

SE 3XA3: MIS Gifitti

Team #2, Gifitti
Riley McGee, mcgeer
Pavle Arezina, arezinp
Nicolai Kozel, kozeln

November 13, 2016

Contents

1	Introduction	1
2	Module Declarations	1
3	Module Specifications	2
3.1	Hardware-Hiding Module	2
3.2	Behavior-Hiding Module	2
3.3	Software Decision Module	2
3.4	Image Loading Module	2
3.4.1	Variables	2
3.4.2	Methods	2
3.5	Image Processing Module	3
3.5.1	Variables	3
3.5.2	Methods	3
3.6	Image Conversion Module	3
3.6.1	Variables	3
3.6.2	Methods	3
3.7	GIF Transformation Module	4
3.7.1	Variables	4
3.7.2	Methods	4
3.8	GIF Model Module	5
3.8.1	Variables	5
3.8.2	Methods	5
3.9	View-Model Module	5
3.9.1	Variables	5
3.9.2	Methods	6

List of Tables

1	Revision History	1
----------	-----------------------------------	----------

List of Figures

1 Introduction

This document describes the internal workings of each module implemented as a part of Gifitti. Described are the methods, input/output, as well as any environment and state variables that the module uses. This document will continue to be updated as more methods and functionalities are added to each module.

2 Module Declarations

This section simply lists the modules used by Gifitti. More details on these modules (and how they interact) can be found in the MG document.

M1: Hardware-Hiding Module

M2: Behaviour-Hiding Module

M3: Software Decision Module

M4: Image Processing Module

M5: Image Loading Module

M6: Image Conversion Module

M7: GIF Transformation Module

M8: GIF Model Module

M9: View-Model Module

Table 1: **Revision History**

Date			Version	Notes
November	12th	1.0		Initial creation of document
2016				
...	

3 Module Specifications

3.1 Hardware-Hiding Module

Since the hardware hiding module is implemented through the OS and is not actually coded as part of Gifitti, there is nothing to add to the MIS.

3.2 Behavior-Hiding Module

This module serves as a communication layer between the hardware-hiding module and the software decision module. This module covers M4.M5, and M8. Again, this module is not actually coded so there is nothing to add to the MIS.

3.3 Software Decision Module

This module includes any data structures and algorithms used in the system that do not provide direct interaction with the user. This module covers M7, and M6. Again, this module is not actually coded so there is nothing to add to the MIS.

3.4 Image Loading Module

This module allows for the conversion of system paths to images in a usable form within the software.

3.4.1 Variables

1. Filepath (Input)
2. Image (Output)

3.4.2 Methods

1. GetFilePath(OpenFileDialog o)
This method uses an open file dialog menu within C# in order to allow the user to select a file (path). This file path is then stored within the FilePath variable.

2. LoadImage(string FilePath)
This method takes as input the filepath and converts it to an image file type using M4. This image is saved in the Image variable.

3.5 Image Processing Module

This module converts the input images into a system usable form.

3.5.1 Variables

1. Image (Input)

3.5.2 Methods

1. ConvertTo(FileType ft, Image f)
This method will take as input a filetype and an image and convert the image to the file type requested. This result is then returned.
2. GetFileType(Image f)
This method will return the file type of the image passed as input.

3.6 Image Conversion Module

This module converts the GIF from a GIF Model (M8) to an image exportable by the system (GIF, TIFF, JPEG, PNG, BMP).

3.6.1 Variables

1. allowedFileTypes[]
2. curLoadedGIF (Input)
3. outputImage (Output)

3.6.2 Methods

1. exportGIF(FileType ft)
This will be a public method which will call ConvertTo and return the result of that (if ft is in allowedFileTypes[]).

2. `ConvertTo(FileType ft)`
This method will take as input a filetype and convert `curLoadedGIF` to that file type. This result is then returned.
3. `GetFileType()`
This method will return the file type of the `curLoadedGIF`.

3.7 GIF Transformation Module

This module handles frame by frame manipulation of the GIF Models (M8) in order to complete operations such as resizing and modifying the speed, etc. All the methods and variables of this module are listed below.

3.7.1 Variables

1. `curLoadedGIF` (Input)

3.7.2 Methods

1. `Reset()`
This method will reset the `curLoadedGIF` to its original state when loaded into the program.
2. `Resize(int x, int y)`
This method will resize the GIF to the size passed as parameters.
3. `AddFrame(Frame[] f)`
This method will add to the GIF the frames passed as parameters.
4. `RemoveFrame(Frame[] f)`
This method will remove from the GIF the frames passed as parameters.
5. `Rotate(int x)`
This method will rotate the GIF by the input value in degrees.
6. `ChangeSpeed(int x)`
This method will alter the speed of the GIF by the value passed as a parameter.

3.8 GIF Model Module

This module represents a GIF object within the system. It contains methods and properties that allow the system to properly manipulate the GIF.

3.8.1 Variables

1. Frames[]
2. Speed
3. FileType
4. FilePath (Input)

3.8.2 Methods

1. GetFrames()
This method will return the GIF as an array of frames.
2. Play()
This method will play the GIF by looping through the frames array.
3. Stop()
This method will stop all playback of the GIF.
4. GetFilePath
This method will return the filepath(name) as a string.

3.9 View-Model Module

This module links views to the ViewModels using C# partial class declarations to separate view-model, and view aspects of the system.

3.9.1 Variables

1. curView
2. views[]

3.9.2 Methods

1. `getView()`
This method will return the current view.
2. `changeView(View v)`
This method will change the view to the input view `v`.