COMP3015 Project Specification Form

# Member Information

|  |  |  |
| --- | --- | --- |
|  | Full Name | Student ID |
| Member 1 | Lin Zi Hen | 17211956 |
| Member 2 | Tang Wei | 17202396 |

# KidPaint Version

Select the implemented approach by adding a “X” in the bracket:

|  |  |  |
| --- | --- | --- |
| (X) Basic Approach | ( ) Advanced Approach | ( ) Peer-to-peer Approach |

# Application Layout Protocol

Describe the transferred data for a chat message:

*Hint: you may use the method shown on page 6 of the lab manual named “Socket Programming – Transmission Control Protocol I”*

|  |
| --- |
| A DataOutputStream global variable has been declared. When the user types any input, the content will be concatenated with the username into one String variable. Then, an identifier int (-1) will be written in the stream data follow by length of the String variable and lastly the String variable which contains the username and message content. The data will then be sent to server which has established a TCP connection.  On the server side, server will decode the data with respective order (identifier, length, content) and broadcast to all clients with the same order.  Back to the client side, the client will receive the data and decode in similar order. If the identifier of the data is -1, the content will be appended to chat area. |

Describe the transferred data for the color change of a single pixel:

|  |
| --- |
| The same output stream of chat message is shared with color change mechanism. When the color changes, the coordinate of the pixel (column and row), and the value of pixel will be concatenated into one String variable. An identifier int (0), length of the String variable and String variable is sent in order.  On the server side, server behaves the same as data transmission of chat message.  Back to the client side, the client will receive and decode the data. If the identifier of the data is 0, the content will be updated to the respective pixel of UI and UI will be updated. |

Describe the transferred data for the color change of an area (if any):

|  |
| --- |
| We have chosen to update the data pixel by pixel. |

# Program Testing Procedure

Provide an instruction for running the programs:

|  |
| --- |
| **Start the server-side program (if any):**  **Start the client-side program:**  Run the program, enter username and enjoy! |

# Additional Features:

Explain the features including the logic flows and the procedures for testing the features:

|  |
| --- |
| Pen and Bucket eraser: A global Boolean variable ‘erase mode’ is declared and initialized as false. When erase mode enable, it behaves the same as the respective pen and bucket mode, but change to pixel into black. To test, first draw something using pen and bucket, and test the erase mode of both respectively.  File Chooser: For save and import file, instead of import and export to a pre-defined destination, we allow the user to choose their path. To test save, click on the button, choose a folder and type the file name including file type (e.g. test.txt) and save. For import, click on the button, select a file and click open.  Back-end data: In case there will be a second client join later than the first client by a period of time, we’ve setup a data array at server so that when the client first connect to the server, the client UI will display the up-to-date sketch. To test, first join the server as client 1 and start drawing something. And launch another program as client 2 to join. You will see the UI of client 2 will be displaying the same client 1. |

# Data Saving

Describe how the system stores the pixel data in a file. You may also paste the sample below.

|  |
| --- |
| When the save button is toggled, <consoleToFile> function will be called with a path and the recent data (int[][]) as parameter. In <consoleToFile> function, a DataOutputStream will be initialized with the path from the parameter. Then, the stream will write the data pixel by pixel using a double for loop. |