## ****Documentation for Image Resizer and Video Converter in C++****

This documentation covers two tasks: an image resizer library using the libav C library and a video converter program that converts videos to MP4 format, extracts thumbnails, and resizes the thumbnails.

### ****[TASK 1] Image Resizer Library****

#### ****1. Overview****

The ImageResizer library provides functionality to resize image files while preserving their aspect ratio using the libav library. It exposes a function for resizing images to given widths.

#### ****2. Code Structure****

* **Header File:** ImageResizer.h
  + Declares the ImageResizer class, which contains the resizeImage function.
* **Source File:** ImageResizer.cpp
  + Implements the resizeImage function using libav to handle image decoding, scaling, and encoding.

#### ****3. ImageResizer.h****

#ifndef IMAGERESIZER\_H

#define IMAGERESIZER\_H

#include <string>

class ImageResizer {

public:

bool resizeImage(const std::string &inputFile, const std::string &outputFile, int targetWidth);

};

#endif

* **Class:** ImageResizer
  1. **Method:** bool resizeImage(const std::string &inputFile, const std::string &outputFile, int targetWidth)
     + Resizes the image specified by inputFile and saves the output to outputFile with the specified targetWidth, while preserving the original aspect ratio.
     + Returns true if the resizing is successful, otherwise returns false.

#### ****4. ImageResizer.cpp****

* The implementation performs the following steps:
  1. Decodes the input image using libav.
  2. Calculates the new height to maintain the original aspect ratio.
  3. Uses sws\_scale to resize the image.
  4. Encodes the resized image and saves it to the specified output file.

### ****[TASK 2] Video Converter and Thumbnail Extractor****

#### ****1. Overview****

The VideoConverter class provides methods for converting video files to MP4 format, extracting a thumbnail, and resizing the thumbnail to multiple sizes using the ImageResizer library.

#### ****2. Code Structure****

* **Header File:** VideoConverter.h
  + Declares the VideoConverter class and its methods.
* **Source File:** VideoConverter.cpp
  + Implements the VideoConverter class's methods using libav to handle video processing and image resizing.

#### ****3. VideoConverter.h****

#ifndef VIDEOCONVERTER\_H

#define VIDEOCONVERTER\_H

#include <string>

class VideoConverter {

public:

bool convertToMP4(const std::string &inputFile, const std::string &outputFile);

bool extractThumbnail(const std::string &videoFile, const std::string &thumbnailFile);

bool generateResizedThumbnails(const std::string &thumbnailFile);

};

#endif

* **Class:** VideoConverter
  1. **Method:** bool convertToMP4(const std::string &inputFile, const std::string &outputFile)
     + Converts the input video file to MP4 format.
     + Returns true on success, otherwise returns false.
  2. **Method:** bool extractThumbnail(const std::string &videoFile, const std::string &thumbnailFile)
     + Extracts a thumbnail image from the specified video file.
     + Returns true on success, otherwise returns false.
  3. **Method:** bool generateResizedThumbnails(const std::string &thumbnailFile)
     + Uses the ImageResizer library to generate three different thumbnail sizes: small (250px), medium (350px), and large (650px).
     + Returns true if all resized thumbnails are generated successfully, otherwise returns false.

#### ****4. VideoConverter.cpp****

* The implementation includes:
  1. **Video Conversion (**convertToMP4**):**
     + Uses libav functions to convert any input video format to MP4.
  2. **Thumbnail Extraction (**extractThumbnail**):**
     + Seeks to the middle of the video and captures a frame to use as a thumbnail.
  3. **Thumbnail Resizing (**generateResizedThumbnails**):**
     + Calls the resizeImage function from the ImageResizer library to create three resized versions of the thumbnail.

### ****Main Program****

#### ****1. File:**** main.cpp

#include "VideoConverter.h"

#include <iostream>

int main(int argc, char \*argv[]) {

if (argc != 2) {

std::cerr << "Usage: " << argv[0] << " <video file>" << std::endl;

return 1;

}

std::string inputVideo = argv[1];

std::string outputVideo = "converted\_video.mp4";

std::string thumbnailFile = "thumbnail.jpg";

VideoConverter converter;

if (!converter.convertToMP4(inputVideo, outputVideo)) {

std::cerr << "Failed to convert video to MP4" << std::endl;

return 1;

}

if (!converter.extractThumbnail(outputVideo, thumbnailFile)) {

std::cerr << "Failed to extract thumbnail" << std::endl;

return 1;

}

if (!converter.generateResizedThumbnails(thumbnailFile)) {

std::cerr << "Failed to generate resized thumbnails" << std::endl;

return 1;

}

std::cout << "Conversion and thumbnail extraction completed successfully." << std::endl;

return 0;

}

#### ****2. Program Description****

* The main.cpp file is a command-line program that takes a video file as input and performs the following tasks:
  1. Converts the video to MP4 format.
  2. Extracts a thumbnail from the converted video.
  3. Resizes the thumbnail to three different sizes using the ImageResizer library.
* The program generates four output files:
  1. converted\_video.mp4: The converted video.
  2. thumbnail\_small.jpg, thumbnail\_medium.jpg, and thumbnail\_large.jpg: The resized thumbnails.

### ****Dependencies****

* libavformat, libavcodec, libswscale, and libavutil are required to compile and run the program.
* Make sure the development packages for FFmpeg are installed to provide the necessary libraries and header files.

### ****Building the Project****

1. **Compile the Image Resizer:**
   * Compile ImageResizer.cpp to produce ImageResizer.o.
2. **Compile the Video Converter:**
   * Compile VideoConverter.cpp to produce VideoConverter.o.
3. **Link and Create the Executable:**
   * Link the object files (main.o, ImageResizer.o, VideoConverter.o) and the required FFmpeg libraries to create the final executable.

### ****Execution****

To run the program:

./converter <video\_file>

Replace <video\_file> with the path to the input video file.

### ****Conclusion****

The code provides a comprehensive solution for resizing images and converting videos using libav, meeting the specified requirements. The library-based approach allows for easy reuse and future extension.