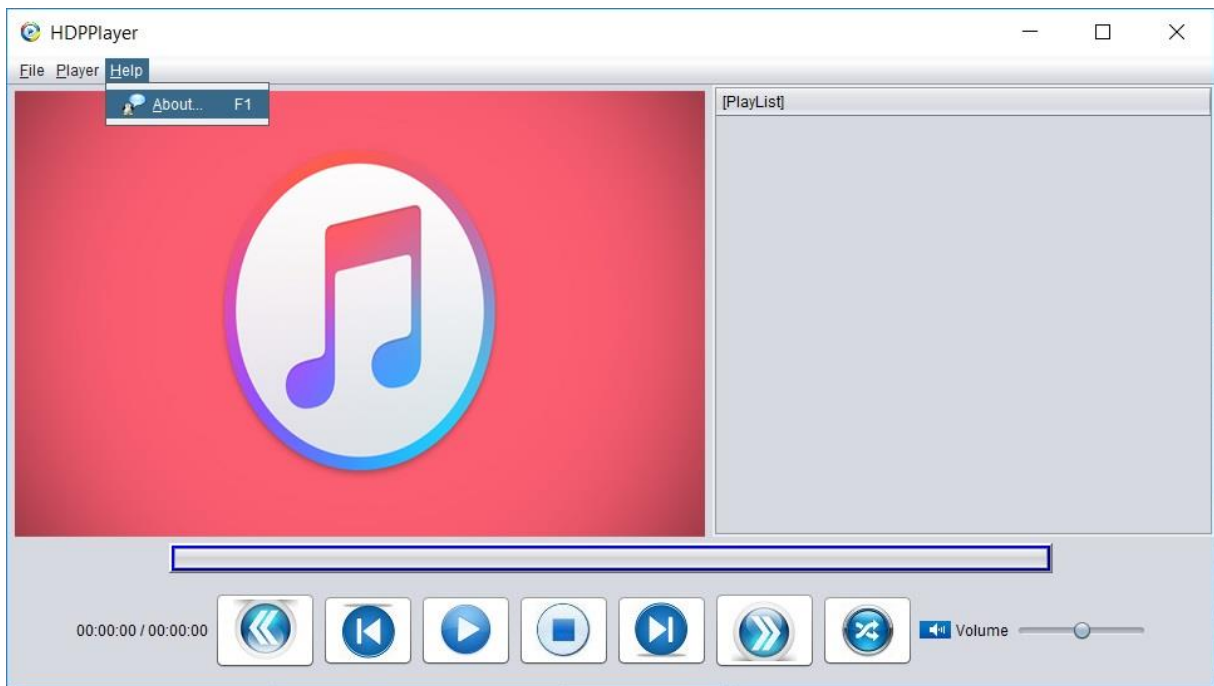
- Open File : Mở 1 file nhạc hoặc nhiều file nhạc do người dùng tự chọn.
- Open Playlist: mở file playlist đã được lưu trên máy tính trước đó.
- Save Playlist: Lưu playlist đã chọn thành định dạng .m3u để sau này có thể mở lại, không cần chọn lại bài hát.
- Exit : Thoát chương trình



Hình 3.2 : File Menu

-Help Menu

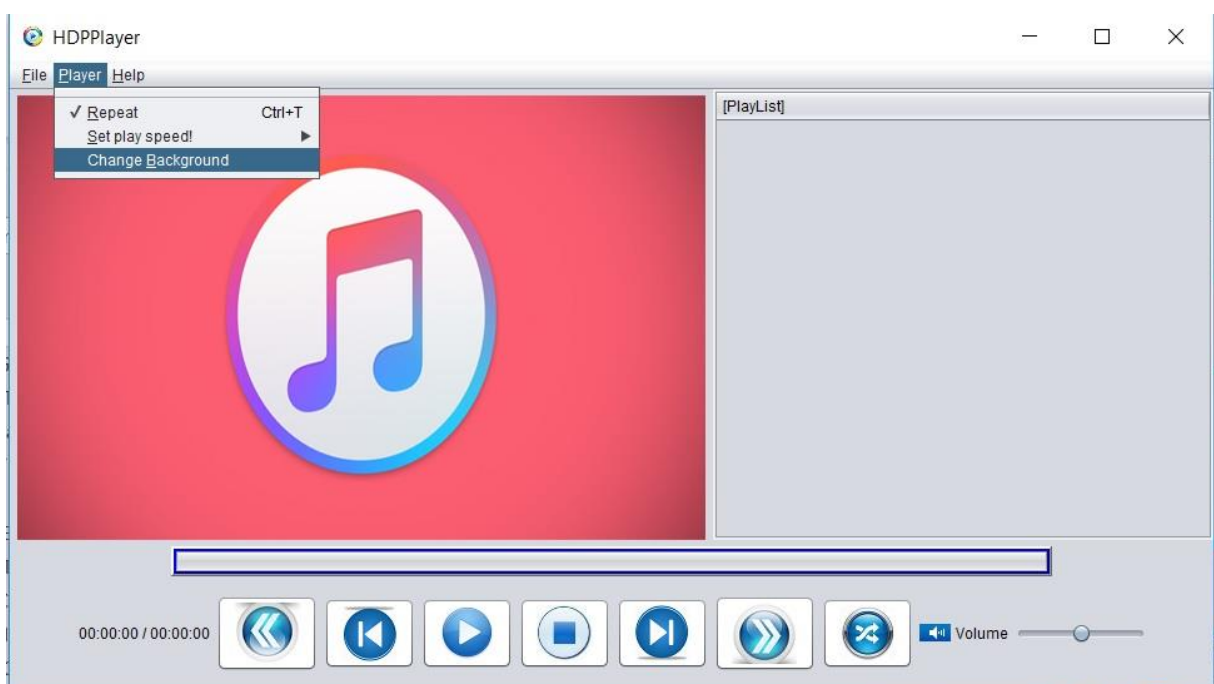
- About : Thông tin về người xây dựng phần mềm.



Hình 3.3 : Help Menu

-Player Menu










- Repeat : Khi người dùng chọn chế độ này thì chương trình sẽ tự động lặp lại playlist khi đã chạy hết.
- Set play speed : Thay đổi tốc độ chạy nhạc theo nhu cầu người dùng với 3 mức: slow, normal và fast.
- Change background : thay đổi hình nền giao diện chương trình.





Hình 3.4: Thay đổi background cho chương trình.

IV. Chức năng các button trên Tab Chơi Nhạc :

| | |
|---|--|
|  | Chuyển về bài đầu tiên trong danh sách |
|  | Quay lại bài trước bài đang chơi trong danh sách |
|  | Tiếp tục chơi nhạc |
|  | Tạm dừng nhạc |
|  | Dừng và tua lại từ đầu bài hát |
|  | Tới bài kế tiếp bài đang chơi trong danh sách |
|  | Chuyển về bài cuối cùng trong danh sách |
|  | Xáo trộn thứ tự chơi file media ban đầu |
|  | Tăng giảm âm lượng |

PHẦN 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

- Kết luận về ưu nhược điểm:

Do lần đầu tiên tìm hiểu và tạo chương trình nghe nhạc nên chương trình vẫn chưa được nhiều chức năng như những phần mềm chuyên nghiệp khác, mặt khác chương trình chỉ chạy nhạc có sẵn trong ổ đĩa máy tính mà không tìm kiếm chạy nhạc trên internet. Nhưng chương trình là một thành quả tìm hiểu và thiết kế trong một khoảng thời gian, có đầy đủ các chức năng cơ bản của một chương trình nghe nhạc.

- Hướng phát triển cho đề tài, cho sản phẩm và khả năng ứng dụng :

Do thời gian tìm hiểu và tạo chương trình không được nhiều nên em chỉ tạm dừng ở đây, nhưng em sẽ cố thêm các chức năng đã thiếu ở phần trên để có thể đáp ứng cho người dùng một chương trình nghe nhạc thật ổn định, giao diện thân thiện với người dùng.

TÀI LIỆU THAM KHẢO

- [1] Website <http://www.oracle.com/>.
- [2] Một số nguồn tài liệu khác trên internet.

PHỤ LỤC

HƯỚNG DẪN CÀI ĐẶT

Để chạy được chương trình máy tính cần được cài hệ điều hành Windows hoặc Linux.
Chỉ cần mở thư mục chứa file rồi chạy file HDPPlayer.jar.