Abgabedokumentation

Fabian Pfaff

July 30, 2019

Zielplattform

MPyC ist ein Terminal User Interface(TUI) für das Musikbackend MPD. Es wurde auf einem Linux basierten Betriebssystem entwickelt und getestet. MPD wird auch als Windows executable angeboten. Somit ist es denkbar, dass MPyC auch in einer Windowskonsole funktioniert.

verwendete Module

Für das Projekt wurden bis auf das Modul mpd nur Module verwendet die bei der Standardinstallation vorhanden sind.

Modul	Verwendung
mpd	erlaubt die Kommunikation mit MPD
curses	Für die Erstellung eines TUI
time	Sleeptimer im Mainloop um FPS zu beschränken

Erstellte Klassen

Die erstellten Klassen befinden sich alle in der Datei classes.py Für das Projekt wurden folgende Klassen entwickelt

Viewport

Die Viewport-Klasse erstellt mittels curses ein Windows. Eine wichtige Eigenschaft dieser Klasse ist, dass diese speichert welche Komponente sie beinhaltet. Die Klasse beinhaltet verschiedenste Funktionen wie render, on_resize um ein allgemeines Interface für die Inhalte zu bieten. Die Mainfunktion erstellt 3 viewports um das TUI in drei Teile zu unterteilen. So kann einfach der Inhalt des Hauptfensters ausgetauscht werden ohne die Mainfunktion mit den notwendigen Daten überfluten.

Topbar

Die Topbar dient der Statusanzeige. So wird hier der aktuelle Zustand von MPD angezeigt. Es gibt einen Statusindikator für Playing/Paused, die Anzeige des aktuellen Künstler mit Song angezeigt, sowie eine Progressbar des aktuellen Songs.

Commandline

Dies ist die Klasse welches den unteren Viewport befüllt. In dieser Klasse wird der Wiedergabezustand(Repeat, Single, Random, Consume) angezeigt. Weiterhin befindet sich hier die Anzeige wie viele Sekunden der Song schon fortgeschritten ist und die Songlänge. Wie der Name der Klasse vermuten lässt, dient die Klasse auch als Eingabe für Befehle(z.B. Suche). Dies wurde in der bisherigen Version nicht implementiert.

Playlist

Die Playlist zeigt die aktuelle Playlist an. In dieser Klasse kann die Paylist manipuliert werden. So kann man einzelne Songs aus der Liste verschieben bzw. Vertauschen, löschen und abspielen. Das Inteface gibt Informationen über den Künstler, Album, Track No, Titel und Songlänge an.

Library

Die Library zeigt die von MPD verwalteten Musikbibliothek an. Man kann hier durch die Bibliothek navigieren.

Help

Der Helpscreen dient der Anzeige der Tastenbelegung. Es ist auch stets das erste was im Hauptfenster angezeigt wird. In der Anzeige wurden Emojis für die Pfeiltasten verwendet, da die gängigsten Linuxdistributionen direkt Emoji-Fonts für die Konsole mitliefern.

Nichtimplementierung

Die Folgenden Elemente wurden nicht implementiert

Suchfunktion

Die Suchfunktion wurde nicht mehr implementiert, da diese mit dem bisherigen Design nicht tragbar ist. Bisher wird alles in einem einzelnem Thread abgearbeitet. Jedoch fordert die Suchfunktion das Warten und Anzeigen einer Nutzereingabe. Der zum Schluss vorhandende Zeitraum wurde dann genutzt um das Programm gegen Grenzfälle abzusichern.

Albumcover

Die Funktionalität ein Albumcover in einem Terminalemulator (also nicht tty) wurde schnell wieder verworfen, da das Pythonpaket "Überzug" nicht mit allen gängigen Terminalemulatorn funktioniert. Auch ist das verwenden der Software w3image nicht konsistent und sorgt für Problemen mit nicht konsistenten Rereshraten

FFT-Spektrum

Die Erstellung eines FFT-Spektrum des aktuellen Songs wurde nicht weiterentwickelt, da mir das Wissen um Audiospektren und deren praktischen Anzeige fehlt. Auch stellte sich heraus, dass auch hierfür Threading von Nöten ist um Asynchron die Rohdaten aus /tmp/mpp.fifo auszulesen und zu verarbeiten. Bisherige Versuche befinden sich unter testing/spectrum_test.py

Lyrics

Die Lyricssuche wurde nicht angefasst, da sie sehr weit unten in der Liste meiner persöhnlichen Wünsche stand.

mehre Elemente markieren

Das Verschieben/Löschen von mehreren wurde aus zeilichen Gründen nicht implementiert. Die Realisierung fände durch speichern in einer Liste statt. Die Funktionen für Verschieben/Löschen müssten dann angepasst werden ob die befüllt ist oder nicht. Auch benötigt dies eine neue Art der Markierung um dem Nutzer anzuzeigen, dass die Elemente ausgewählt sind.

Verbesserungen

Die wichtigste Verbesserung ist es den Quellcode so abzuändern um Multithreading zu erlauben. Dies erlaubt es die bisherigen nicht implementierten Funktionen zu realisieren. Für weiteres besseres Design ist es denkbar, dass die Klassen für das Hauptfenster von einer abstrakten Klasse erben. Dies würde es vereinfachern den Movement Code nicht neu zu erfinden.

Eine weitere Verbesserung ist die bessere Handhabung von zu langen Strings. So wäre eine Scrollingfunktionalität denkbar die eine Lauftschrift erstellt. Die Rechenlast aufgrund der Playlistklasse kann verringert werden indem nicht für jeden Frame eine neue Abfrage gemacht wird.

Da MPD als Musik Server entwickelt wurde, muss dieses nicht zwingend auf dem selben Rechner laufen wie der Client. Im bisherigen Fall laufen Client und Server auf der gleichen Maschine und die Adresseinstellungen in mpd.conf müssen mit den Daten in main.py übereinstimmen. So funktioniert MPyC nur in dieser Server/Client

Konfiguration. Deshalb ist die Erweiterung um einen Config-file-reader denkbar um auch das verbinden mit anderen MPD-Servern zu erlauben.

bekannte Fehler

Bei Resizing des Terminals in vertikaler Richtung bleibt das aktuell highgelightete Element womöglich nicht in Sicht.