2016

Requirements

MASTER AUDIO TECHNOLOGY FUNCTIONS

CVETKOVIĆ VANJA, DIMITRIJEVIĆ PREDRAG, NENADOVIĆ ĐURO

FAVULTY OF MATHEMATICS, BELGRADE | Studentski Trg 16, Belgrade

Contents

Introduction

In this chapter we are going to discuss about some general topics about app MATF.

Purpose

Its main purpose is to be simple audio manipulating tool. It needs to implement audio file viewer, player and various manipulations.

Intended audience

Master Audio Technology Functions is an app for amateurs, and beginners in audio manipulation area.

Additional information

App needs to be simple, reusable, flexible, and efficient.

Overall Description

Here we are going to give a slightly detailed description of the app.

Product perspective

In perspective app needs to implement various features and functionalities described below. It is the set of the most important functions to achieve goals described in intro section.

Product functions

Main function is to manipulate audio files. Implementation should include graphical representation of audio file, playing, pause, stop file, combine more than one audio file in one, adjust volume, have nice looking GUI, probably with themes, and colour styles, implement audio file standardisation in specific note (changing of frequency), virtual keyboard, audio effects and more. All of that can be found in Product Backlog.

User classes and characteristics

Every class represent specific element of GUI which has value for user. Classes should be main parts of app, every of them representing element which has essential value. GUI and implementation of functionalities shouldn't be separated for user to have easy way to report eventual problems and developers to easily respond to it.

Operating environment

Operating environment should be Windows OS (Windows 7 or above) and Linux, if possible. Minimum hardware requirements are not fixed but they should follow modern OS and modern app requirements.

User environment

Environment for user should be intuitive, easy to use and visually appealing. Even in situation of heavy usage it must be responsive, fast, with no lagging. User will need basic PC to run the app.

External Interface Requirements

In this chapter we are going to discuss about some external interfaces.

User interfaces

Interface should not be overbooked with controls and menus, but contain only main controls and options. As described, it should be simple, easy to use, intuitive. App should have audio file visualizer element, player controls elements, flow control elements and also audio manipulation elements.

Software interfaces

Interfaces between classes should be set of controls between them that they can use. There must be also set of global variables but that should be retained as few as possible. All of the methods declared in interfaces must be reusable, and have a concrete value to system.

System Features

In this section we are going to discuss about some of the main features. Three of them are presented here, but some other features must be developed in order to achieve these ones. See the Product Backlog for more details.

Graphical representation of audio file

Here we are going to describe requirement for graphical representation of audio file.

Description and priority

User should be able to see graphical representation of loaded audio file. This is high priority feature.

Action/result

Given a user has opened app, when user load the audio file, then user should see graphical representation of audio file.

Functional requirements

This should be fast, in order to show to user graphical representation in real time, as audio file is playing.

Reproducing audio file

Here we are going to describe requirement for reproducing and playing the audio file.

Description and priority

User should be able to play file, stop it, and pause it, as minimum requirement. This is a high priority feature.

Action/result

Given a user has loaded file, when user performs action to reproduce it, then user should reproduce the file.

Functional requirements

This feature should be implemented efficiently so it can be implemented for user to see the results in real time. Also all graphical elements should follow the implementation such as cursor for playing, etc.

Combining multiple audio files in one

Here we are going to take a look in feature which implements combining multiple audio files in one.

Description and priority
This feature should implement playing and combining multiple audio files in one play flow, so that all selected files can be reproduced at the same time in given order. Also exported and saved in a project file. This is a critical priority feature.

Action/result
Given user have loaded the files and arranged them, when he start reproduction of files, then files should be reproduced.

Functional requirements
Feature should be intuitive, in a way that user can easily arrange files in desired order. Implementing easy access tools could be helpful.

Sound effects features
Here we are going to discuss about sound effect feature.

Description and priority
Implement sound effects such as frequency changing per file, echo, fade out, fade in, distortion, etc. For details, see the Product Backlog.

Action/result
When user have loaded audio file, when user perform sound effect on audio file, then sound effect should be applied.

Functional requirements
Tis feature should be fast, but this time, quality is in front of speed. Focus is on implementation but then speed should be optimized.

Other Non-functional Requirements
In this chapter, we are going to discuss about some performance and quality issues and requirements, and about required documentation.

Performance requirements
Performance is important, but functionalities comes first. To achieve good performances, systematic approaches to problems are crucial, so implementation shoud reuse all mechanisms which desired framework offers so app should be fast and reliable. Minimum performance requirements are that it works in real time in main features. The rest should be optimized as best as it can.

Software quality attributes
Accent is on quality, but in a special way: reliability. App needs to be reliable, not to crash, or loose data. This is the main requirement.

Project documentation
Project should be documented on a usage level. But the main focus is for app to be intuitive and easy to use, so documentation is just added value.