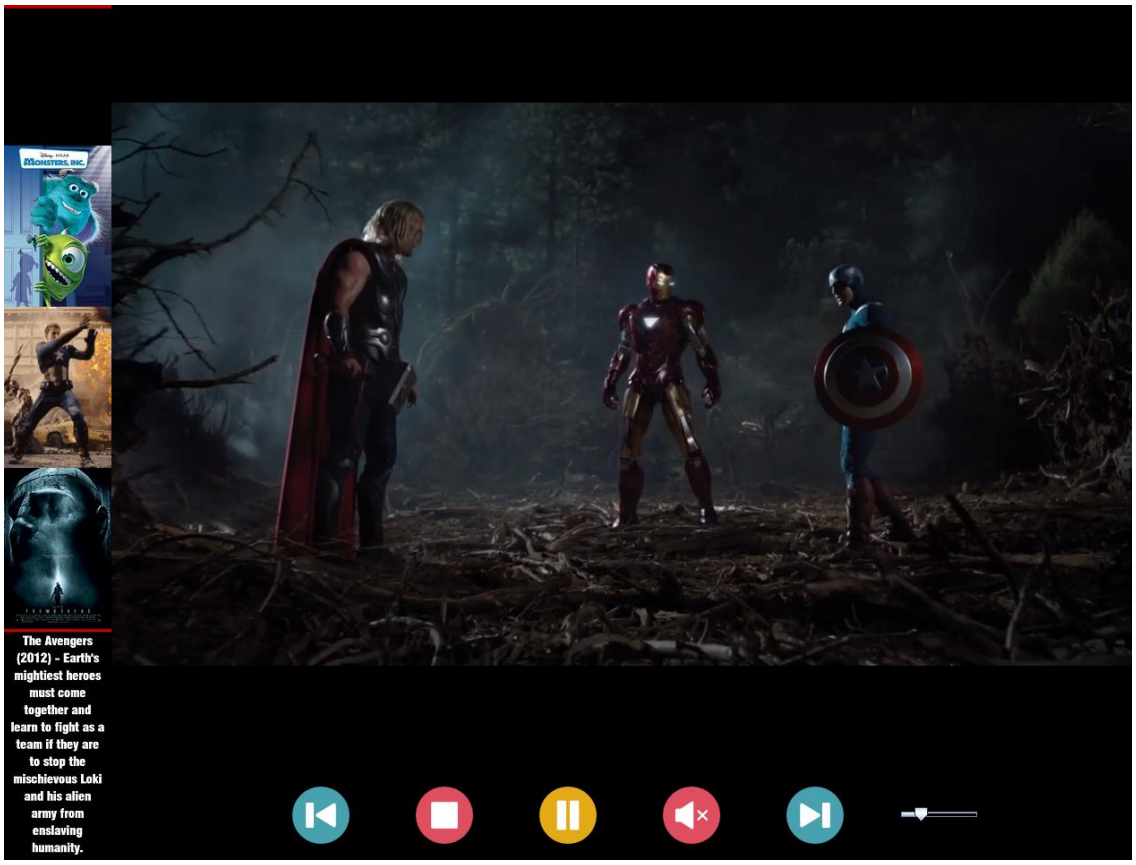


Further Digital Electronics

SOFTWARE ENGINEERING PRINCIPLES & JAVA PROGRAMMING
OVERVIEW DOCUMENT

tm1048 (Partner: rt840) | ELE00003I | 26/05/2017

1. Overview

What?

The specification of the assignment was to make a video player that streamed from a server to a client GUI coded in Java. The server was to send a list of videos over to the client, for the client to then display the video options and prompt the customer to select a video. Upon making the selection, the client was meant to send the chosen video name back to the server. Finally, the server was supposed to stream the video across a specific address at a certain port. The Client then played the streamed content onto the GUI for the customer to watch.

Who and Why?

The aim was to produce a video player that enabled any user anywhere to access our videos within a user interface that was both friendly, intuitive and pleasing to the customer. For this reason, the GUI was a very important factor. As well as being helpful or impressive to the user, it is also important to remember that this is the first release. For this reason, it must be remembered that while adding features is important, it must be working in for the customer. The media fair that this application was designed for is on a fixed date therefore by this point only the bugless features can be implemented.

2. The Design

UI Design

The UI for this video player was said to follow the lines of being "easy and intuitive, improving the user engagement". After collaborating ideas, my partner and I decided carry out the design shown below. For ease of use, the implementation of the video posters in the sidebar [1] was a very good idea to us as it makes options very clear to the user, gaining the engagement of a customer of any age or technical ability on a computer. For intuitiveness, the implementation of standard media player buttons [2] gave a familiar feel for any customer that is familiar with any standard video player (VLC, Windows Media Player etc.).

Group Project

The first thing to be considered was the fact that this was a group project. As there were two programmers working on this project, it was decided between myself and my partner that working on the same parts of the design at the same time would be inefficient. Therefore, past the first 3 iterations (Client/Server communication and basic swing GUI), the full UI design of our application was split into two parts; The Sidebar and the Control Panel. The plan from then on was to work with one another to ensure our implemented designs coincided, but work on separate parts to ensure positive workflow. Initially, I undertook the implementation of the Sidebar, as well as any other tasks necessary to improve playback quality for the customer.

3. Sidebar

The initial idea for the sidebar was to create a section of the UI for easy, clean and engaging access to the videos that the customer might want to display. From following the labs, we had a working comboBox, but we both thought that a pane such as this would improve ease of use, as well as look better in the final design. The original ideas for functionality were as follows:

- Media Player name at top ("MEME Player")
- Current Video Playing status
- Buttons corresponding to each individual video, displaying upon pressing.
- Hide/Show button and settings button.
- Animation upon mouse click to show interaction with button for the user.

The final design for our sidebar is shown in the figure below:

As you can see, the majority of the ideas that we had planned out were implemented correctly. There were some ideas that were reconsidered, and others that were expanded to give a more professional and visually pleasing GUI. The implemented features are:

NOTE: The code for this section was an entirely individual contribution

- Media Player name at top. Replaced with animated current video status upon selection.
- Buttons for each video. GIF of movies presented with mouse hover over. Buttons also disabled for a small period of time after to stop 'spamming' of buttons resulting in stream interference.
- Video description at bottom of sidebar for user to see general synopsis for video shown. This only shows when the mouse hovers over the corresponding button for more immersion when video is playing.
- Pleasing 'red and black' GUI design to split up each component of the bar for visual enhancement.

After creating this bar, there were a couple of things that I did not have time to implement due to time constraints, as well as ideas for future iterations (MEME Player 2, 3 etc.), these were:

- Hide/Show sidebar and settings button.
  - Settings could include screen size, language changes, subtitles etc.
- More user interaction when the button is clicked.
- More visual enhancement
- Right click functionality to give options to view more information about the video.

Methods

Below you can see a breakdown of the methods used to make each component of this sidebar possible:

NOTE: All sidebar GUI implementation was done using the Java SWING Libraries by myself

(http://circe.univ-fcomte.fr/Docs/Java/Tutorial.20060804/ui/features/components.html)

Component (All within MEMEClient.java)	**Methods**
Main JPanel	sideBar(JPanel buttonPanel)
Media Player Name / Playing Status	sideBar(JLabel PlayingStatus), ActionListeners(buttonX)
Selection Buttons	sideBar(JButton button1, button2 , button3)
Video Description	sideBar(JLabel labelDesc)
Button Interaction (GIF)	ButtonX.MouseListener
Button Clicks	ButtonX.ActionListener
Temporary Button Disable	Disable(button, delay)

Tests

To test the functionality of our code, we used a mixture of manual tests (use of mouse to test button feedback, print statements, error messages etc.) and Junit test classes. The tests we carried out on the sidebar were as follows:

MemeClientTest.java

[Joint additions]

videoFileReturnsCorrectValue ()

- This test method checks whether the correct videoFile object was getting fetched from the server upon being requested by the client. This was adjusted to test all three videos and all tests were successful

[Tests created by myself]

Button1Test, Button2Test, Button3Test

- This test method simulates a click of the specified button and sends off that video's videoFile object over to the server, checking to see if the return values are correct.

Button1ImageTest, Button2ImageTest, Button3ImageTest

- This simulates a button click, but this time checks that the desired image matches the icon that is set to the button after the click. This passed for all buttons.

I also performed a significant amount of manual tests throughout the implementation of my design. These consisted of making small changes and running the code to present the program visually and test rigorously to ensure that they weren't producing bugs, ironing them out if found. These consist of:

- Ensuring that a 'mouseOver' action of each button brought the correct change to the icons, as well as ensuring that upon being clicked they still behaved as expecting. This helped me establish that the button disable method I created greyed over the icon for the buttons which was not desired.
- Ensuring that the correct description corresponded to the correct buttons upon hover over. This helped with realizing I needed to set the icon to a blank image for when the mouse exits.

Teamwork

As mentioned above, the design workload of the media player was split after the labs to increase workflow. However, while we did work in the same room, collaborating as we went, upon our agreed completion time my partners work did not quite meet what we had decided, leaving additional work to be done to achieve functionality and consistent UI design. This was ok for me as I was happy to work through the code that she had created, making the improvements needed to get it up to scratch.

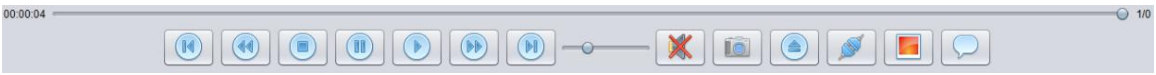
Unfortunately, this led to a situation that I did not anticipate and found difficult to handle. Mistakenly, my lab partner pressed the power button of the PC I was working on in the labs, turning it off and losing my entire days' work which totaled to around 7 hours of straight programming. This meant that we had to work to get our program back to the state it was as soon as possible due to the deadline drawing ever closer. To speed up the process as much as possible, I saw it fit that I recreate everything we had created from the last back up to full completion (the Entire GUI including control panel). I did want to work as a pair, however considering time constraints it was only going to slow down the process of recovery.

Therefore, I consider the work on the control panel [2] also my own contribution to the project, explaining below accordingly.

Control Panel

From completing the labs, we were provided with the VLCJ standard Media Control Panel as shown below:

(https://github.com/caprica/vlcj/blob/master/src/test/java/uk/co/caprica/vlcj/test/basic/PlayerControlsPanel.java)



We both decided that this was a sensible base to start from as it had all the controls that we wanted for the final product of version 1. These were:

- Previous/Next Video Button (Plays previous/next video) [2 buttons]
- Play/Pause Button [1 Button toggle]
- Stop Button [1 Button toggle]
- Mute/Sound Button (Toggles between sound and mute) [1 Button toggle]
- Full Screen Button [1 Button]
- Sound Slider [1 Slider]
- Time Location Slider [1 Slider]

This was a fairly straightforward task as it was adapting the VLCJ Control Panel to match our GUI and also add functionality for Play/Pause toggle and Previous/Next Iteration. The Final Control Panel is seen below:

The implemented features are:

NOTE: The code for this section was an entirely individual contribution

- Previous/Next Buttons that iterate through the videos
- Stop Button
- Pause/Play toggle button that changes icon accordingly
- Mute/Sound toggle button that changes icon accordingly
- Volume Slider

There were again a couple of instances where I refactored away or removed ideas due to time constraints and the evolution of the design. This included:

- Full Screen Button (difficult to implement. Feature would be looked into for next iteration).
- Time Location Slider (difficulty implementing due to streaming of data from the server. Another possible feature for the next version).
- Ability to hide the control panel when a movie is playing.

Methods

Below you can see the methods used in order to create this panel:

NOTE: These were all adapted from the VLCJ template found and adapted by myself: (https://github.com/caprica/vlcj/blob/master/src/test/java/uk/co/caprica/vlcj/test/basic/PlayerControlsPanel.java)

Component (All within PlayerControlsPanel.java)	**Methods**
Creating Controls (Play, Pause etc.) – Also transparency of buttons	createControls();
Layout of Controls	LayoutControls();
Event Listeners for Buttons (and Media Player unchanged).	registerListeners();
Next Previous Action Listener (Setter; called in MEMEClient)	NextPreviousActionListener();

Tests

To test the functionality of our Control Panel code, I used a mixture of manual tests (use of mouse to test button feedback, print statements, error messages etc.). These are as follows:

- Rigorously testing the media buttons upon addition to ensure the action listeners and features inside them worked as they should.
- Checking that the toggle buttons had implemented the icon changes correctly.
- Running modifications to one button to look into the different icon changes such as transparency.

Classes Completed in Labs

The following were completed **together** in the labs, including the formulation of the server, videoFile and XMLReader classes. These are:

MEMEServer.Java	ServerTest.Java
XMLReader.Java	XMLReaderTest.Java
VideoFile.Java	RunBoth.Java – Addition by Partner

The Final Product of our design is displayed below:

Referenced Resources

Loading Screen

https://media.giphy.com/media/lj6840b9wZyjC/source.gif

Avengers Thumbnail

https://en.wikipedia.org/wiki/The_Avengers_(2012_film)#/media/File:TheAvengers2012Poster.jpg

Monsters Inc. Thumbnail

http://movies.disney.com/monsters-inc

Prometheus Thumbnail

https://vignette1.wikia.nocookie.net/avp/images/7/7f/Prometheus_Poster.png/revision/latest?cb=20170125135410

Icon Pack Controls Panel

http://www.flaticon.com/packs/essential-collection

Monsters Inc. GIF

https://68.media.tumblr.com/78ae861f1599529ddf03b2f0353d8a68/tumblr_o4pur4aOti1slugvp01_500.gif

Avengers GIF

https://lovelace-media.imgix.net/uploads/859/e1d8d200-eed8-0133-8016-0e31b36aeb7f.gif?w=740&h=417&fit=max&auto=format

Prometheus GIF

https://media.giphy.com/media/qyAIT3fMKjskM/giphy.gif

Production of Now Playing Gifs and MEMEPlayer

https://giphy.com/create/gifcaption

Avengers Description

http://www.imdb.com/title/tt0848228/?ref_=nv_sr_2

Prometheus Description

http://www.imdb.com/title/tt1446714/?ref_=nv_sr_1

Monsters Inc. Description

http://www.imdb.com/title/tt0198781/?ref_=nv_sr_1 -