ACKNOWLEDGMENT

We the summer interns team of Multimedia Transcoder are over whelmed in all humbleness and gratefulness to acknowledge our depth to all those who have helped us to put our ideas and assigned work, well above the level of simplicity and into something concrete.

We all whole heartedly thank Prof. Deepak B. Phatak for selecting us as a part of his valuable project, constantly motivating us for doing better and showing complete confidence in our work.

We are indebted to our Project Manager Mr. Parag Tiwari for his constant help and support in light of this project. We are very thankful to our mentor Mr. Vivek Nadar. With the help of his valuable suggestions, guidance and encouragement, we all were able to complete our tasks properly and with satisfaction. Also in the process, we learnt a lot of other technical and non-technical things from him and we consider ourselves to be very fortunate to have such mentor.

We would like to thank Mr. Bikas Chhatri and Mr. Dilip Sable for making our stay here as summer interns comfortable and for all their administrative help. Without them, a lot of time would have gone in doing extra work rather than the main project work.

Finally we also like to thank all other colleagues working in different projects under Prof. Deepak B. Phatak for helping us at small problems as well as critical junctures.

Team Members

Tejwinder Singh (National Institute of Technology, Jamshedpur)

Komal Choudhary (University College of Engineering, Kota)

Shanal Jauhari (National Institute of Technology, Trichy)

Mentors

Mr. Parag Tiwari – Project Manager at IIT Bombay

paragt@it.iitb.ac.in

Mr. Vivek Nadar – Jr Software Engineer at IIT Bombay

vivek0511@gmail.com

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included. We have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Multimedia Transcoder Team

Date: 05-07-2012

Abstract

Nowadays video is being produced and consumed in more component representation formats, more device types and over a variety of networks than ever. Transcoding is a process of translating or converting one coded signal representation to another. However most of the time transcoding becomes computationally intensive process and complex task if done through command line/terminal. Due to this, one might need a graphical user interface which is easy to use by anyone .This "Multimedia Transcoder" project aims at providing a user-friendly GUI to easily and quickly convert videos/audios from one format to another, specify different codecs, resolutions, aspect ratios, frame rates, bit rates.

It facilitates automated conversion of input data formats (lecture videos, lecture slides) to data formats supported by popular web browsers (like Firefox, Chrome, Opera, Safari, IE and Ubisurfer browser for Aakash tablets) and OS platforms (like Windows 7/Ubuntu Linux/MacOX/Android 2.3) .

Contents

Chapter 1

Introduction

Rich media resources transmitted over wired and wireless networks are becoming more and more important in our lives. Users of devices, such as computers, PDAs, mobile phones, require accessing the Internet and networks from anywhere and at any time regardless of resources, network conditions, and capabilities of client devices. Intelligent media conversions, such as scalability, transcoding and modality changes, are the important means for multimedia adaptation and Universal Multimedia Access (UMA).

Transcoding is defined, in general, as the conversion of one compressed signal to another. Transcoding is the direct digital-to-digital data conversion of one encoding to another, such as for movie data files or audio files. This is usually done in cases where a target device (or workflow) does not support the format or has limited storage capacity that mandates a reduced file size, or to convert incompatible or obsolete data to a better-supported or modern format.

Transcoding can be performed just while files are being searched, as well as for presentation. The research work concerning transcoding is mainly focused on efficiency, complexity, latency, processing power, error resilience, bit rate change, resolution change, frame rate change, picture quality and so forth. To solve the problems, the coding methods, algorithms and international coding standards used for design and implementation of transcoding architectures are critical.

1.1 Purpose

Choosing the perfect video converter software is like choosing the perfect car, it all depends on what you need the program to do and how far you want to go. In some cases, all you need is the basic sub-compact without luxury upgrades to get you from point A to point B. Others need a powerful, feature-filled, mega-program that will convert just about every file format under the sun.

The main purpose of most video converter programs is to convert media already on your computer to a more compatible format on other computers or digital devices. There are three main types of uses for video converter software; to convert home video, online video and DVD or VCD video.

Emailing and sharing home videos with relatives near and far is a great way to keep in touch. However, the camcorder or video editing software you use is usually partial to certain file formats and you don’t always know the software your relatives will use to play the video back. Frequently, a well-meaning birthday hello or introduction to the new baby turns into a lengthy technical phone call. With video converter software, you can simply change your Windows Media Video to .AVI or another friendly format.

Video converter software is particularly useful when you own an iPod, but have a computer that uses Windows. A conundrum many are running into is they are uploading and editing their home videos on their home computer, but they can’t transfer and play them on their iPod. Although iTunes works with Windows, most of the other Mac based software doesn’t; including video files.

If your main purpose is to convert home videos directly to your personal computer, via USB, or to your portable media player look for simple, familiar formats like .WMV, .AVI, DivX, Xvid and MPEG-2. If you are using a Mac or iPod look for .MOV and .MP4.

Transcoding can be performed just while files are being searched, as well as for presentation. For example, Cineon and DPX files have been widely used as a common format for digital cinema, but the data size of a two-hour movie is about 8 terabytes (TB). That large size can increase the cost and difficulty of handling movie files. However, transcoding into a JPEG2000 lossless format has better compression performance than other lossless coding technologies, and in many cases, JPEG2000 can compress images to half-size.

Transcoding is commonly a loss process, introducing generation loss; however, transcoding can be lossless if the input is compressed without any loss and the output is either compressed without any loss or uncompressed. The process of loss-to-loss transcoding introduces varying degrees of generation loss. In other cases, the transcoding of lossy to lossless or uncompressed is technically a lossless conversion because no information is lost, however the process is irreversible and is more suitably known as destructive.

1.2 Scope

The software helps the user to take existing video and change the format, bitrate and/or resolution in order to view it on another video device. This software provides a convenient and easy-to-use graphical user interface to easily and quickly convert audio/video files to different formats supported by different platforms (e.g. the newly developed Aakash Tablet), resolution, bit-rate, frame-rate. The software also helps the user to extract audio from a given video file.

One can also split a video multiple times and assign a unique name to it depending upon the theme. The application is very easy and convenient to use.

1.3 Definitions, Abbreviations

FFmpeg – Fast Forward Motion Picture Experts Group.

Transcoding- It is the direct digital-to-digital data conversion of one encoding to another, such as for movie data files or audio files.

Codec - Codec is a software (computer program) that is able to do the task of both encoding and decoding.

GUI – Graphical User Interface

AAC – Advanced Audio Coding

FFPlay – fast Forward Play

FLV – Flash Video

OGG – Operation Good Guys

OGV – Ogg Video

WAV – Windows Wave

WMV – Windows Media Video

AVI – Audio Video Interleave

MPEG - Motion Picture Experts Group

1.4 Motivation

The motivation behind developing Multimedia Transcoder was mainly to make video converting easy, using the FFmpeg application. The motivation for the project was to provide a video converter, which can convert a video in a given format to a desired format, video to audio and to split a given video into the desired length and parts.

1.5 Transcoding vs Conversion

Transcoding: Changes the signal encoding from one format to another without otherwise altering the signal. The encoding format is changed, but resolution, aspect ratio, bit-rate, frame-rate, audio channels, etc. remain the same.

Conversion: Converts encoding and Signal format but not necessarily resolution or aspect ratio. Converters with Scaling can adjust the input signal to fit the resolution and aspect ratio of the screen.

- The main thing is to know what you have for an input signal and want for an output signal before you begin. If you need a video input or output, then that part is easy - use a converter with composite input or output to match what you need. You just need to know what you want on the other end.Secondly, you have to know the input and output resolutions and signal types before choosing an appropriate Transcoder, Converter, or Converter/Scaler.

- Video Conversion always has an effect on your signal quality. The less conversions you do, the better off you will be.

- Quality of the Electronics can have a very large effect on the ultimate signal quality. Since, conversion is NOT, a simple thing.

1.6 Drawbacks of Transcoding

The key drawback of transcoding in lossy formats is decreased quality. Compression artifacts are cumulative, so transcoding causes a progressive loss of quality with each successive generation, known as digital generation loss. For this reason, transcoding is generally discouraged unless unavoidable.

It is better to retain a copy in a lossless format (such as TTA, FLAC or WavPack for sound), and then encode directly from the lossless source file to the lossy formats required. For digital audio editing, one is advised to save audio in a raw or uncompressed format and edit (a copy of) that version, only converting to lossy formats for distribution.

Although the loss is unavoidable, it can be minimized by choosing the appropriate resolution and bit-rate.

Chapter 2

Objectives

To convert and split VIDEO (can be more than one) from one particular format to any of the following VIDEO formats:

- Mp4
- Avi
- Webm
- Ogv
- Mpeg
- Wmv
- Flv
- Mp4(H.264)

with the change of some properties like

1. Resolution

2. Aspect Ratio

3. Frame Rate

4. Video Bit Rate

To convert and split VIDEO (can be more than one) from one particular format to any of the following AUDIO formats

- Mp3
- Wav

with the change of some properties like

1. Audio Sample Rate

2. Audio Bit Rate

3. Audio Channels

using JavaFX and FFmpeg.

Chapter 3

Technologies Used

FFmpeg

FFmpeg is a free software project that produces libraries and programs for handling multimedia data. The most notable parts of FFmpeg are **libavcodec**, an audio/video codec library used by several other projects, **libavformat**, an audio/video container mux and demux library, and the **FFmpeg** command line program for transcoding multimedia files. FFmpeg is published under the GNU Lesser General Public License 2.1+ or GNU General Public License 2+ (depending on which options are enabled).

ffmpeg is a command-line tool to convert one video file format to another. It can also grab and encode in real-time from a TV card.

It is a wonderful library for creating video applications or even general purpose utilities. ffmpeg takes care of all the hard work of video processing by doing all the decoding, encoding, muxing and demuxing for you. This can make media applications much simpler to write. It's simple, written in C, fast, and can decode almost any codec you'll find in use today, as well as encode several other formats.

There is a sample program that comes with ffmpeg called ffplay. It is a simple C program that implements a complete video player using ffmpeg.

JavaFX 2.1

JavaFX is a software platform for creating and delivering rich Internet applications (RIAs) that can run across a wide variety of connected devices. The current release (JavaFX 2.1, April 2012) enables building applications for desktop, browser and mobile phones. TV set-top boxes, gaming consoles, Blu-ray players and other platforms are planned.

Before version 2.0 of JavaFX, app developers used a statically typed, declarative language called JavaFX Script to build JavaFX applications. However, Java code could be integrated into JavaFX programs. JavaFX Script was compiled to Java bytecode, so JavaFX applications could run on any desktop and browser that runs the Java Runtime Environment (JRE) and on top of mobile phones running Java ME.

The new JavaFX 2.0 and later is implemented as a native Java library and is therefore written in native Java code. JavaFX Script has been scrapped by Oracle but development is being continued in the Visage project.

On desktop, the current release supports Windows XP, Windows Vista and Mac OS X operating systems. Beginning with JavaFX 1.2, Oracle has released beta versions for Linux and OpenSolaris. On mobile, JavaFX is capable of running on multiple mobile operating systems, including Symbian OS, Windows Mobile, and proprietary real-time operating systems.

JavaFX 2.1 includes the following main features:

- H.264 and AAC support
- First official version for Mac OS X (desktop only)
- LCD text
- UI enhancements including combo box controls, charts (stacked chart), and menu bars
- Webview component now allows JavaScript to make calls to Java methods

JDK and JRE versions from 1.7.0_04 and onwards include JavaFX bundled to be installed with them. When Java 8 is available, JavaFX will become part of the JRE/JDK.

Chapter 4

4.1 JavaFX Application

The JavaFX platform is the evolution of the Java client platform designed to enable application developers to easily create and deploy rich internet applications (RIAs) that behave consistently across multiple platforms. Built on Java technology, the JavaFX platform provides a rich set of graphics and media API with high-performance hardware-accelerated graphics and media engines that simplify development of data-driven enterprise client applications.

The main focus areas for the JavaFX 2 release include the following features, many of which are also described in the JavaFX Architecture and Framework document:

- **Java APIs for JavaFX** that provide all the familiar language features (such as generics, annotations, and multithreading) that Java developers are accustomed to using. The APIs are designed to be friendly to alternative JVM languages, such as JRuby and Scala. Because the JavaFX capabilities are available through Java APIs, you can continue to use your favorite Java developer tools (such as IDEs, code refactoring, debuggers, and profilers) to develop JavaFX applications.

- **A new graphics engine** to handle modern graphics processing units (GPUs). The basis of this new engine is a hardware accelerated graphics pipeline, called Prism, that is coupled with a new windowing toolkit, called Glass. This graphics engine provides the foundation for current and future advancements for making rich graphics simple, smooth, and fast.

- **FXML, a new declarative markup language** that is XML-based and is used for defining the user interface in a JavaFX application. It is not a compiled language and, hence, does not require you to recompile the code every time you make a change to the layout.

- **A new media engine** that supports playback of the web multimedia content. It provides a stable, low latency media framework that is based on the GStreamer multimedia framework.

- **A web component** that gives the capability of embedding web pages within a JavaFX application using the WebKit HTML rendering technology. Hardware accelerated rendering is made available using Prism.

- **A refreshed browser plug-In** for JavaFX 2 that allows the loading of JavaFX applets based on Prism.

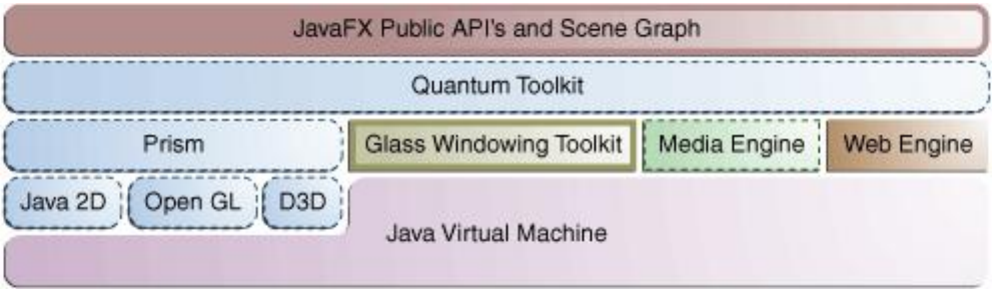
- **A wide variety of built-in UI controls**, which include Charts, Tables, Menus, and Panes. Additionally, an API is provided to allow third parties to contribute UI controls that the user community can use.

- **Sample applications** that showcase the different features of the JavaFX 2 technology, along with a large number of code samples and snippets.

- **An updated doclet** used with the Javadoc tool to generate JavaFX API documentation in HTML format. Detailed information on how to use this updated doclet can be found in Using a Doclet with JavaFX.

4.2 Architecture – JavaFX (Front End)

The sections following the diagram describe each component and how the parts interconnect. Below the JavaFX public APIs lies the engine that runs your JavaFX code. It is composed of subcomponents that include the new JavaFX high performance graphics engine, called Prism; the new small and efficient windowing system, called Glass; a media engine, and a web engine. Although these components are not exposed publicly, their descriptions can help you to better understand what runs a JavaFX application.



The JavaFX platform combines the best capabilities of the Java platform with comprehensive, immersive media functionality into an intuitive and comprehensive one-stop development environment. These new Java APIs for JavaFX features:

- Allow the use of powerful Java features, such as generics, annotations, and multithreading.
- Make it easier for Web developers to use JavaFX from other popular dynamic languages, such as JRuby, Groovy, and JavaScript.
- Allow Java developers to use other system languages, such as Groovy, for writing large or complex JavaFX applications.
- Allow the use of binding similar to that in JavaFX Script language. This includes support for high performance lazy binding, binding expressions, bound sequence expressions, and partial bind reevaluation. Alternative languages (like Groovy) can use this binding library to introduce binding syntax similar to that of JavaFX Script.
- Extend the Java collections library to include observable lists and maps, which allow applications to wire user interfaces to data models, observe changes in those data models, and update the corresponding UI control accordingly.

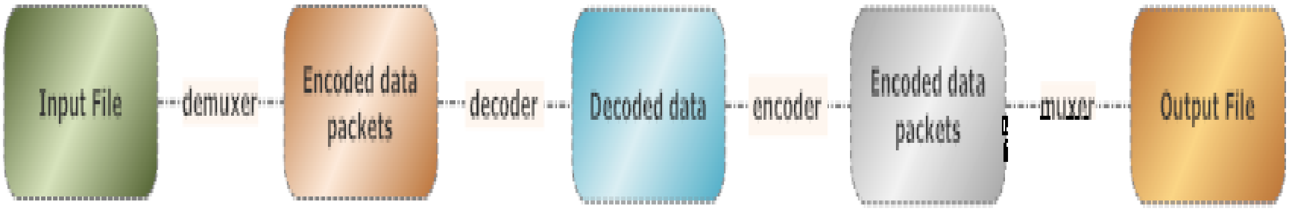
Chapter 5

5.1 FFmpeg (*The Backend Software used*)

FFmpeg is a complete, cross-platform solution to record, convert and stream audio and video. It includes libavcodec - the leading audio/video codec library.

FFmpeg reads from an arbitrary number of input "files" (which can be regular files, pipes, network streams, grabbing devices, etc.), specified by the -i option, and writes to an arbitrary number of output "files", which are specified by a plain output filename. Anything found on the command line which cannot be interpreted as an option is considered to be an output filename.

Each input or output file can, in principle, contain any number of streams of different types (video/audio/subtitle/attachment/data). Allowed number and/or types of streams can be limited by the container format. Selecting, which streams from which inputs go into output, is done either automatically or with the –map.



FFmpeg provides various tools:

- FFmpeg is a command line tool to convert multimedia files between formats.
- FFplay is a simple media player based on SDL and the FFmpeg libraries.
- FFprobe is a is a simple multimedia stream analyzer.

FFmpeg provides various developers libraries:

- libavutil is a library containing functions for simplifying programming, including random number generators, data structures, mathematics routines, core multimedia utilities, and much more.
- libavcodec is a library containing decoders and encoders for audio/video codecs.
- libav format is a library containing demuxers and muxers for multimedia container formats.
- libavdevice is a library containing input and output devices for grabbing from and rendering to many common multimedia input/output software frameworks, including Video4Linux, Video4Linux2, VfW, and ALSA.
- libavfilter is a library containing media filters.
- libswscale is a library performing highly optimized image scaling and colour space/pixel format conversion operations.
- libswresample is a library performing highly optimized audio resampling, rematrixing and sample format conversion operations.

5.2 Installation of FFmpeg

One can download the latest builds of FFmpeg from http://ffmpeg.zeranoe.com/builds/ and install the suitable setup. Then setup system path by editing the "user's path variable".

5.3 FFmpeg options used in “FX Converter”

- *‘-i filename (input)’*

  input file name

- ‘-target *type* (*output*)’

Specify target file type (vcd, svcd, dvd, dv, dv50). *type* may be prefixed with pal-, ntsc- or film- to use the corresponding standard. All the format options (bitrate, codecs, buffer sizes) are then set automatically.

Split options

- ‘-t *duration* (*output*)’

  Stop writing the output after its duration reaches *duration*. *duration* may be a number in seconds, or in hh:mm:ss[.xxx] form.

- ‘-ss *position* (*input/output*)’

  When used as an input option (before -i), seeks in this input file to *position*. When used as an output option (before an output filename), decodes but discards input until the timestamps reach *position*. This is slower, but more accurate. *position* may be either in seconds or in hh:mm:ss[.xxx] form.

Video options

- ‘-vframes *number* (*output*)’

  Set the number of video frames to record. This is an alias for -frames:v.

- ‘-r[:*stream_specifier*] *fps* (*input/output,per-stream*)’

  Set frame rate (Hz value, fraction or abbreviation). As an input option, ignore any timestamps stored in the file and instead generate timestamps assuming constant frame rate *fps*. As an output option, duplicate or drop input frames to achieve constant output frame rate *fps* (note that this actually causes the fps filter to be inserted to the end of the corresponding filtergraph).

- ‘-s[:*stream_specifier*] *size* (*input/output,per-stream*)’

  Set frame size. As an input option, this is a shortcut for the ‘video_size’ private option, recognized by some demuxers for which the frame size is either not stored in the file or is configurable – e.g. raw video or video grabbers. As an output option, this inserts the scale video filter to the *end* of the corresponding filtergraph. Please use the scale filter directly to insert it at the beginning or some other place. The format is ‘wxh’ (default - same as source).

- ‘-aspect[:*stream_specifier*] *aspect* (*output,per-stream*)’

  Set the video display aspect ratio specified by *aspect*. *aspect* can be a floating point number string, or a string of the form *num*:*den*, where *num* and *den* are the numerator and denominator of the aspect ratio. For example "4:3", "16:9", "1.3333", and "1.7777" are valid argument values.

- ‘-vcodec *codec* (*output*)’

  Set the video codec. This is an alias for -codec:v.

Audio options

- -aframes *number* (*output*)’

  Set the number of audio frames to record. This is an alias for -frames:a.

- ‘-ar[:*stream_specifier*] *freq* (*input/output,per-stream*)’

  Set the audio sampling frequency. For output streams it is set by default to the frequency of the corresponding input stream. For input streams this option only makes sense for audio grabbing devices and raw demuxers and is mapped to the corresponding demuxer options.

- ‘-ac[:*stream_specifier*] *channels* (*input/output,per-stream*)’

  Set the number of audio channels. For output streams it is set by default to the number of input audio channels. For input streams this option only makes sense for audio grabbing devices and raw demuxers and is mapped to the corresponding demuxer options.

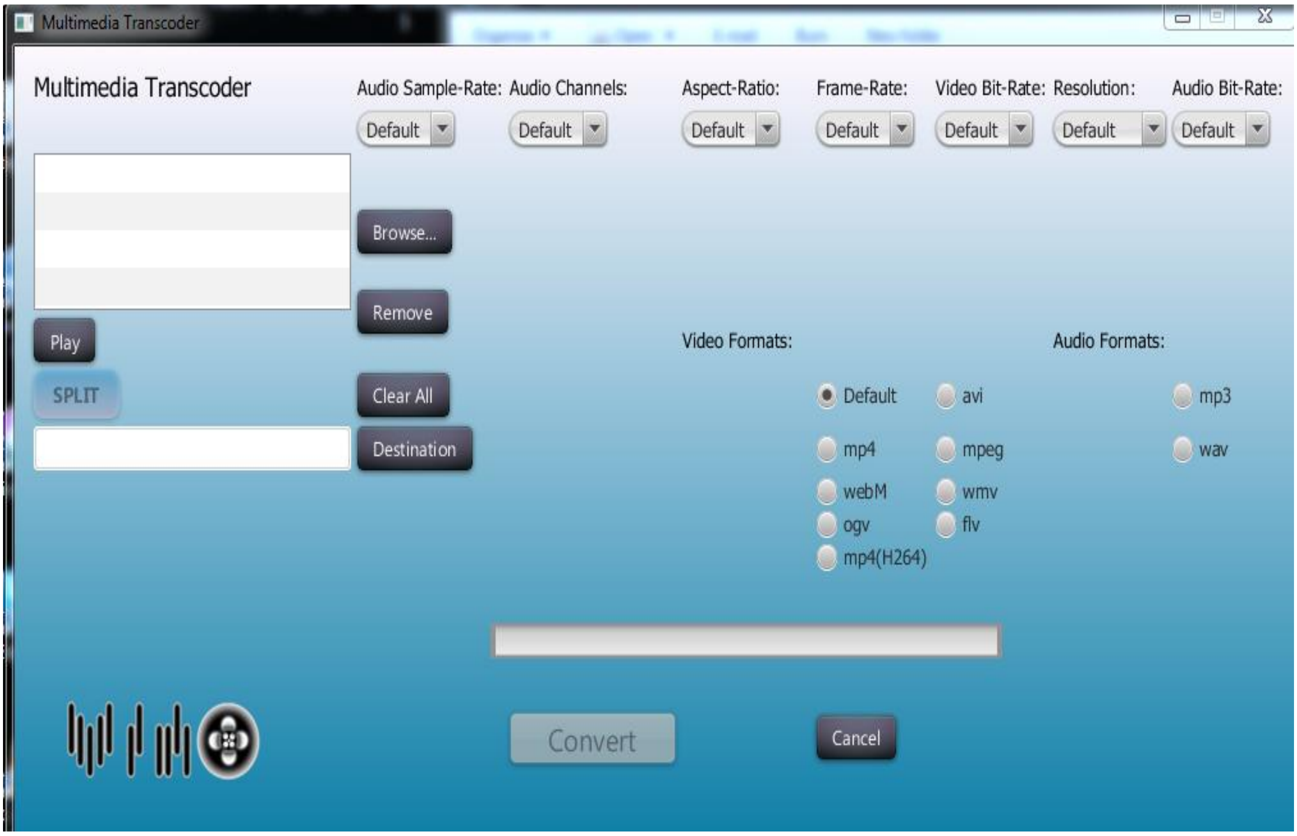
- ‘-acodec *codec* (*input/output*)’

  Set the audio codec. This is an alias for -codec:a.

Chapter 6

Design and Implementation of GUI

Main GUI



Audio Sample-Rate: Default

Audio Channels: Default

Aspect-Ratio: Default

Frame-Rate: Default

Video Bit-Rate: Default

Resolution: Default

Audio Bit-Rate: Default

Video Formats:

● Default ○ avi

○ mp4 ○ mpeg

○ webM ○ wmv

○ ogv ○ flv

○ mp4(H264)

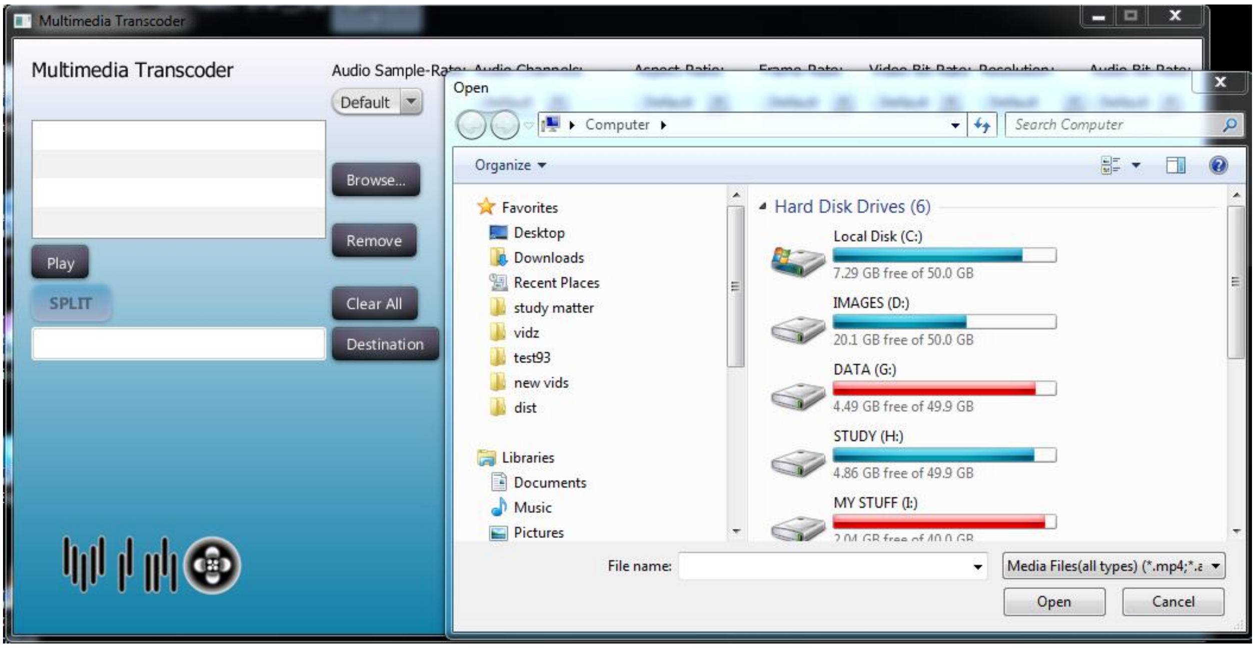
Audio Formats:

○ mp3

○ wav

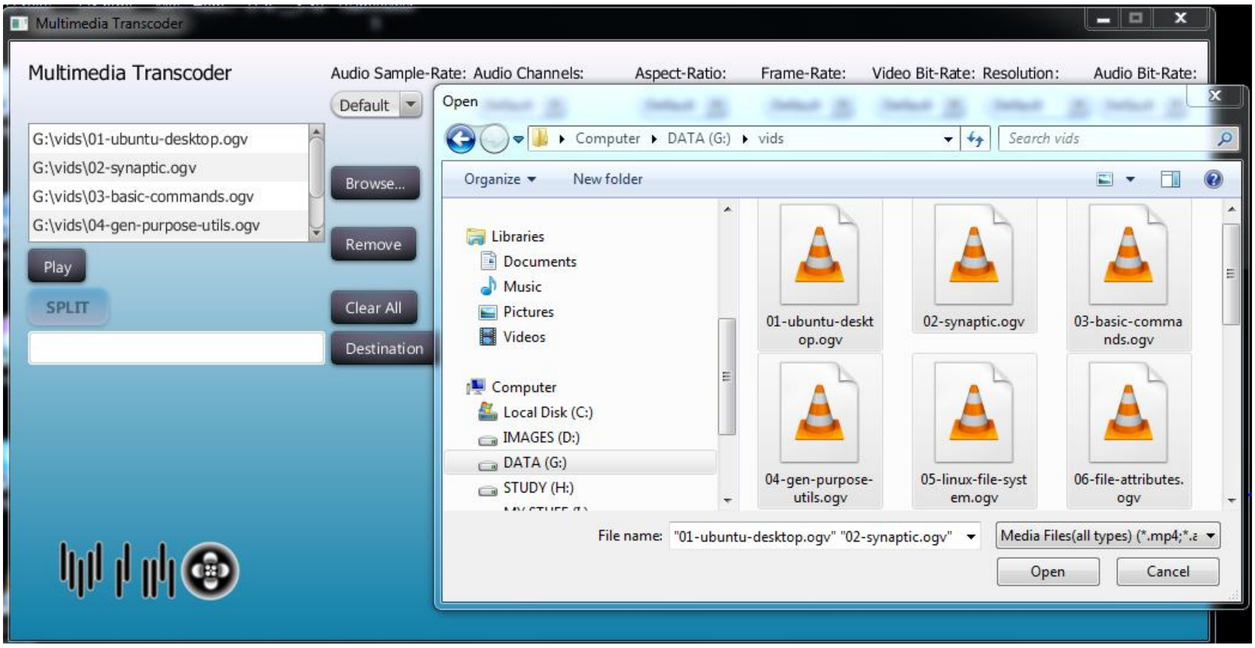
CONVERT PANEL

Browse Button: It is used to browse the files for the input List.



Multi Selection: One can browse multiple files to the Input List.

Drag and Drop multiple files: One can also browse multiple files to the Input List through drag and drop.

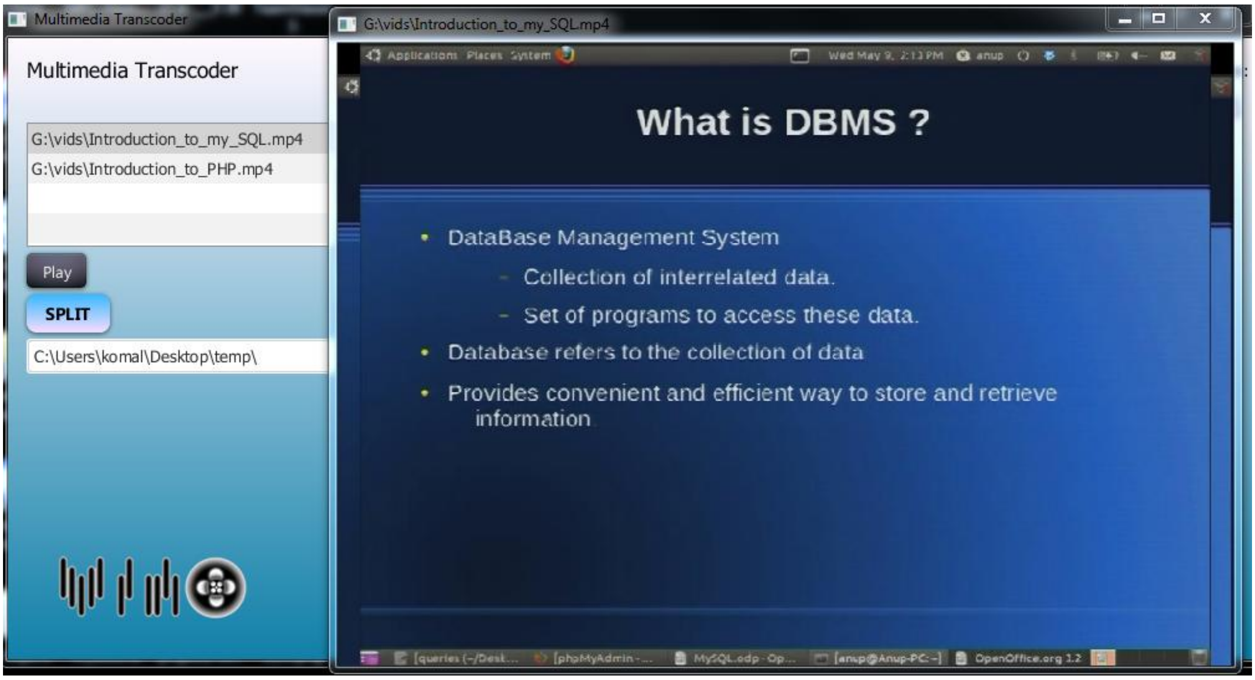


G:\vids\01-ubuntu-desktop.ogv
G:\vids\02-synaptic.ogv
G:\vids\03-basic-commands.ogv
G:\vids\04-gen-purpose-utils.ogv

File name: "01-ubuntu-desktop.ogv" "02-synaptic.ogv"

Media Files(all types) (*.mp4;*.a

Play Button: It is used to play the selected input file from List.



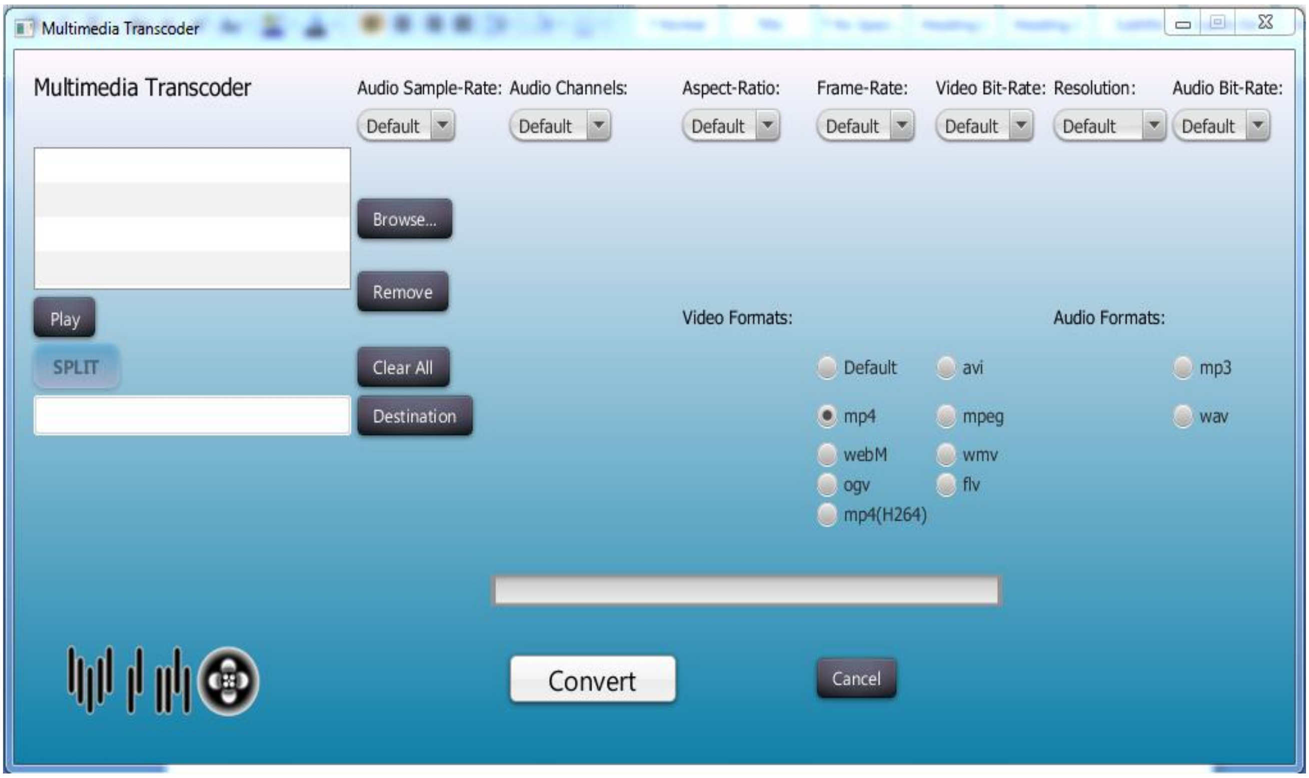
G:\vids\Introduction_to_my_SQL.mp4
G:\vids\Introduction_to_PHP.mp4

C:\Users\komal\Desktop\temp\

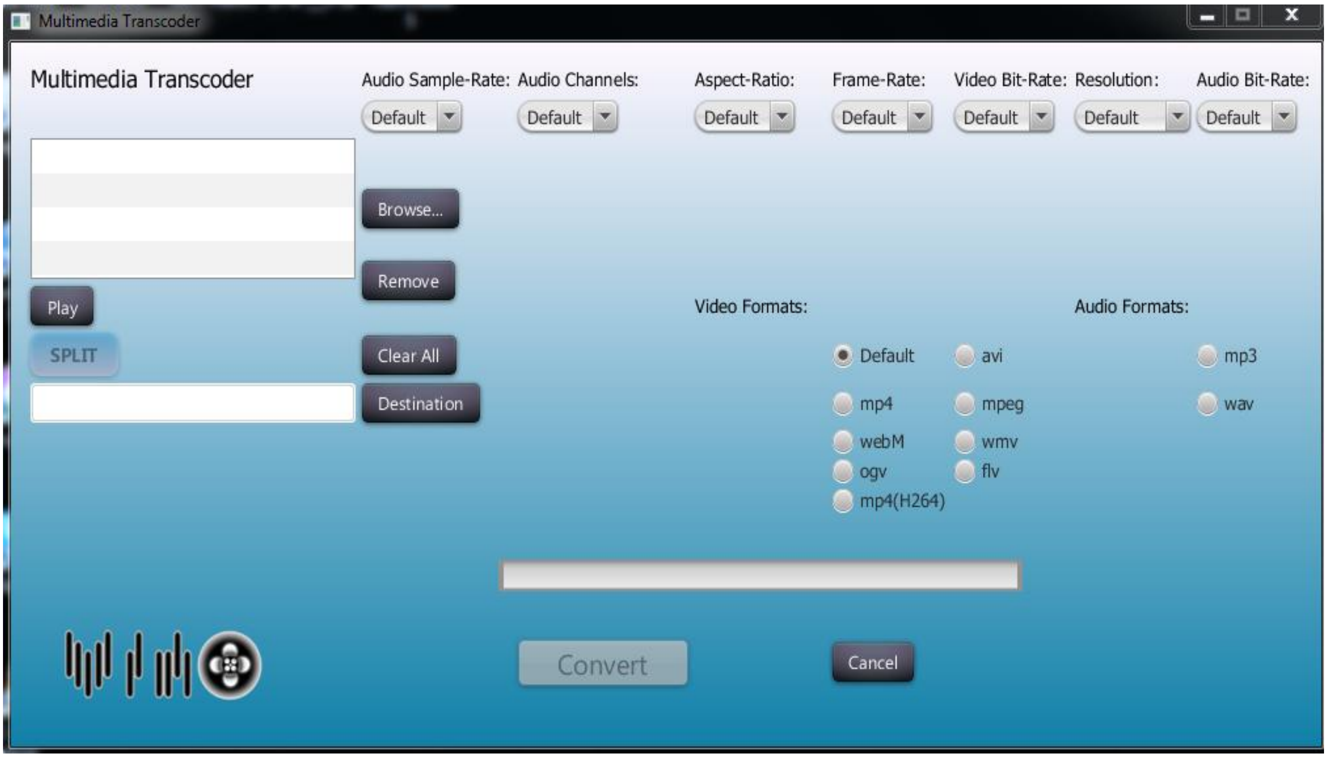
What is DBMS ?

- DataBase Management System
  - Collection of interrelated data.
  - Set of programs to access these data.
- Database refers to the collection of data
- Provides convenient and efficient way to store and retrieve information

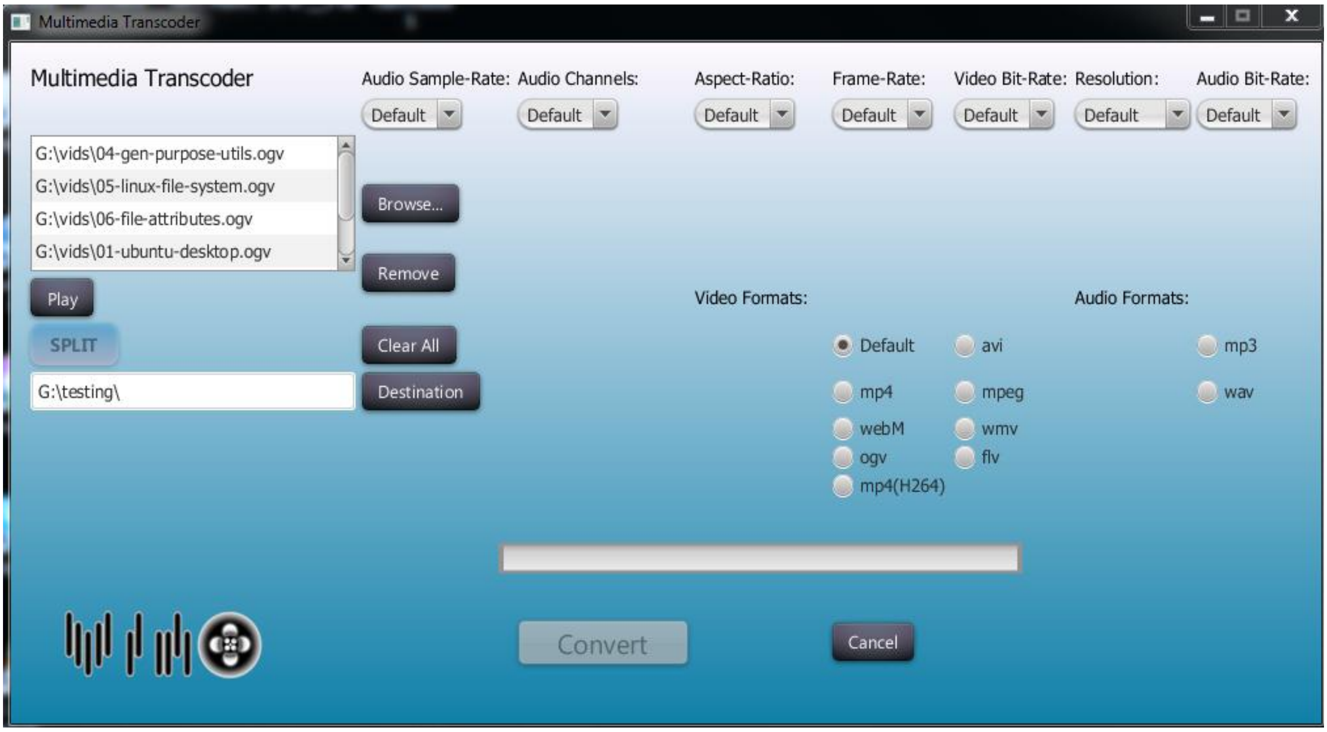
Remove Button: It is used to remove the selected file from Input/Output List.



Clear All Button: It is used to clear the entire Input List.



Destination Button: The default Output folder is the current working directory. The user can choose any destination folder by browsing, using this button.



G:\vids\04-gen-purpose-utils.ogv
G:\vids\05-linux-file-system.ogv
G:\vids\06-file-attributes.ogv
G:\vids\01-ubuntu-desktop.ogv

G:\testing\

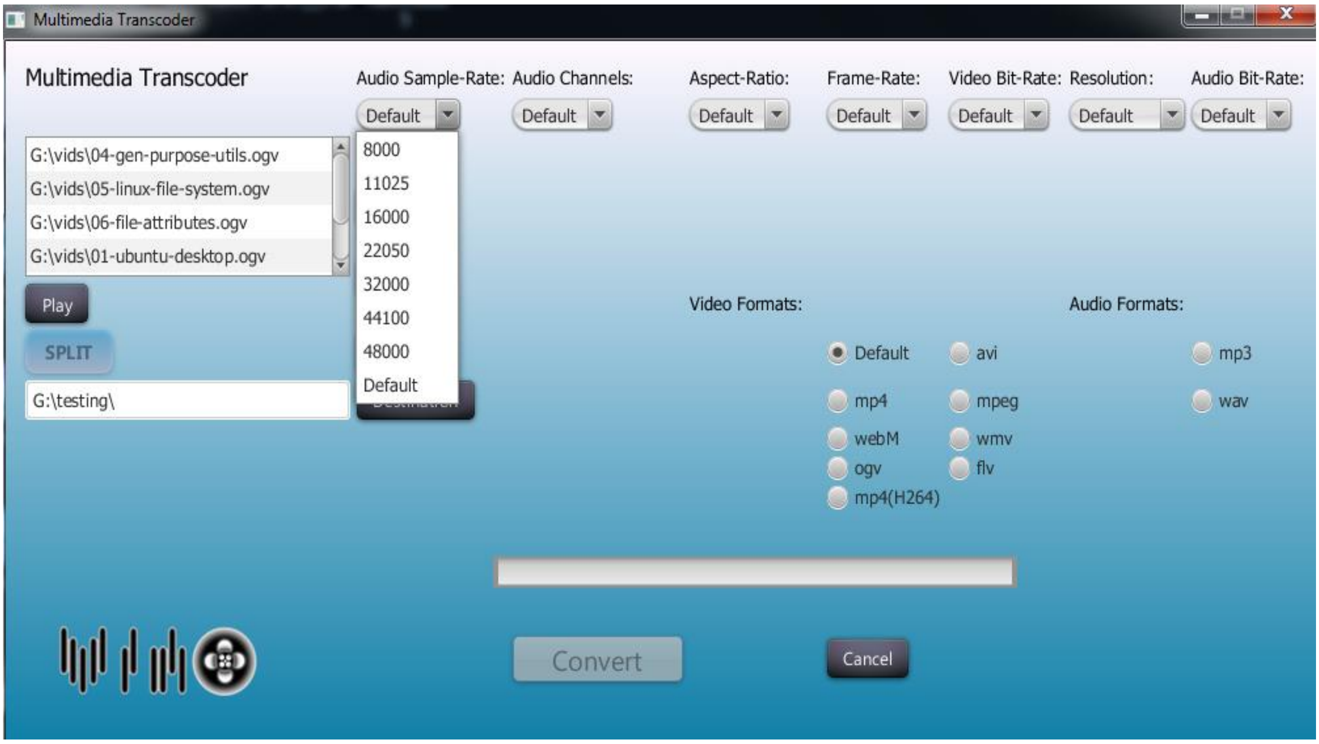
Audio Sample-Rate: Default
Audio Channels: Default
Aspect-Ratio: Default
Frame-Rate: Default
Video Bit-Rate: Default
Resolution: Default
Audio Bit-Rate: Default

Video Formats: ● Default ○ avi ○ mp4 ○ mpeg ○ webM ○ wmv ○ ogv ○ flv ○ mp4(H264)

Audio Formats: ○ mp3 ○ wav

Audio-Video Properties: Additional options can be selected by using settings button.

- Video Bit rate: It is used to select different bitrates from listed values.
- Video Frame Rate: It is used to select different frame rates from listed values.
- Resolution: It is used to set the different resolutions for videos.
- Aspect Ratio: It is used to set the aspect ratio for video.
- Audio Bit rate: It is used to select different bitrates from listed values.
- Sample Rate: It is used to select different frame rates from listed values.
- Audio Channels: It is used to set number of audio channels to either 1 or 2. It is 2 by default.


Multimedia Transcoder

Multimedia Transcoder

Audio Sample-Rate: Default
Audio Channels: Default
Aspect-Ratio: Default
Frame-Rate: Default
Video Bit-Rate: Default
Resolution: Default
Audio Bit-Rate: Default

G:\vids\04-gen-purpose-utils.ogv
G:\vids\05-linux-file-system.ogv
G:\vids\06-file-attributes.ogv
G:\vids\01-ubuntu-desktop.ogv

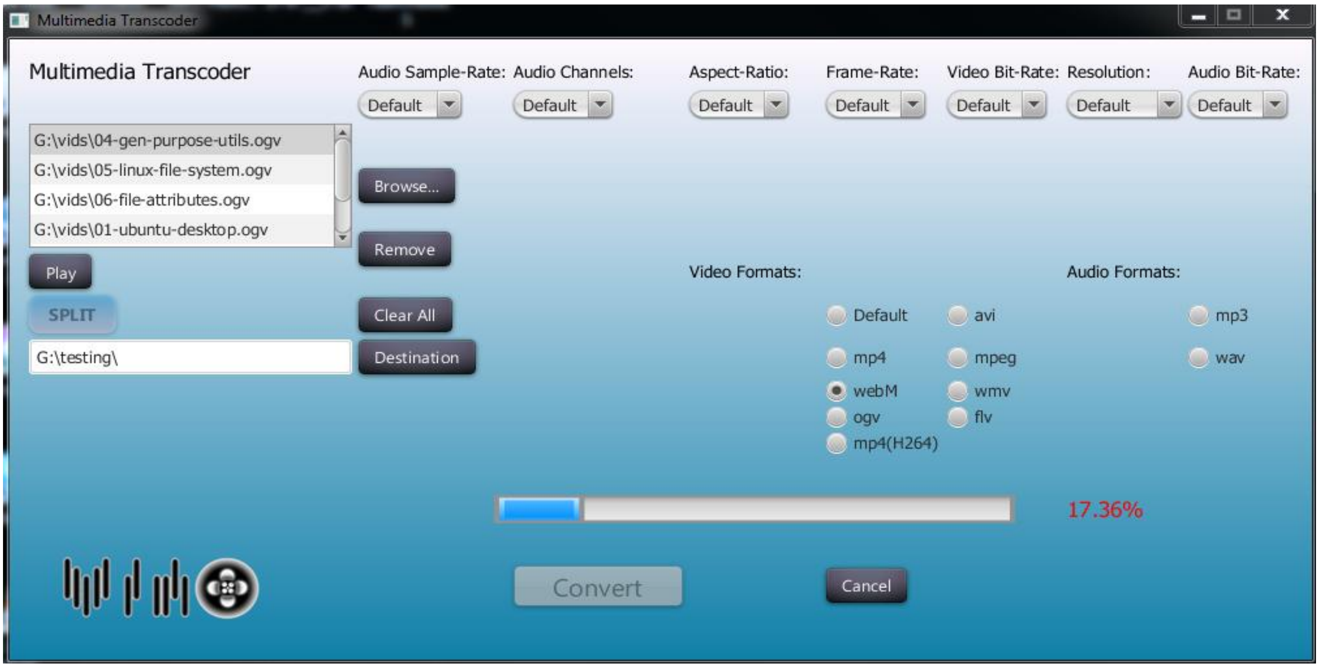
G:\testing\

Video Formats: ● Default ○ avi ○ mp4 ○ mpeg ○ webM ○ wmv ○ ogv ○ flv ○ mp4(H264)

Audio Formats: ○ mp3 ○ wav

Format Radio Buttons: They are used to convert the Input files to a particular format.

Convert Button: It is used to convert the input files to different formats and codecs.


Multimedia Transcoder

Multimedia Transcoder

Audio Sample-Rate: Default
Audio Channels: Default
Aspect-Ratio: Default
Frame-Rate: Default
Video Bit-Rate: Default
Resolution: Default
Audio Bit-Rate: Default

G:\vids\04-gen-purpose-utils.ogv
G:\vids\05-linux-file-system.ogv
G:\vids\06-file-attributes.ogv
G:\vids\01-ubuntu-desktop.ogv

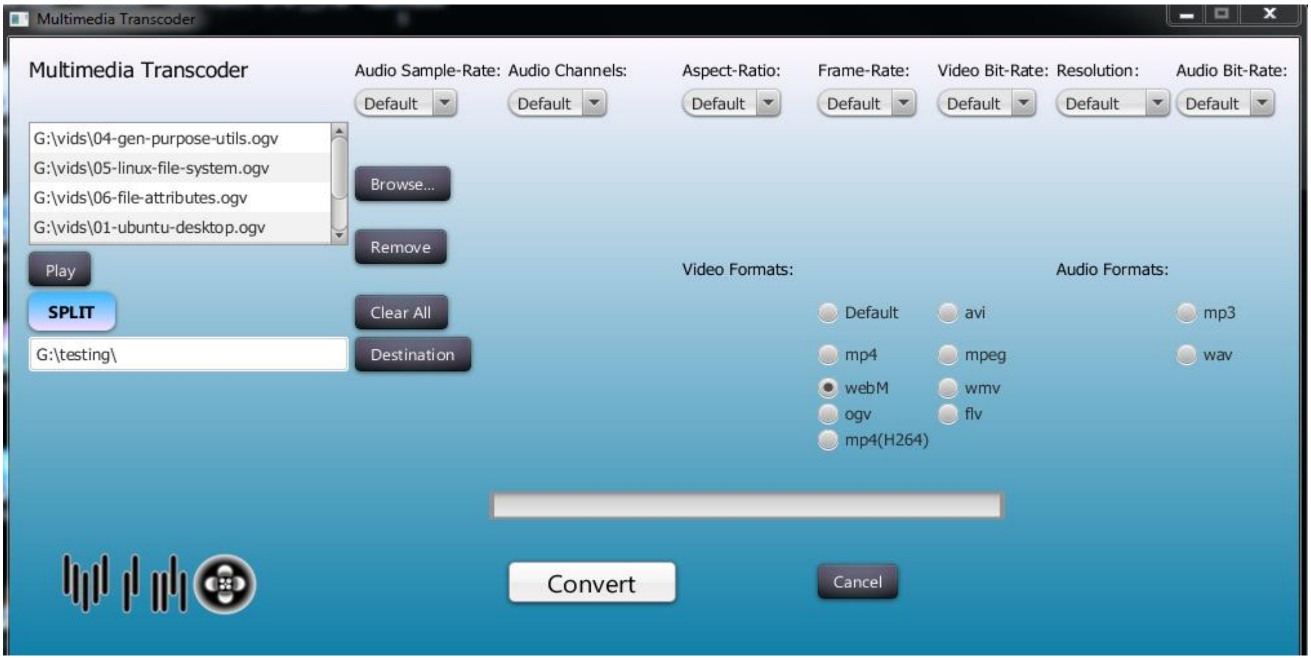
G:\testing\

Video Formats: ○ Default ○ avi ○ mp4 ○ mpeg ● webM ○ wmv ○ ogv ○ flv ○ mp4(H264)

Audio Formats: ○ mp3 ○ wav

17.36%

<u>Cancel Button</u>: The conversion of files can be cancelled using cancel button. It will delete the file which is presently converting.



Multimedia Transcoder

Multimedia Transcoder

Audio Sample-Rate: Default

Audio Channels: Default

Aspect-Ratio: Default

Frame-Rate: Default

Video Bit-Rate: Default

Resolution: Default

Audio Bit-Rate: Default

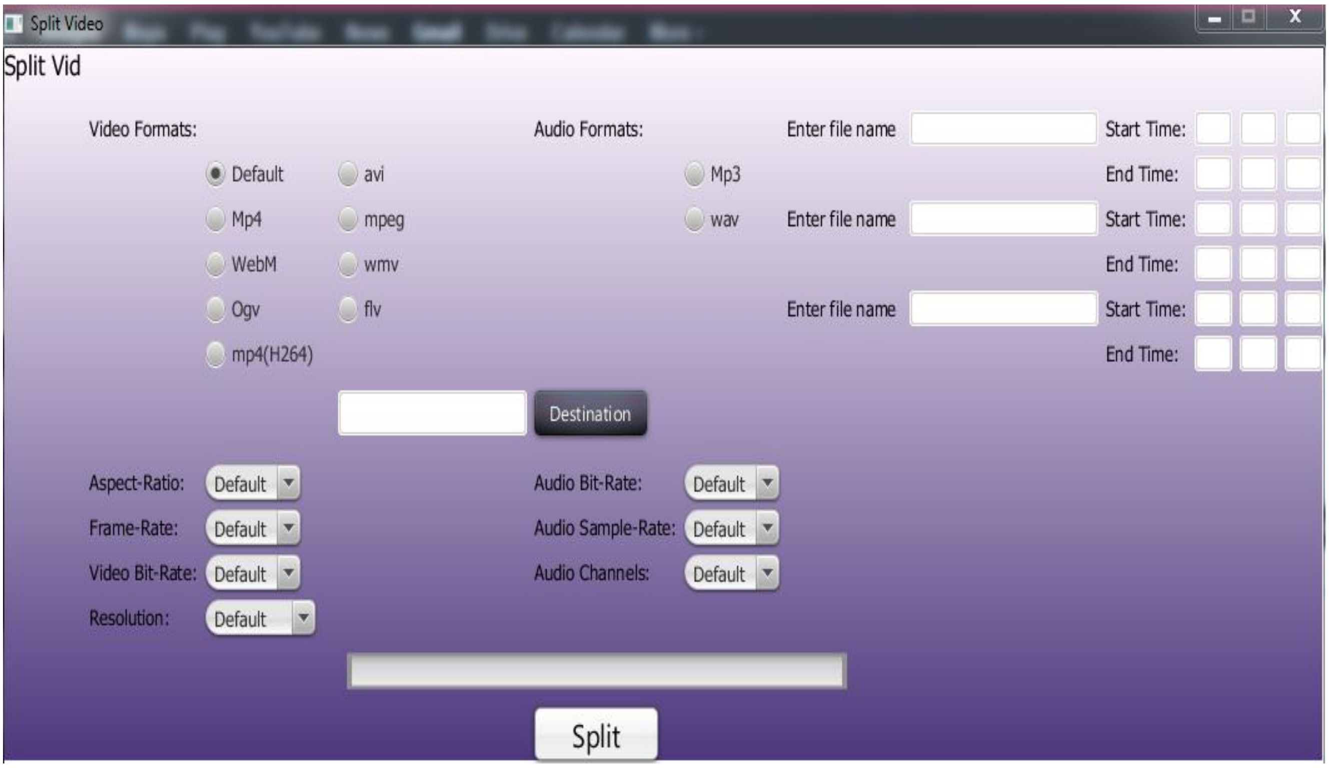
G:\vids\04-gen-purpose-utils.ogv
G:\vids\05-linux-file-system.ogv
G:\vids\06-file-attributes.ogv
G:\vids\01-ubuntu-desktop.ogv

G:\testing\

Video Formats: ○ Default ○ avi ○ mp4 ○ mpeg ● webM ○ wmv ○ ogv ○ flv ○ mp4(H264)

Audio Formats: ○ mp3 ○ wav

SPLIT PANEL



Split Video

Split Vid

Video Formats: ● Default ○ avi ○ Mp4 ○ mpeg ○ WebM ○ wmv ○ Ogv ○ flv ○ mp4(H264)

Audio Formats: ○ Mp3 ○ wav

Enter file name Start Time: End Time:

Enter file name Start Time: End Time:

Enter file name Start Time: End Time:

Aspect-Ratio: Default

Frame-Rate: Default

Video Bit-Rate: Default

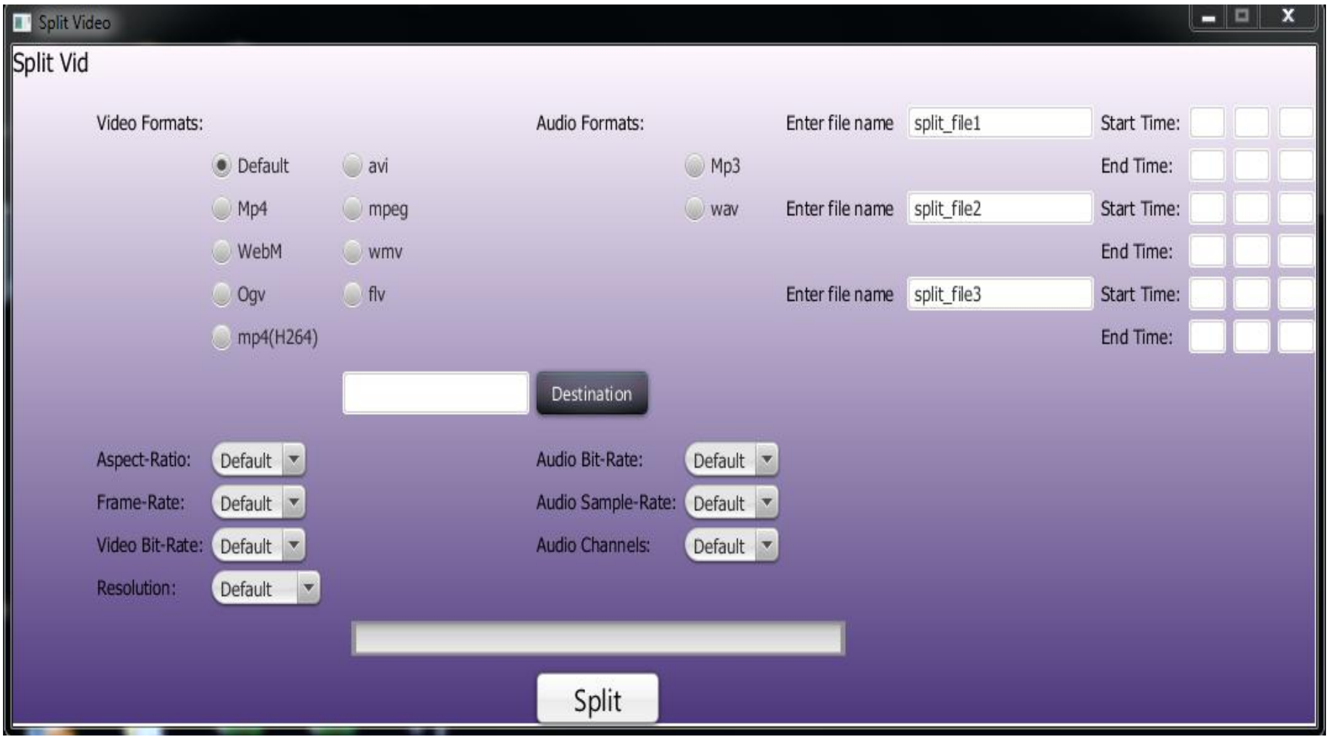
Resolution: Default

Audio Bit-Rate: Default

Audio Sample-Rate: Default

Audio Channels: Default

File Names: The different parts of same converted file can be given different names.



Split Video

Split Vid

Video Formats: ● Default ○ Mp4 ○ WebM ○ Ogv ○ mp4(H264) ○ avi ○ mpeg ○ wmv ○ flv

Audio Formats: ○ Mp3 ○ wav

Enter file name split_file1 Start Time: End Time:

Enter file name split_file2 Start Time: End Time:

Enter file name split_file3 Start Time: End Time:

Destination

Aspect-Ratio: Default

Frame-Rate: Default

Video Bit-Rate: Default

Resolution: Default

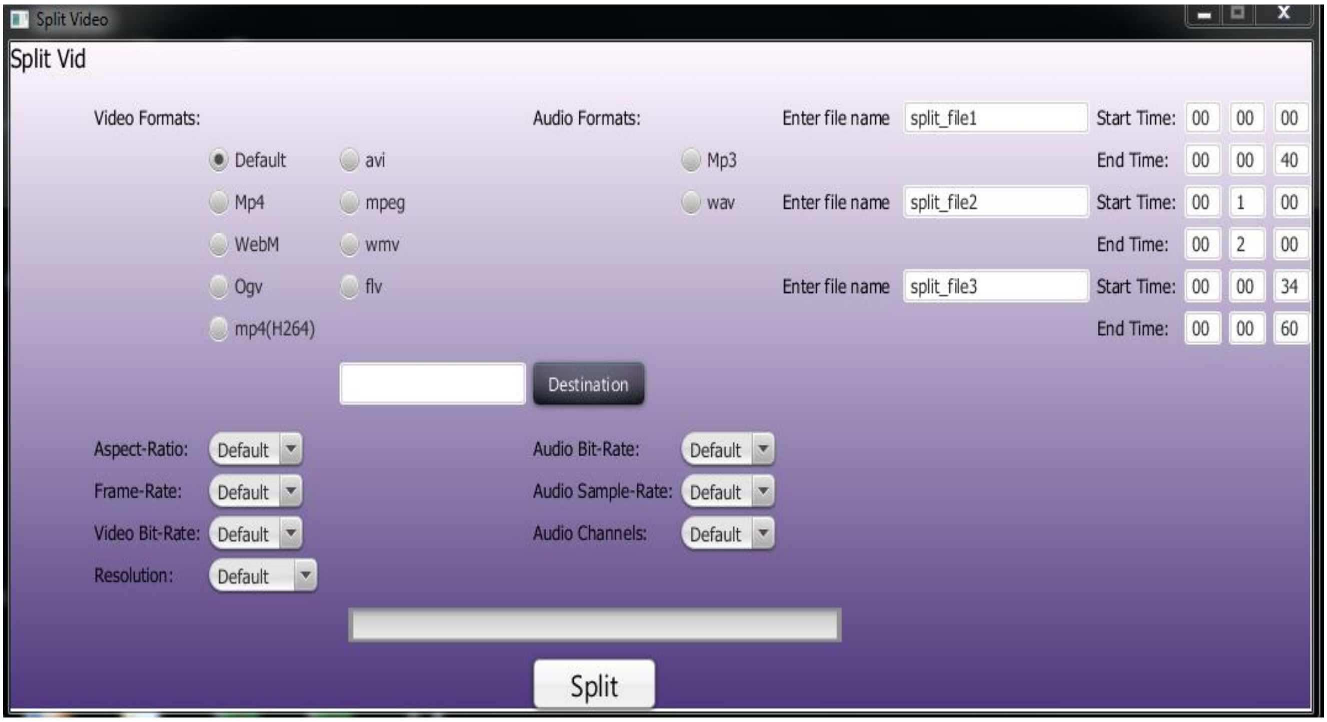
Audio Bit-Rate: Default

Audio Sample-Rate: Default

Audio Channels: Default

Split

Start and End Times: The video can be split in different parts, for that the text fields will get the start and end time in the format of hh:mm:ss.



Split Video

Split Vid

Video Formats: ● Default ○ Mp4 ○ WebM ○ Ogv ○ mp4(H264) ○ avi ○ mpeg ○ wmv ○ flv

Audio Formats: ○ Mp3 ○ wav

Enter file name split_file1 Start Time: 00 00 00 End Time: 00 00 40

Enter file name split_file2 Start Time: 00 1 00 End Time: 00 2 00

Enter file name split_file3 Start Time: 00 00 34 End Time: 00 00 60

Destination

Aspect-Ratio: Default

Frame-Rate: Default

Video Bit-Rate: Default

Resolution: Default

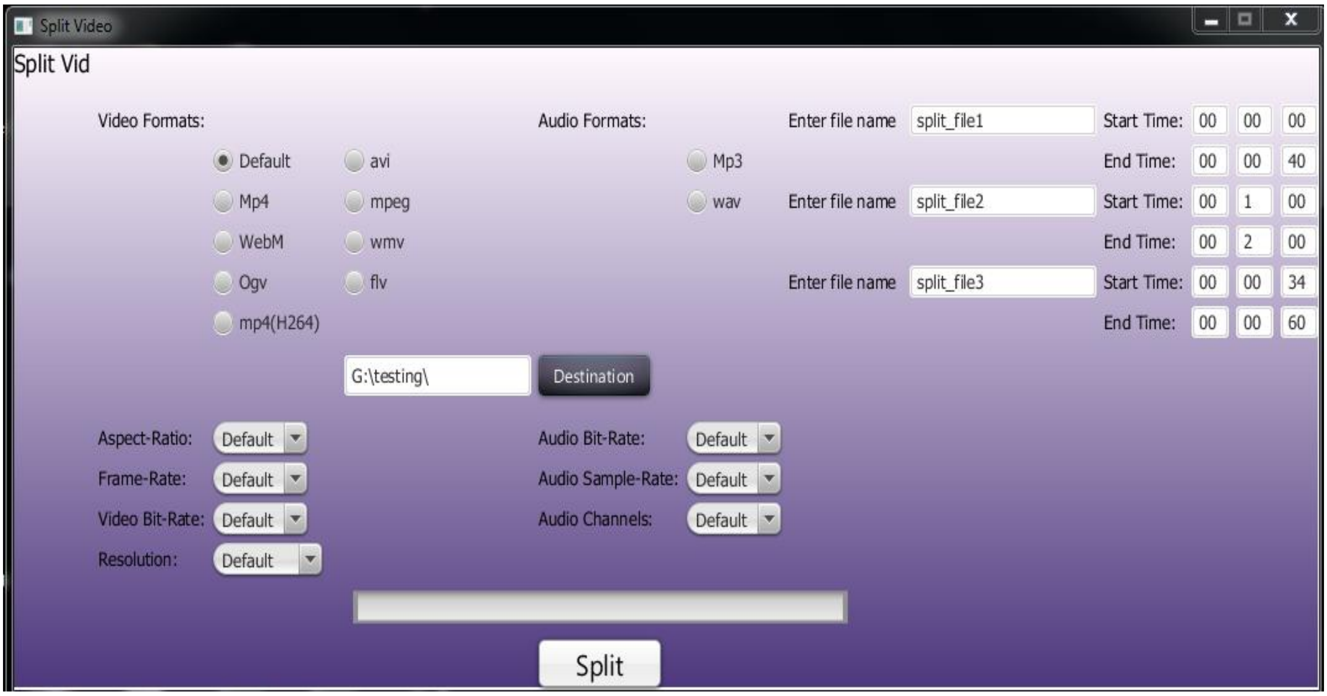
Audio Bit-Rate: Default

Audio Sample-Rate: Default

Audio Channels: Default

Split

Destination Button: The default output folder is the current working directory. The user can choose any destination folder by browsing, using this button.



Split Video

Split Vid

Video Formats: ● Default ○ avi ○ Mp4 ○ mpeg ○ WebM ○ wmv ○ Ogv ○ flv ○ mp4(H264)

Audio Formats: ○ Mp3 ○ wav

Enter file name split_file1 Start Time: 00 00 00 End Time: 00 00 40

Enter file name split_file2 Start Time: 00 1 00 End Time: 00 2 00

Enter file name split_file3 Start Time: 00 00 34 End Time: 00 00 60

G:\testing\

Aspect-Ratio: Default

Frame-Rate: Default

Video Bit-Rate: Default

Resolution: Default

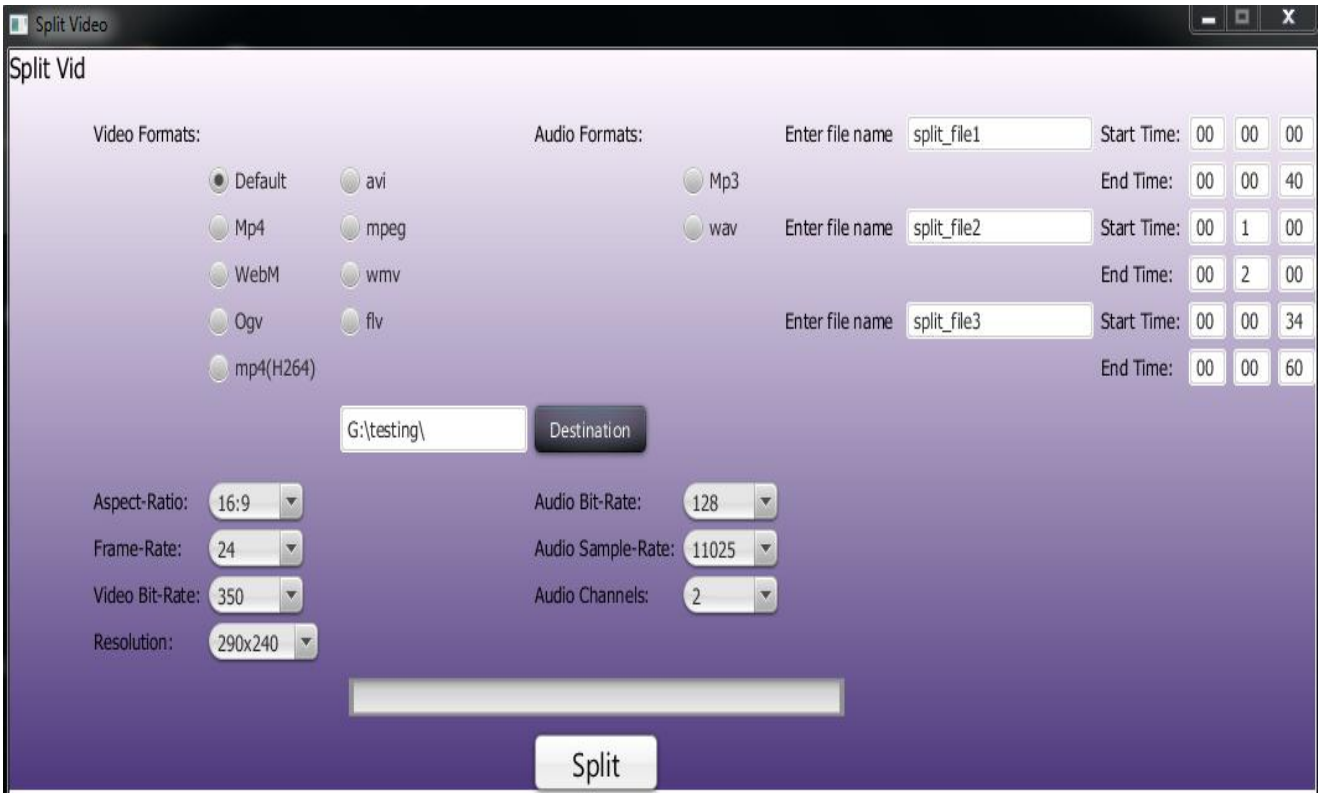
Audio Bit-Rate: Default

Audio Sample-Rate: Default

Audio Channels: Default

Audio-Video Properties

- Audio Bit rate: Select different bitrates from listed values.
- Sample Rate: Select different frame rates from listed values.
- Audio Channels: Set the number of audio channels to either one or two. It is 2 by default.
- Video Bit rate: It is used to select different bitrates from listed values.
- Video Frame Rate: It is used to select different frame rates from listed values.
- Resolution: It is used to set the different resolutions for videos.
- Aspect Ratio: It is used to set the aspect ratio for video.



○ Default ● (selected) — avi, Mp4, mpeg, WebM, wmv, Ogv, flv, mp4(H264); Audio: Mp3, wav

Enter file name: split_file1 — Start Time: 00 00 00 — End Time: 00 00 40

Enter file name: split_file2 — Start Time: 00 1 00 — End Time: 00 2 00

Enter file name: split_file3 — Start Time: 00 00 34 — End Time: 00 00 60

G:\testing\

Aspect-Ratio: 16:9

Frame-Rate: 24

Video Bit-Rate: 350

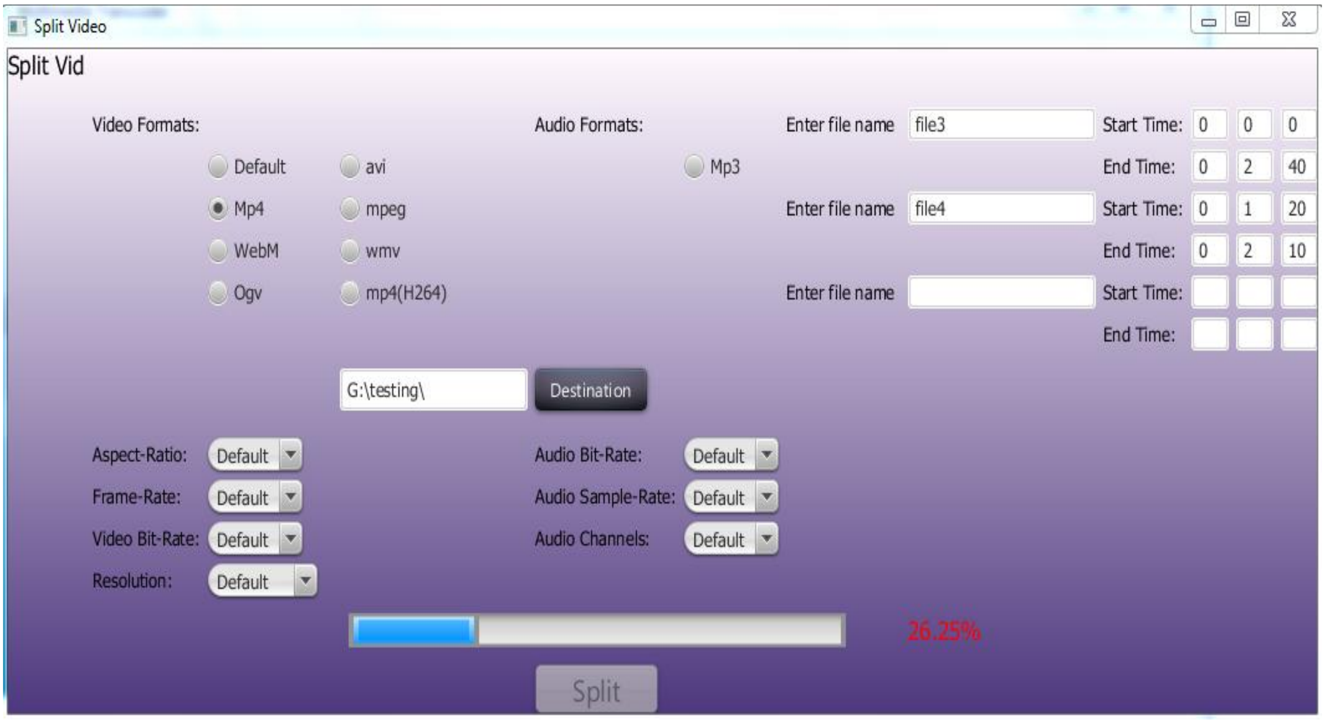
Resolution: 290x240

Audio Bit-Rate: 128

Audio Sample-Rate: 11025

Audio Channels: 2

<u>Split Button</u>: It will convert and split the selected video in one or more parts (max 3) .



Default, avi, ● Mp4 (selected), mpeg, WebM, wmv, Ogv, mp4(H264); Audio: Mp3

Enter file name: file3 — Start Time: 0 0 0 — End Time: 0 2 40

Enter file name: file4 — Start Time: 0 1 20 — End Time: 0 2 10

Enter file name: — Start Time: — End Time:

G:\testing\

Aspect-Ratio: Default

Frame-Rate: Default

Video Bit-Rate: Default

Resolution: Default

Audio Bit-Rate: Default

Audio Sample-Rate: Default

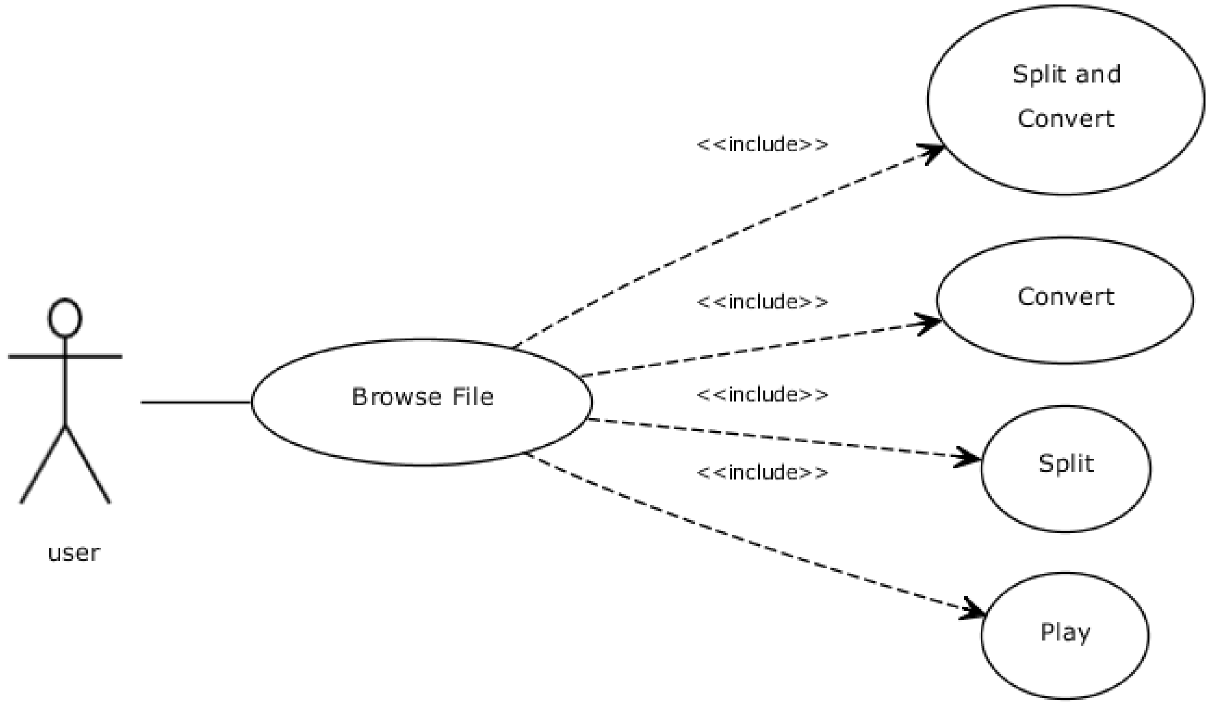
Audio Channels: Default

26.25%

Chapter 7

UML Diagrams

7.1 USE CASE



Use Case 1: **Add files**

Primary Actor: End user

Precondition: A new file is to be browsed.

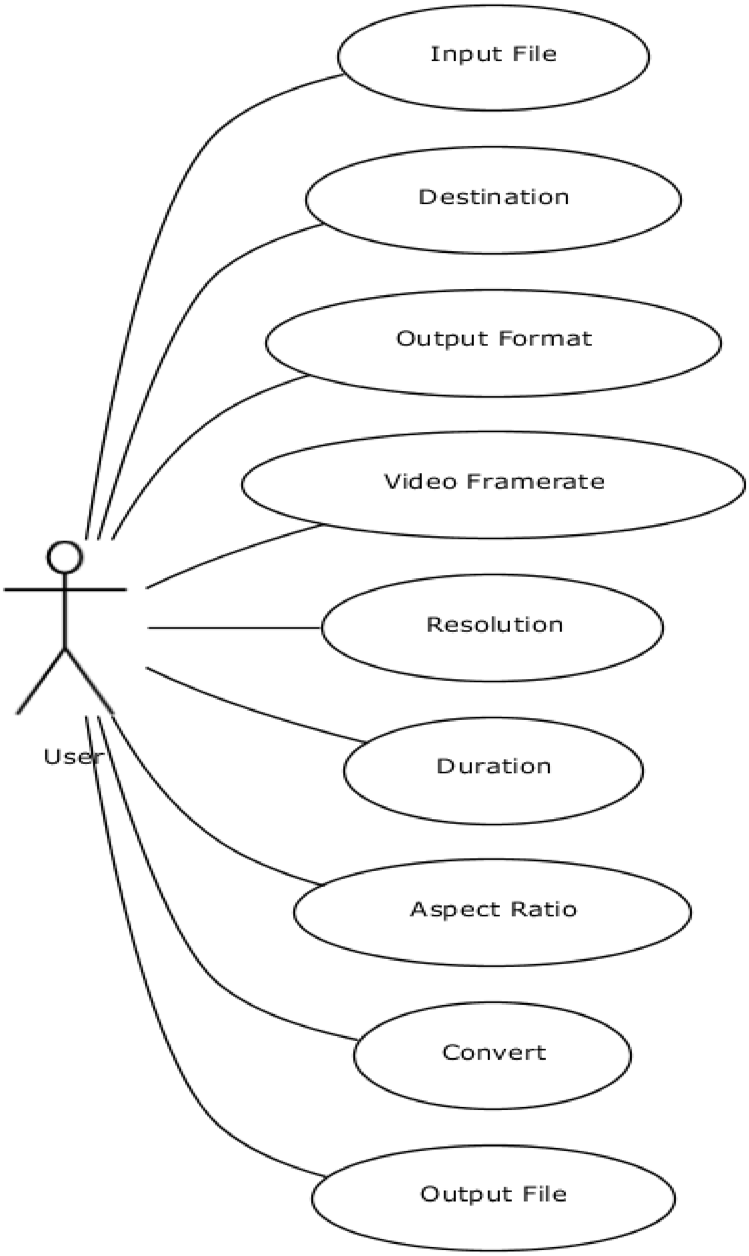
Main scenario:

(a) User can play a file from the list.

(b) User can split a file from the list.

(c) User can convert files as many as needed.

(d) User can split and convert the files.



Use Case 2: **Video to Video conversion**

Primary Actor: End user

Precondition: A new file is to be browsed.

Main scenario:

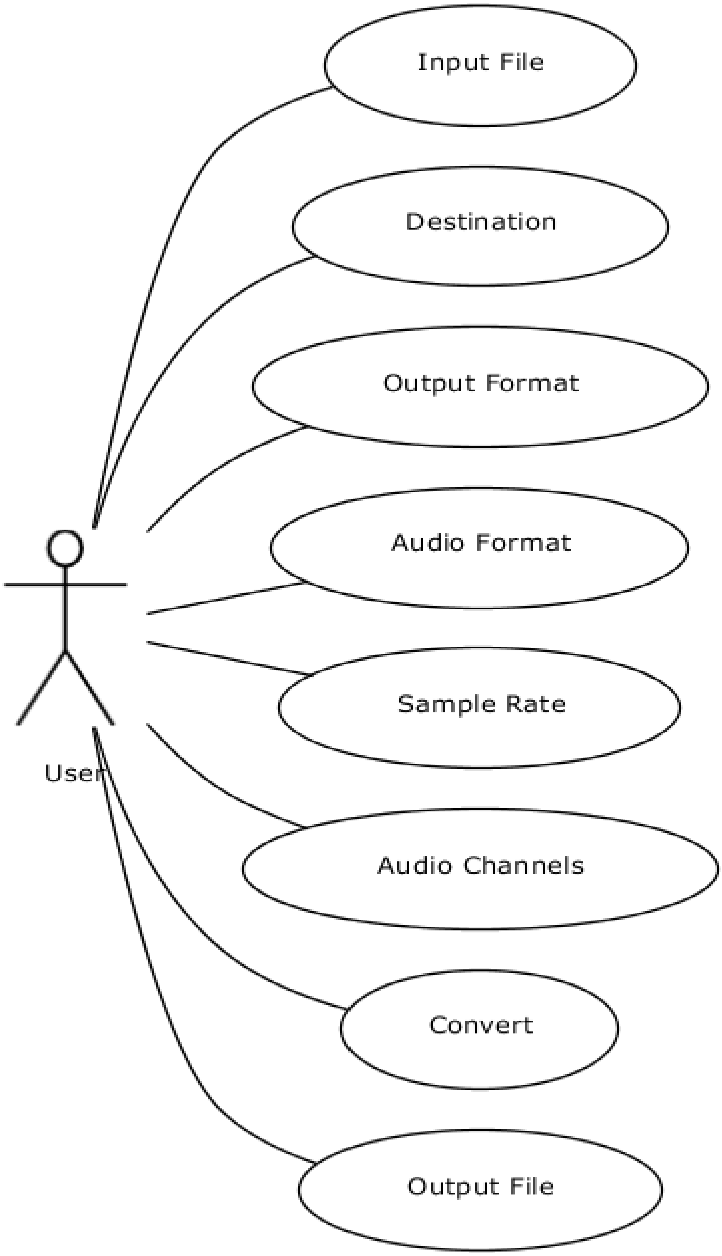
(a) User can browse the videos.

(b) User can change the video’s aspect ratio.

(c) User can change the video’s resolution.

(d) User can change the video’s frame rate.

(e)User can select a particular format.



Use Case 3: **Video to Audio conversion**

Primary Actor: End user

Precondition: A new file is to be browsed.

Main scenario:

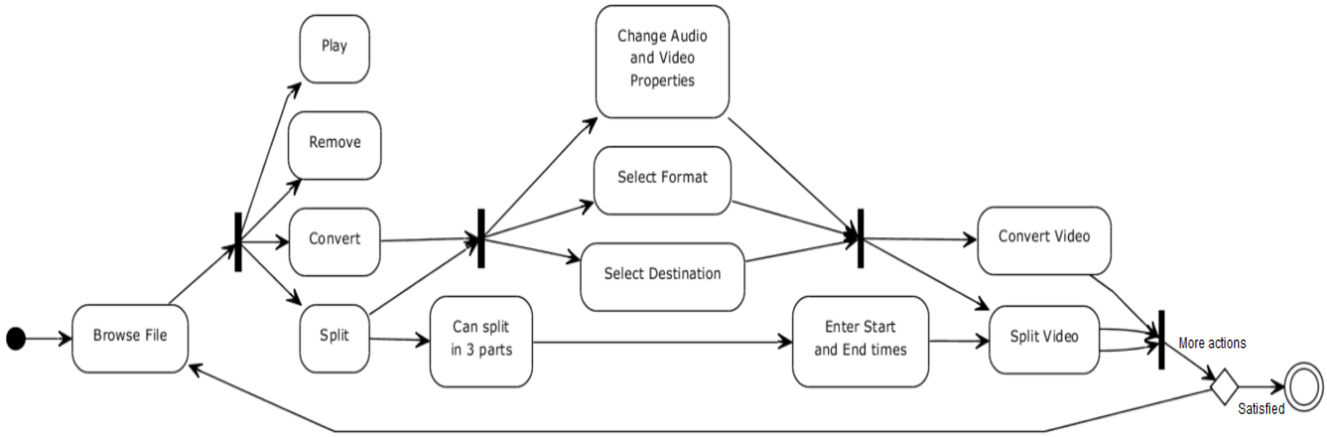
(a) User can browse the videos.

(b) User can select the audio bit rate.

(c) User can select the audio channels.

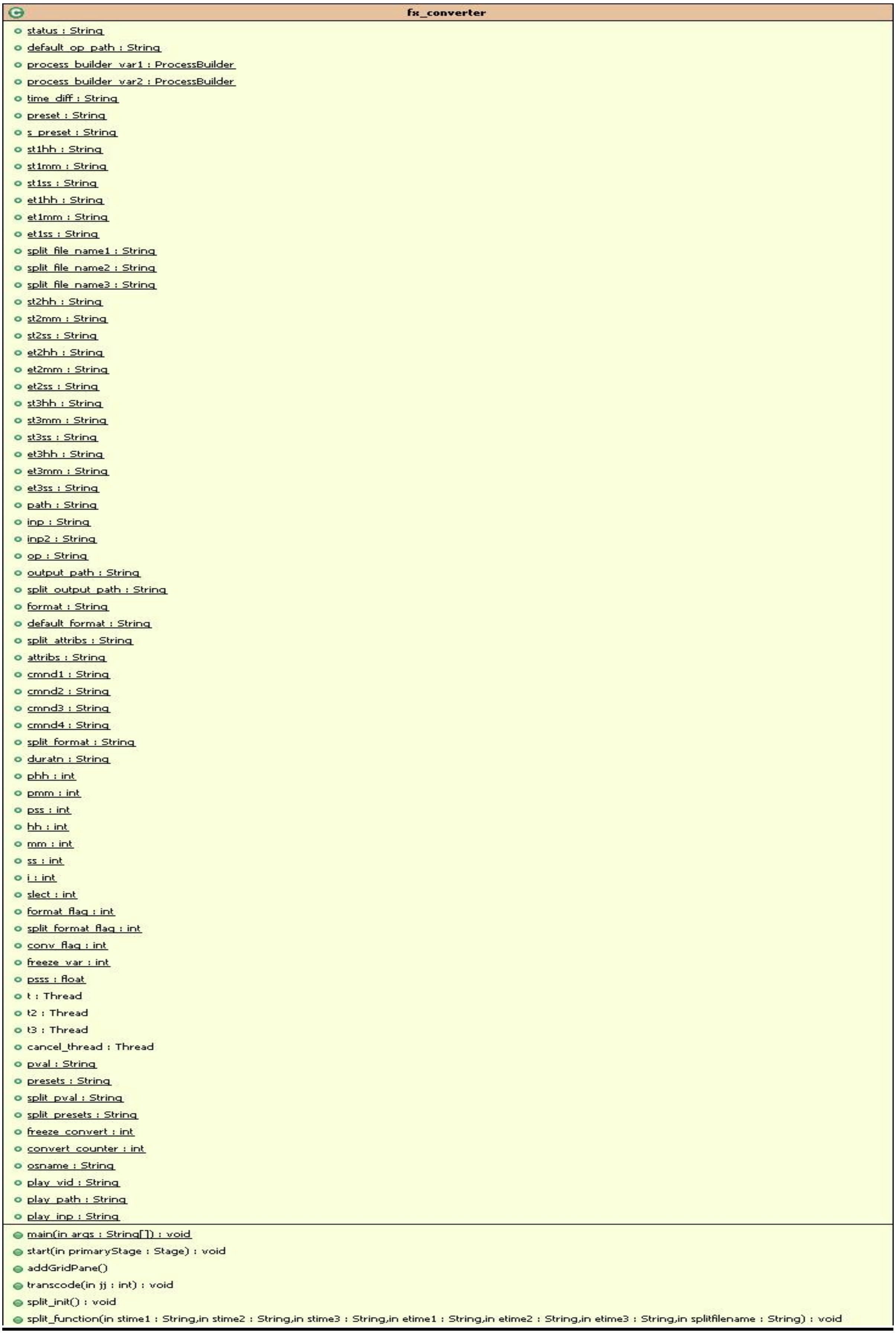
(d) User can select a particular audio format and convert.

7.2 ACTIVITY DIAGRAM

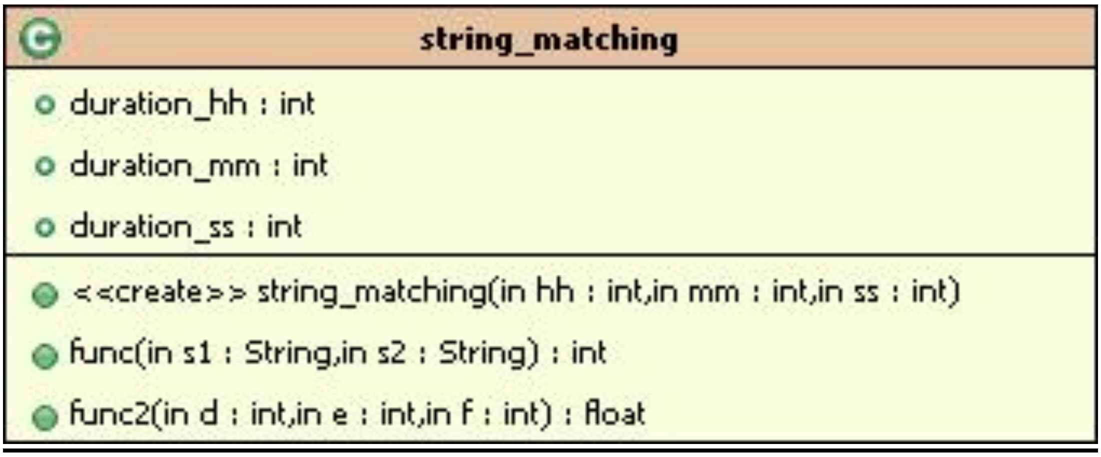


1. Browse files to the list.
2. Can drag and drop multiple files
3. Select a file to play.
4. Can select a file and remove.
5. Can clear the whole list
6. Change audio video properties
7. Convert all the files
8. Select a file to split
9. Enter start and end times, and split.
10. Want to split more, browse again.

7.3 CLASS DIAGRAMS



```
fx_converter
status : String
default_op_path : String
process_builder_var1 : ProcessBuilder
process_builder_var2 : ProcessBuilder
time_diff : String
preset : String
s_preset : String
st1hh : String
st1mm : String
st1ss : String
et1hh : String
et1mm : String
et1ss : String
split_file_name1 : String
split_file_name2 : String
split_file_name3 : String
st2hh : String
st2mm : String
st2ss : String
et2hh : String
et2mm : String
et2ss : String
st3hh : String
st3mm : String
st3ss : String
et3hh : String
et3mm : String
et3ss : String
path : String
inp : String
inp2 : String
op : String
output_path : String
split_output_path : String
format : String
default_format : String
split_attribs : String
attribs : String
cmnd1 : String
cmnd2 : String
cmnd3 : String
cmnd4 : String
split_format : String
duratn : String
phh : int
pmm : int
pss : int
hh : int
mm : int
ss : int
i : int
slect : int
format_flag : int
split_format_flag : int
conv_flag : int
freeze_var : int
psss : float
t : Thread
t2 : Thread
t3 : Thread
cancel_thread : Thread
pval : String
presets : String
split_pval : String
split_presets : String
freeze_convert : int
convert_counter : int
osname : String
play_vid : String
play_path : String
play_inp : String
main(in args : String[]) : void
start(in primaryStage : Stage) : void
addGridPane()
transcode(in jj : int) : void
split_init() : void
split_function(in stime1 : String,in stime2 : String,in stime3 : String,in etime1 : String,in etime2 : String,in etime3 : String,in splitfilename : String) : void
```



string_matching

duration_hh : int

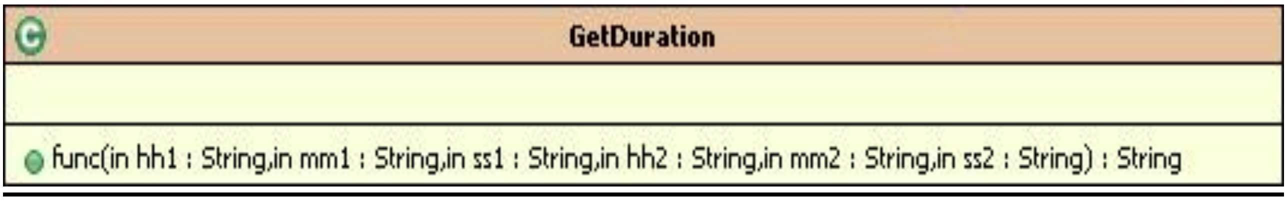
duration_mm : int

duration_ss : int

<<create>> string_matching(in hh : int,in mm : int,in ss : int)

func(in s1 : String,in s2 : String) : int

func2(in d : int,in e : int,in f : int) : float



GetDuration

func(in hh1 : String,in mm1 : String,in ss1 : String,in hh2 : String,in mm2 : String,in ss2 : String) : String

Chapter 8

Conclusion and Future Work

To conclude with, all the assigned tasks have been successfully completed by the team and the above mentioned features are integrated with the project.

Irrespective of this, some future improvements can be done like:

1. File names with commas cannot be converted by this converter. It needs some amendment.

2. Implementation of various splits in the same video. Here a video can be split in maximum 3 parts.

3. HandBrake can be used instead of FFmpeg to improve the efficiency.

4. Design Linux version.

Chapter 9

Bibliography

1.http://www.ffmpeg.org

2.http://docs.oracle.com/javafx/index.html

3.http://www.stackoverflow.com

4.http://www.wikipedia.com