Snake and Ladder Board Game

Documentation

Team 04 - Serpentiform

Chen, JiaJia

Chen, Shuying

Lin, Fei

Date: 21-Apr-2020

Revision: 0.8

**Table of Contents**

[Executive overview 4](#_Toc38380132)

[Audience 4](#_Toc38380133)

[Application intentions 5](#_Toc38380134)

[Assumptions made for this project 6](#_Toc38380135)

[*Gantt chart* 7](#_Toc38380136)

[*Class and method overview* 10](#_Toc38380137)

[<Project> UML 14](#_Toc38380138)

[Client and Server GUI 15](#_Toc38380139)

[Networking connections & Protocols 16](#_Toc38380140)

[Communication class 17](#_Toc38380141)

[Data used 19](#_Toc38380142)

[Data files 19](#_Toc38380143)

[Punch List used 20](#_Toc38380144)

[To do: 20](#_Toc38380145)

[Done: 20](#_Toc38380146)

[Unresolved Issues 20](#_Toc38380147)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason for Change** | **Version** |
| ALL | 18-Mar-20 | Initial release | 0.1 |
| Fei Lin | 18-Mar-20 | Adding information to application intentions | 0.2 |
| Jiajia Chen | 18-Mar-20 | Adding information to the audience | 0.2 |
| Shuying Chen | 18-Mar-20 | Adding information to overview and assumption | 0.2 |
| ALL | 18-Mar-20 | the date of task, basic main function that might be use, start working on punchline | 0.2 |
| ALL | 04-Apr-20 | Insert Gantt Chart, which is a plan of what to do next. Assigned what everyone needs to do. | 0.3 |
| JiaJia Chen | 10-Apr-20 | Reorganize the design doc. Fill up the Punch List used and start to put in the file and the resource we use. | 0.4 |
| All | 11-Apr-20 | Add information about the overview of the classes and functionality. | 0.5 |
| JiaJia, Fei | 11-Apr-20 | UML Complete | 0.5 |
| All | 11-Apr-20 | Client GUI Complete | 0.5 |
| JiaJia | 12-Apr-20 | Explain how client connect to the server and the port number(s) we are going to use. A brief description about the port number. Protocol interface code for both client and server. | 0.6 |
| Shuying | 12-Apr-20 | Fill up the chat interaction between clients and servers. Explanation of how the chat works. | 0.6 |
| All | 12-Apr-20 | State the data and the resources | 0.6 |
| All | 12-Apr-20 | State the problem we have so far | 0.6 |
| Fei, shuying | 13-Apr-20 | revise class and method, and UML | 0.7 |
| All | 21-Apr-20 | updated design document | 0.8 |

# Executive overview

This project is an online board game for single or multiple users, the game is snake and ladder. This project will have two or more screens, one screen is for server, which shows all the information about the message between the server and clients. Another screen is used for playing the actual game, with the image of the snake and ladder and the players. There will be one more screen that is basically designed for multiplayer, which will be the group chat and the connection between all the users, everyone is allowed to join the chat, and type the words or sentences, and everyone is able to see the text.

When the game starts, the user or users will need to toss a die, the number of the die will determine the number of steps that they can move. For example, if a user rolls a 4, then the user would move their piece four places. The user will move their pieces from left to right, starting at 1, following the numbers on the board, then the next row from right to left and repeat. The first user that reaches the highest space on the board, which is 100, wins the game. In addition, the user needs to roll the exact number to reach 100.

Our game will utilize a modern and clean design in order to make the game enjoyable to play and use regularly.

# Audience

The target audience can be the Game app company or learning/teaching website that helps their visitor learn better about creating a game using java.

We will introduce an approachable and easy-to-learn guide through Java. Most of our content will be informational material that can be easy to understand. Our inner workings of application can be attracted to those who are game development companies or teaching websites used as learning game development or learning what java language can do. Our project is helpful for Game development for beginning that it can be a resource to succeed.

We hope to engage the reader to review the materials. This application gives a sense of the opportunity to use the information we provide, such as show the players our code, explain how it works, and allow players to learn Java code. The code will help the new coder know better about programming in java and creating a game.

## Application intentions

The purpose of this application is to let our audience learn the basic sequence and pattern of numbers. They will consistently be rolling dice and counting the numbers on the board.

This game can help our audience develop social language skills in that bad things can happen suddenly and without warning. For instance, a player may be well ahead in the game, expecting to win easily. The player finishes his/her turn on the head of the longest snake and falls dramatically to the last place. This type of reversal of fortune can be a shock to a young audience and can be a valuable and safe early life lesson. It helps to prepare our young audience for life's little reversals. An added bonus, of course, is that in Snakes and Ladders you can still win the game even if you're well behind the leader. This is achieved by vaulting other players when you land on one of the tall ladders and then shimmy to the top.

Our target audience can be people of any age and gender. For example, for the younger user, from age 3 to 8, they can learn some basic computer such as clicking the mouse, typing the text, counting, and knowledge of playing basic board games. For the older user, from age 60 to 80, it will be a great game to have some exercise for their hands, eyes, and counting in mind. This game is approved for all, everyone who is getting bored, or want to relax they can play this game.

# Assumptions made for this project

The assumptions made for this project are that the user has a basic knowledge of using a computer, such as opening the java files, using the command line, and having an internet connection.

In addition, basic knowledge about the Snake and Ladder game, roll a die, or other board game.

* When a player lands on a top of a snake, their playing piece will slide down to the bottom of the snake.
* For the ladders, when a player lands at the base of a ladder, it immediately climbs to the top of the ladder.
* The winner of this game will be the first user that reaches the highest space on the board, which is a hundred.
* The restriction
  + When a player almost reaches 100, the player needs to roll an exact number that reaches 100 in order to win the game.
  + If players roll the number that is over 100, they stay in the same space where they roll the dice.

# 

# *Gantt chart*

|  |  |  |
| --- | --- | --- |
| **Task** | **When** | **Who** |
| Assigned project ideas | 2/26/2020 | All |
| Individually think about possible projects | 2/28/2020 | All |
| Project ideas discussed in class | 3/6/2020 | All |
| Discuss projects with possible team(s) / team member(s) | 3/7/2020 | All |
|  |  |  |
| Teams formed | 2/26/20 | Member: JiaJia, Shuying, Fei |
|  |  |  |
| Initial design document | 3/12/20 | All |
|  |  |  |
| Design Document assigned | 3/10/20 | All |
| Design discussions {incl: scope of project} | 3/18/20 | All |
| Update design document | 3/20/2020 | JiaJia |
| Submit to dropbox | 3/20/20 | shuying chen |
|  |  |  |
| Produce project plan | 4/1/2020 | All |
| Develop plan | 4/5/20 | All |
|  |  |  |
| Meeting to review plan | 4/5/20 | All show up |
| Submit to dropbox | 4/6/20 | JiaJia |
|  |  |  |
| Design task |  | Main role: Jiajia; Helper: fei, Shuying |
| Client Design | 4/8/20-4/9/20 | All |
| Design task | 4/8/20-4/9/20 | All |
| Chat Design | 4/8/20-4/9/20 | All |
| Server Design | 4/8/20-4/9/20 | All |
| design GUI | 4/8/20-4/9/20 | All |
| prepare dice images | 4/12/20 | Jiajia |
| prepare background images | 4/12/20 | jiajia |
|  |  |  |
| Code Client | 4/10-4/20 | Main role: Fei; Helper: Jiajia, Shuying |
| Start create game | 4/10/20 | Main role: Fei; Helper: Jiajia, Shuying |
| create background | 4/10/20 | Fei, JiaJia |
| design algorithm | 4/12-4/20 | All |
| methods for move the pieces | 4/12-4/20 | Fei |
|  |  |  |
| Code Server | 4/12-4/20 | Main role: Shuying; Helper: Fei, JiaJia |
| create multiServer | 4/12-4/20 | Main role: Shuying; Helper: Fei, JiaJia |
| change the GUI (position of the client) for all clients | 4/12-4/20 | Main role: Shuying; Helper: Fei, JiaJia |
| extends thread | 4/12-4/20 | Main role: Shuying; Helper: Fei, JiaJia |
|  |  |  |
| Code Chat | 4/12-4/20 | Main role: Shuying; Helper: Fei, JiaJia |
| create GUI | 4/10/20 | Main role: Shuying; Helper: Fei, JiaJia |
| create actionlistener | 4/12-4/20 | jiajia |
|  |  |  |
| Testing | 4/20 - 4/26 | Jiajia, fei |
|  |  |  |
| Submit all programs | 4/27/2020 | JiaJia |
|  |  |  |
| Final Design Document |  |  |
| Complete final document | 4/12/2020 | All |
| Team review | 4/20/2020 | All |
| Submit document | 4/13/2020 | JiaJia |
| The Final Project and Presentation | 4/27/2020 | All |
| Submit Individual peer reviews | 5/4/2020 | All |

# *Class and method overview*

Overview of the classes and functionality.

**Client:**

* Main method
  + call constructor to start the gui
* Client constructor
  + call the chat GUI for connection
* winner method
  + This is the method that will pop up the winner message window
  + parameter(message) - the message that will show in the window
  + no return values
* game method
  + This is the method to create the GUI for a game which contains the board game by using the Button, players images, toss dice button, restart button, and help menu bar that contain the about message.
  + no return and no parameter
* player\_move method
  + This is the method that will move the other player to their position by replace board game image to player image
  + parameter(move) - the position of the player moves to
  + parameter (player)-which player to move
  + no return values
* own\_move method
  + This is the method that will move the player itself to their position by replacing the board game image to the player image and calculate the final position.
  + parameter (move) - the position of the player moves to
  + parameter(player)- which player to move
  + no return values
* afterMove method
  + This is the method that will move This is the method that will change back to background image after the player moves.
  + parameter (move) -the position of the player that move before
  + no return values
* dice method
  + This is the method that will toss dice for the player to move.
  + return the number for player to move
  + no parameter
* diceImage method
  + This is the method that changes the dice image.
  + parameter (step) - the number of dice want to change
  + no return values
* exit method
  + exit the program
  + no parameter and no return value
* restart method
  + start the new game
  + no parameter and no return value
* chat method
  + This is the method to create the GUI for chat which contains a send button, connect button, and menu bar that contain the help message.
  + no parameter and no return value
* connect method
  + connect to server, use to received and send message
  + no parameter and no return value
* sendGame method
  + send all the information to the server
  + parameter(message) - the message that will show in the window
  + no return values
* actionPerformed
  + buttons and menu perform
  + parameter (ActionEvent ae) - button that has been clicked.
  + no return values
* loadFile method
  + pop up the about and help message from the text file
  + parameter (fileName)- the name of the text file
  + no return values
* run method
  + This run method will receive the message from the server and display in the chat board and if the message is about the game message of the player, move that player to its position and display the dice that player tosses.
  + no parameter and no return value

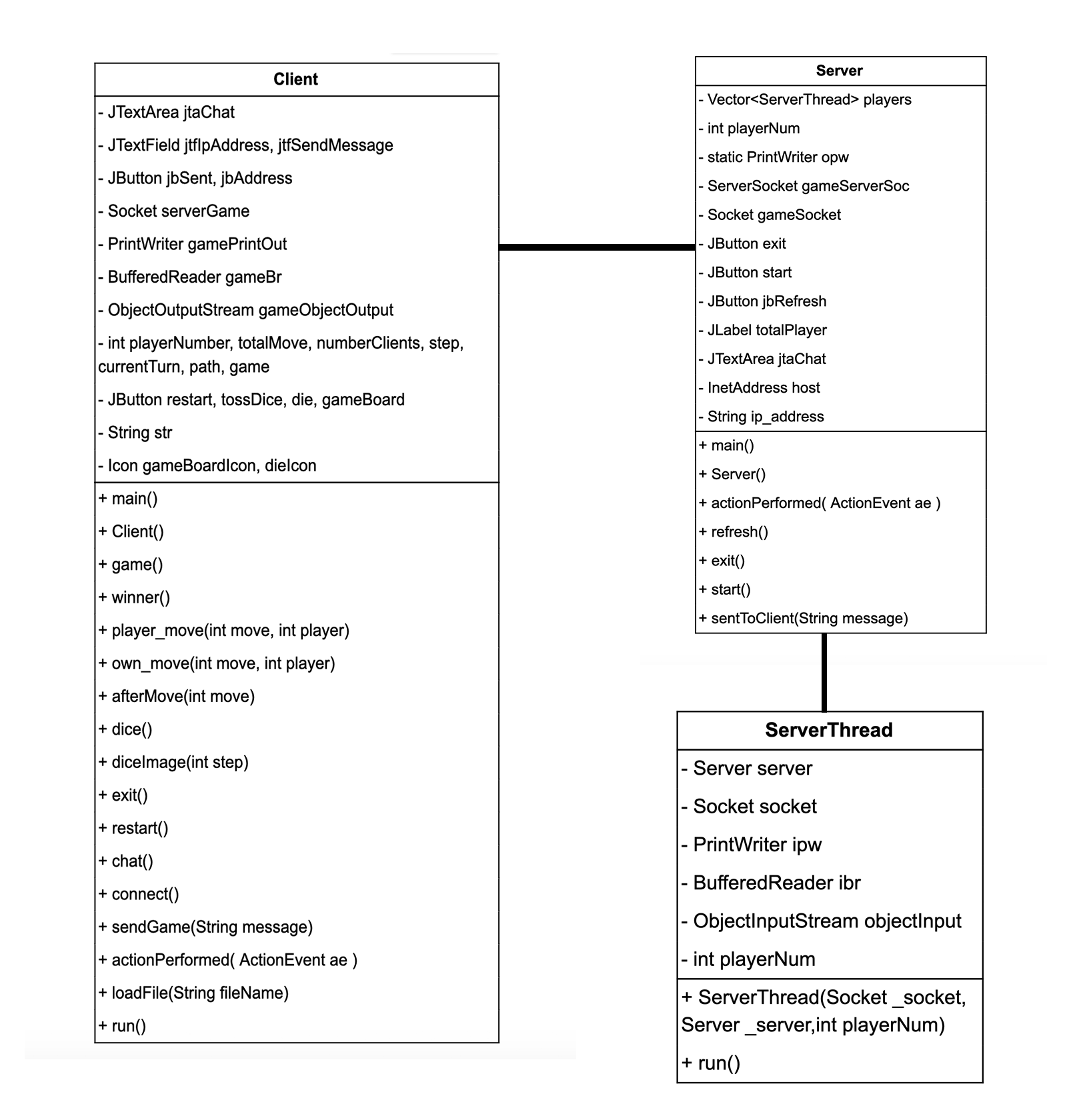
**Server:**

* Main method
  + call constructor to start the connection
* Server constructor
  + determine the how many clients then send to the server
  + Show IP address for connect
  + Create a GUI show all message for chat
  + accept socket
  + print total players
  + no parameter
* actionPerformed
  + buttons and menu perform
  + parameter (ActionEvent ae)
  + no return values
* refresh method
  + show total players in connection
  + no parameter and no return value
* exit method
  + exit the program
  + sent server down to all clients
  + close socket
  + no parameter and no return value
* start method
  + use to start the game gui
  + when game start, sent start game to all the clients
  + first player that get connect to the server will play the game first
  + no parameter and no return value
* sendToClient method
  + send all the information to the all clients
  + parameter String message, no return value

**ServerThread:**

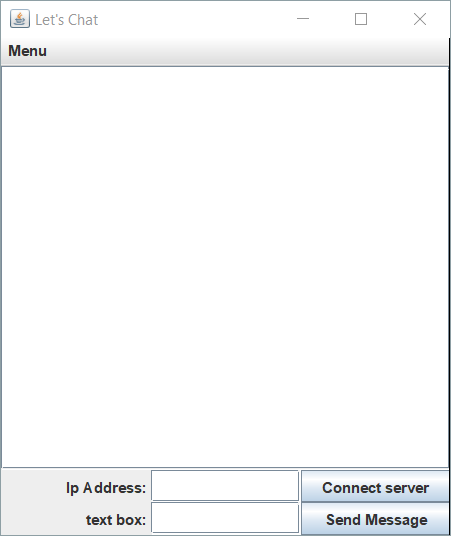
* ServerThread class
  + thread class for multi-client
* ServerThread constructor
  + no return values
  + parameters socket, server, playerNum
* run method
  + read message from the clients
  + send message to the clients
  + call the exit method
  + announce winner

# <Project> UML



# Client and Server GUI

GUI for clients Chat and Game board

A screenshot of a cell phone

Description automatically generated

GUI for server:

A screenshot of a social media post

Description automatically generated

# Networking connections & Protocols

Client connection to server: Players can input the IP address to connect to the server.

Port number(s): 16789 for game and chat

A port number is a way to identify a specific process to which an Internet or other network message is to be forwarded when it arrives at a server. Port defines an application on the network for a specific client and server.

Protocol interface code for both client and server:

import java.net.\*;

import java.io.\*;

**Server:**

use for connection

gameServerSoc = new ServerSocket (16789);

gameSocket = null;

gameSocket = gameServerSoc.accept();

ServerThread ts = new ServerThread(gameSocket, this, playerNum);

**Client:**

use for connection

serverGame = new Socket(jtfIpAddress.getText(), 16789);

gameObjectOutput = new ObjectOutputStream( serverGame.getOutputStream() );

gamePrintOut = new PrintWriter (serverGame.getOutputStream());

gameBr = new BufferedReader (new InputStreamReader (serverGame.getInputStream()));

# Communication class

**Chat interaction between Clients and Server.**

To run the program, the server runs first, then multiple clients connect to the server. When the clients are connected to the server, they can start the game by clicking the start buttons on the server. When clients connect to the server, they can’t send messages until they start the game. The server will display how many players are in the conversation. When the client connects, the server will send a welcome message to all the clients. When one of the client's exits, all the other clients will exit too, including the server.

Clients are allowed to talk to other clients by using Chat Gui. There is a text field and text area for displays and typing in the words, sentences, and numbers. Clients can send the message by typing the words and click the send button. Once the client sends a message out, the server will read it and receive the message. Then the server will transfer the message to all other clients. The server will be tracked on all the messages from the clients, including the message from the game board.

After the server starts the game. The server will release the game board GUI for each client. Each player will wait until their turn to toss the dice. The players can tell by their chat receive message or their toss dice button can be clicked. When the die is tossed, it will randomly display a number from one to six. Then the dice method will return a value to the move method. The move method sends the return value to the server, and the server will receive the number of dice and display the player move for all the clients. Once one of the players reaches position 100, the client will send a message to the server, and the server will announce the winner for all clients. The players can determine to play again by clicking the restart button or exit. On the game board, the players will see a pop-up window showing the winner.

When the server or one of the clients is disconnected, it will automatically shut down for all the connected clients.

|  |  |  |
| --- | --- | --- |
| **Client** | **Communication** | **Server** |
|  |  | Startup |
|  |  | Waits for client to connect  Accept connection |
| Client connection | Connection |
| new Clients connected | welcome message | Server print out which client is joining (welcome message) |
| Client sends text out  (information) | some information | Server reads (information) |
|  |  | Server process the (information)from Clients, Sends (message) to client(s) |
| Client receives (message) | message | Sends (message) to client(s) |
| Client (reaches highest position and win) | win message | Server received win message, read and process it |
| All the clients received the <win message> |  | Server declared the Winner by sent out the (win message) |
|  |  |  |
| Client click the toss die button, which sent out the (move), | move  move method | Server receive the number of move, and process the (move method) |
| *All Clients’ game gui are change* |  | *Server sent out the process of (move method) and change all clients’ game Gui* |

# Data used

Image: <https://www.vectorstock.com/royalty-free-vector/snakes-and-ladder-game-template-vector-23813393>

Sample of Snake and Ladder java code: <https://rosettacode.org/wiki/Snake_And_Ladder#Java>

Dice images will be found online.

[https://image.baidu.com/search/detail?ct=503316480&z=0&ipn=d&word=筛子%201到6&step\_word=&hs=0&pn=2&spn=0&di=17930&pi=0&rn=1&tn=baiduimagedetail&is=0%2C0&istype=0&ie=utf-8&oe=utf-8&in=&cl=2&lm=-1&st=undefined&cs=4060570382%2C451230636&os=1234429836%2C3104516760&simid=3509062187%2C720304968&adpicid=0&lpn=0&ln=1260&fr=&fmq=1587502126523\_R&fm=&ic=undefined&s=undefined&hd=undefined&latest=undefined&copyright=undefined&se=&sme=&tab=0&width=undefined&height=undefined&face=undefined&ist=&jit=&cg=&bdtype=0&oriquery=&objurl=http%3A%2F%2Fkfwimg.ikafan.com%2Fupload%2F83%2Fcc%2F83cc289953ec99b79986ea9346e7abc4.png&fromurl=ippr\_z2C%24qAzdH3FAzdH3Fooo\_z%26e3Bhwuwg\_z%26e3BvgAzdH3Fj17AzdH3Fcd88abcm\_z%26e3Bip4s&gsm=3&rpstart=0&rpnum=0&islist=&querylist=&force=undefined](https://image.baidu.com/search/detail?ct=503316480&z=0&ipn=d&word=%E7%AD%9B%E5%AD%90%201%E5%88%B06&step_word=&hs=0&pn=2&spn=0&di=17930&pi=0&rn=1&tn=baiduimagedetail&is=0%2C0&istype=0&ie=utf-8&oe=utf-8&in=&cl=2&lm=-1&st=undefined&cs=4060570382%2C451230636&os=1234429836%2C3104516760&simid=3509062187%2C720304968&adpicid=0&lpn=0&ln=1260&fr=&fmq=1587502126523_R&fm=&ic=undefined&s=undefined&hd=undefined&latest=undefined&copyright=undefined&se=&sme=&tab=0&width=undefined&height=undefined&face=undefined&ist=&jit=&cg=&bdtype=0&oriquery=&objurl=http%3A%2F%2Fkfwimg.ikafan.com%2Fupload%2F83%2Fcc%2F83cc289953ec99b79986ea9346e7abc4.png&fromurl=ippr_z2C%24qAzdH3FAzdH3Fooo_z%26e3Bhwuwg_z%26e3BvgAzdH3Fj17AzdH3Fcd88abcm_z%26e3Bip4s&gsm=3&rpstart=0&rpnum=0&islist=&querylist=&force=undefined)

Players images:

<https://www.pinterest.com/pin/408772103672738093/?nic_v1=1aRtLeQeT3VmAIg9%2Fph0c5DBHEiKr2G5xRWNej3BJNYgey8TNF4cLEb0bZKp%2BdAg1w>

# Data files

None

# Punch List used

## *To do:*

## *Done:*

* Find an image of Snake and Ladder online or using Photoshop - 4/10/2020
* Organize and start to code - 4/10/2020
* Icon and board game image - 4/10/2020
* Partial code was written - 4/12/2020
* finish up the design document - 4/12/2020
* start the actual programming on the Snake and Ladder - 4/11/2020
* Start programming - 4/11/2020
* Start GUI - 4/11/2020
* Start programming on the Client and Server - 4/11/2020
* Start game code - 4/11/2020
* Fix Exception error - 4/21/2020
* Need to Javadoc comment All the code - 4/21/2020
* Update design document - 4/21/2020
* Final Presentation – 4/30/2020
* Final Submission – 4/27/2020

# Unresolved Issues

We have a problem about the replacement image of the player move. If there is more than one player in the same position. Not sure how to appear both of them are in the same position because one player image will replace another player image.