UNO

CS:342 Project 5

Group No. 02

Contents

Project Overview

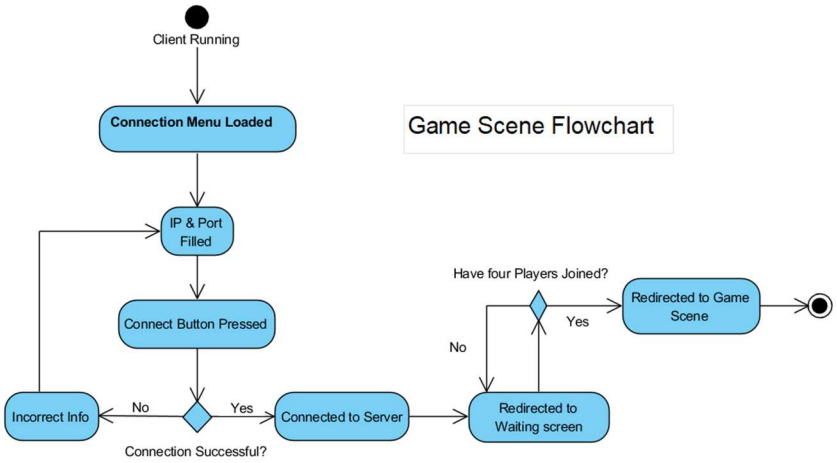
The game of Uno is fun, quick, and while it is easy to learn, it is difficult to master. We are implementing a game of four player game of UNO which can be run over a WIFI connection for local games when cards are not available.

How to Run

The server must be started first. The clients must be run on computers on the same WIFI as that of the server. Clients, when run, will show a connection screen. In this screen, the IP address of the Server on the WIFI, and the port number it is using must be entered. Upon entering these and pressing connect, the Client should be taken to the wait screen until three other players have joined.

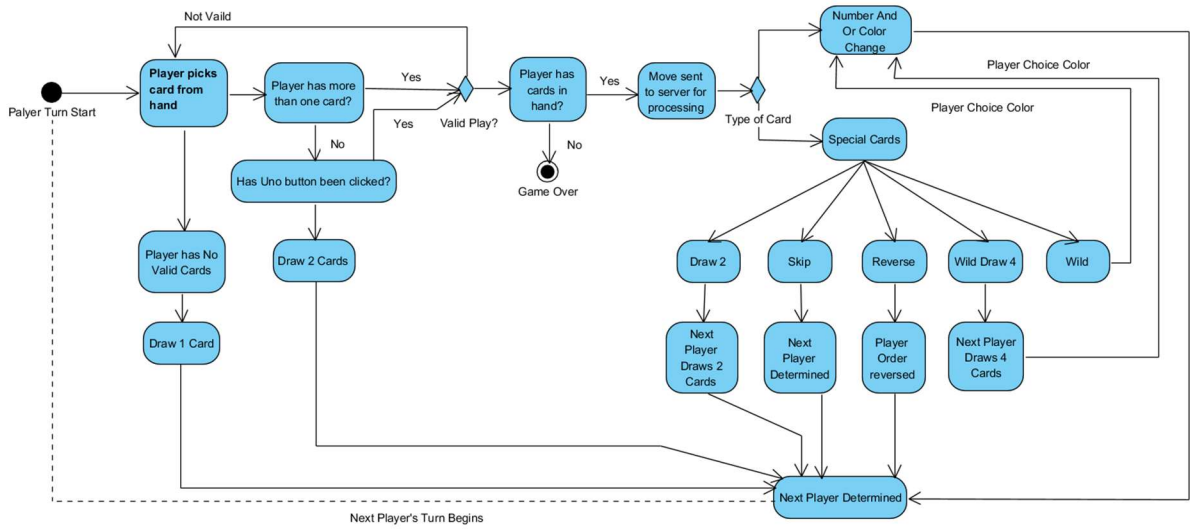
How to Play

Once the server is running, and your client connects to it from the Connection screen, the client will be redirected to the Wait screen, and will remain on that screen until three more players have joined. Once all four players are on the Wait screen, the game will begin.



Once gameplay has begun, players will be dealt a hand of and player 1 will begin their turn. Players may place down cards in their hands, which will have their effects as per the rules. If a player cannot play a card, one will be drawn, and their turn ended. If a player has one card left, they must click the UNO! Button before playing their card, if they do not, they must draw two cards and end their turn. The next player will be determined, and their turn will begin. Repeating the cycle.

This will continue until one player had an UNO, declares it via the button, and plays their card. Effectively ending the game.



Rules of Play

The rules of UNO are simple, and they must be adhered to. The server will check player moves for validity.

On a player's turn, they will check their hand to see if they have a card they can play given the current top of the discard pile. If they have a card of the same number, color, or special cards such as the skip, reverse, wild, draw 2, and wild draw four may be played given any situation. If the current player has only one card in their hand, they must declare UNO! Before playing it, failing to do so, will force them to draw two cards and skip their turn. If they have more cards, they may play one. If the player does not have any cards they can play, then they must draw one and end their turn

Wild cards can be played upon any other card. When doing so, the player must declare a color and play another card of that same color to reset the game with the new settings.
The Wild Draw 4 card allows the same play as the Wild card, but also forces the next player in line to draw 4 cards into their hand.
A Draw 2 card forces the next player to draw two cards before their turn begins.
The Reverse card toggles the order of play between clockwise and counterclockwise.
And Finally, the Skip card skips the next players turn entirely.

The game is won when a player has one card, declares UNO, can play that card and does so.

High Level Design

This section shows the important parts of the project, their purpose, design, and implementation.

Server

The server is implemented using Java sockets and threads. The server holds the following classes, all of which are elaborated on further below [HERE].

Server Classes
Server
ClientListener
Client
Game
Player
Deck
Card

The server and its clients communicate using Operation codes (Op-Codes from here onward) which consist of a string deliminated by commas which hold information telling the server or client which functions to call, and the arguments to those functions. The Op-Code table below shows the Op-Codes the client will send the server, which the server will need to handle.

Op-Code	Description	Called When
Client-Socket Connection	• Parameters : None • Sends unique Player ID • Sends Client List of other connected clients (if Any) • Notifies other clients that this client has connected • If number of connected clients is 4, then tells clients to start the game and send any necessary information (e.g. Player who starts, hand)	A client connects to the server.
Turn	• Parameters: {Play, Card} or Draw • If the client chooses Draw, draws a card from the deck and adds it to the clients hand • If the client chooses play, checks to see if the play is valid, if it is not, reject the play. • Determine which clients turn is next	A client chooses their move for the current round
UNO	• Parameters: None •	
Hand Request	• Parameters: Hand	Called by client to let server know it is ready to receive a dealt hand

Client

The Client is implemented using JavaFX. We chose this because of our familiarity to it. The client holds the following classes, all of which are elaborated on further below [HERE].

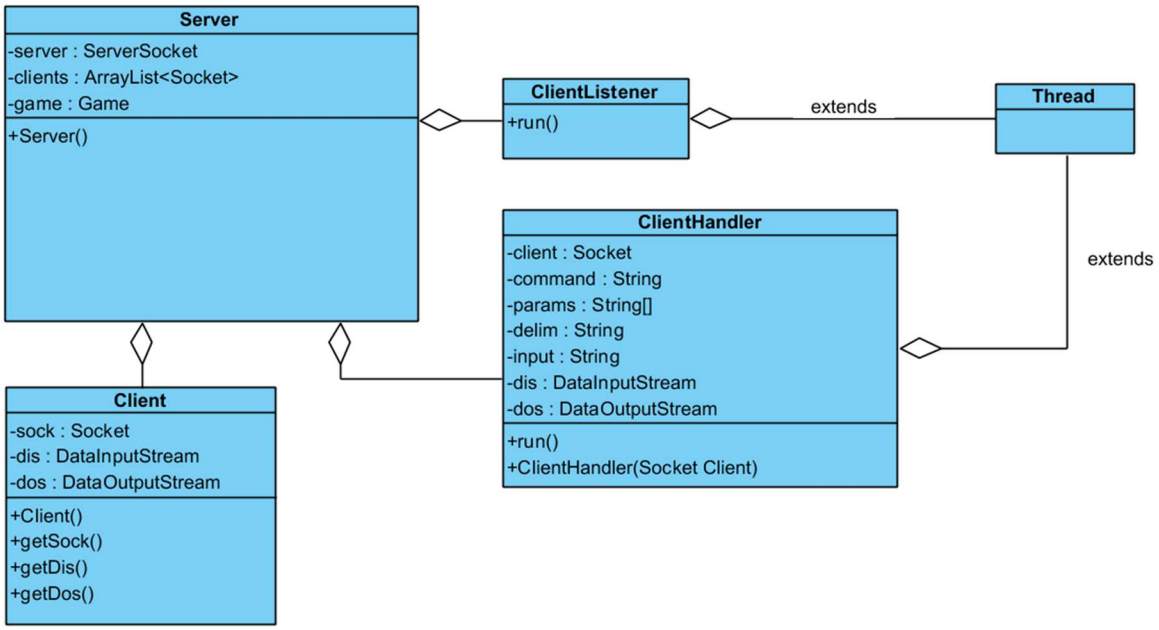
Client Classes
CLient
Card
serverComm
Main

The client, much like the server communicates using Op-Codes, Below all of the Op-Codes the server will call for the client to handle are listed and described

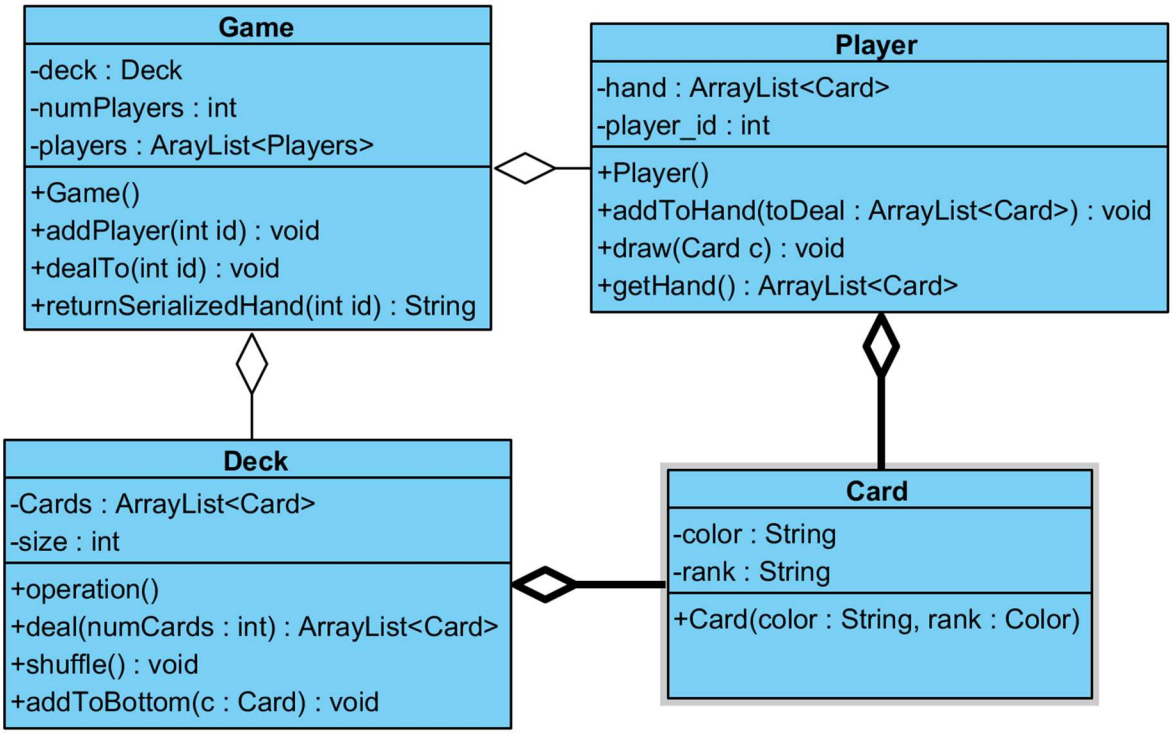
Op-Code	Description	Called When
Client socket Connection	• Prepares connections	
Player Joined	• Parameters : Player_ID • Keep information on this player for game information later on.	A new player has joined the server.
Message	• Parameters: Message • Receives message and displays it on client.	Server has s message for the client to post such as, “Player one wins, or Player 2 has disconnected”
Start Game	• Tells server to switch to game screen	Four players have connected to the server and are ready to play
Your Turn	• Parameters: Player_ID • Tells client it is their turn to play. • Client toggles button interactivity	Clients turn in the round has come
Game Over	• Tells clients the game has ended	The game ends
Send Hand	• Parameters : (param: [(yellow,9);(blue,2); (red,reverse)] etc..) • The parameters of this function are deliminated by “;” to be strings, and “,” within those strings give card information.	Server needs to send a new hand to the client
All Player Cards	• Number of cards each player has	

Low Level Design

Server

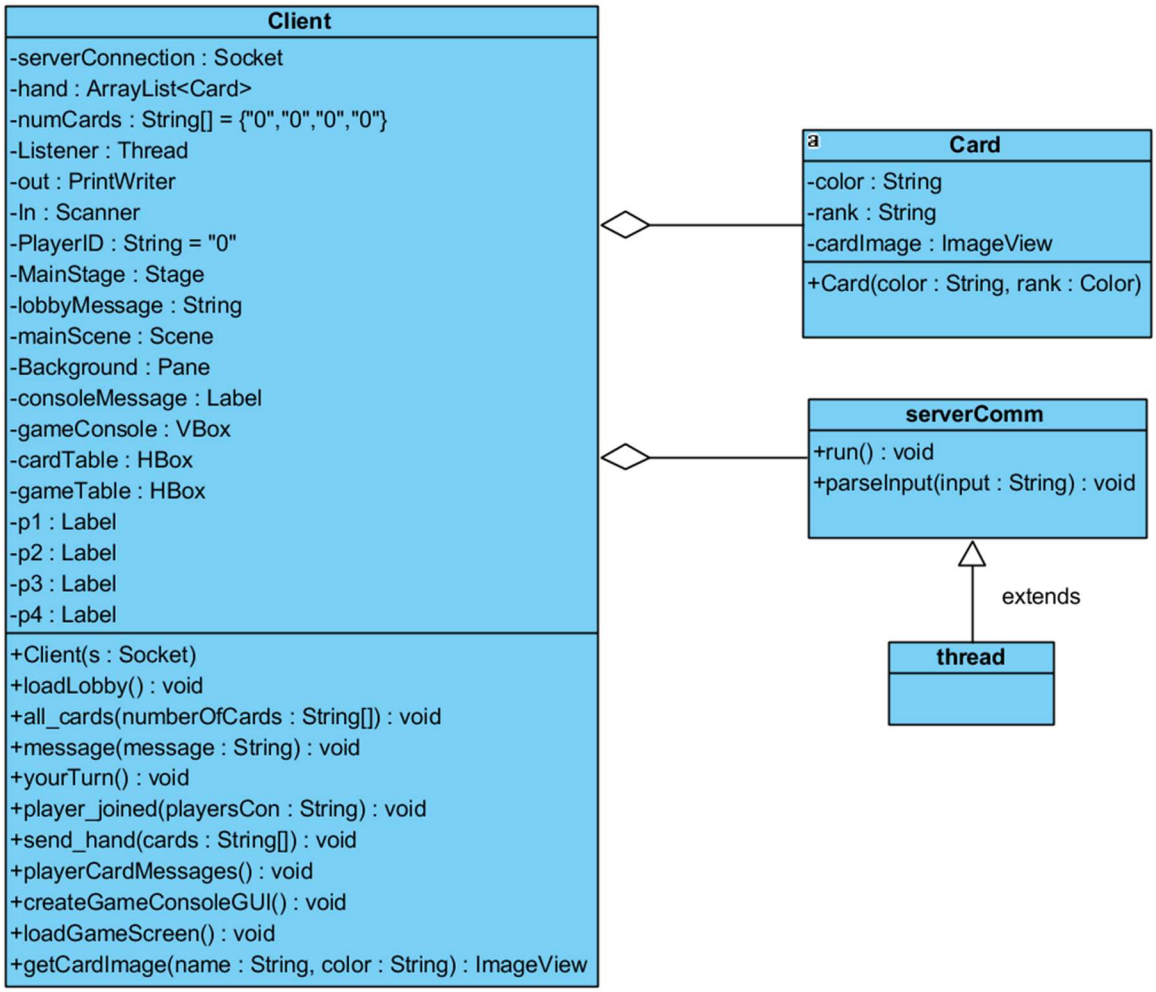


Game



Client

Client.java



Main.java