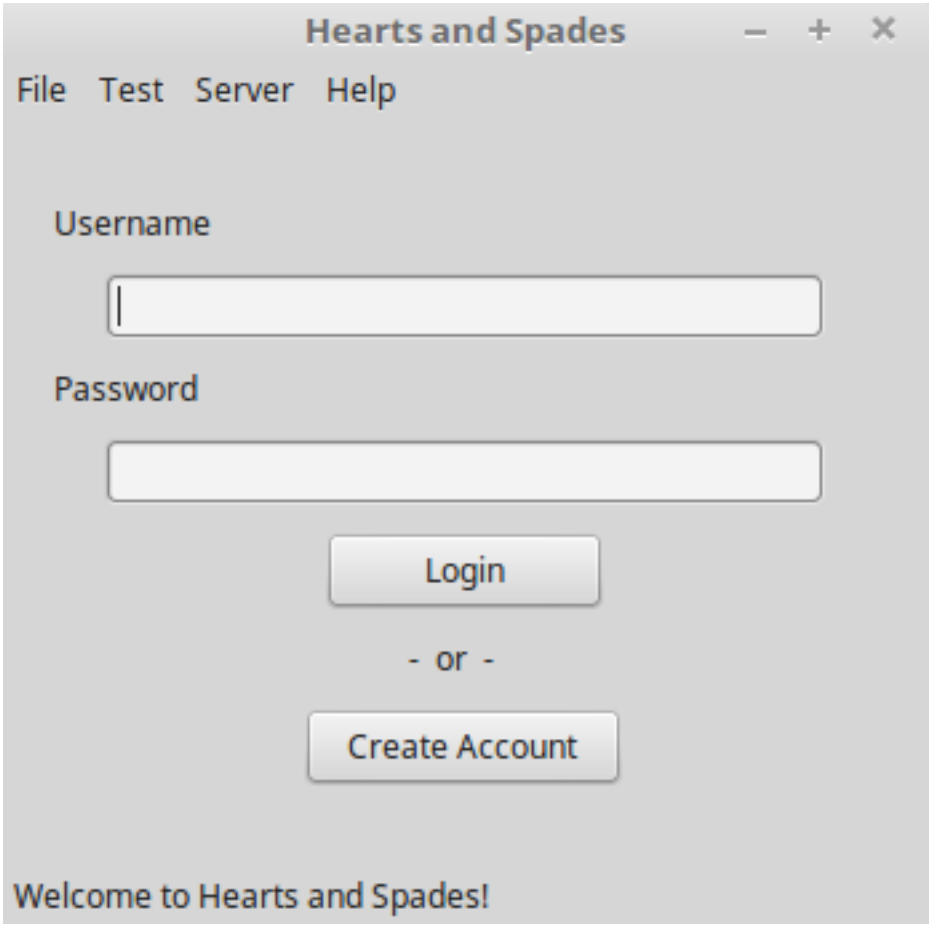
Software Design Description

Brandon Smith, Nieka Gutenberger, Joseph Coppin, Ryan Frazier, Trevor Jewkes

November 12, 2016

1 Software Design

1.1 Login Screen

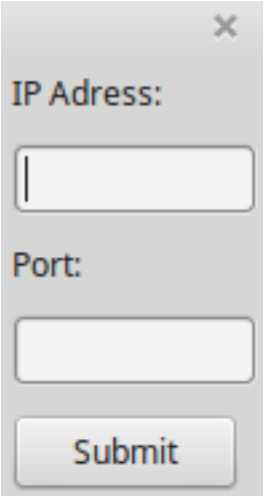


This screen is the first screen the user will see. It has a text box for the user to enter a user name and password. It also has two buttons a login which sends the username and password to the server, and brings up the lobby screen; and a Create Account button which takes the user to screen where they can create their account prior to playing games.

The account creation has text fields for all the information needed to create an account including Name, Username, Password and Verify Password, Finally the page includes a Create Account button which sends all information to the server so it can create the account.

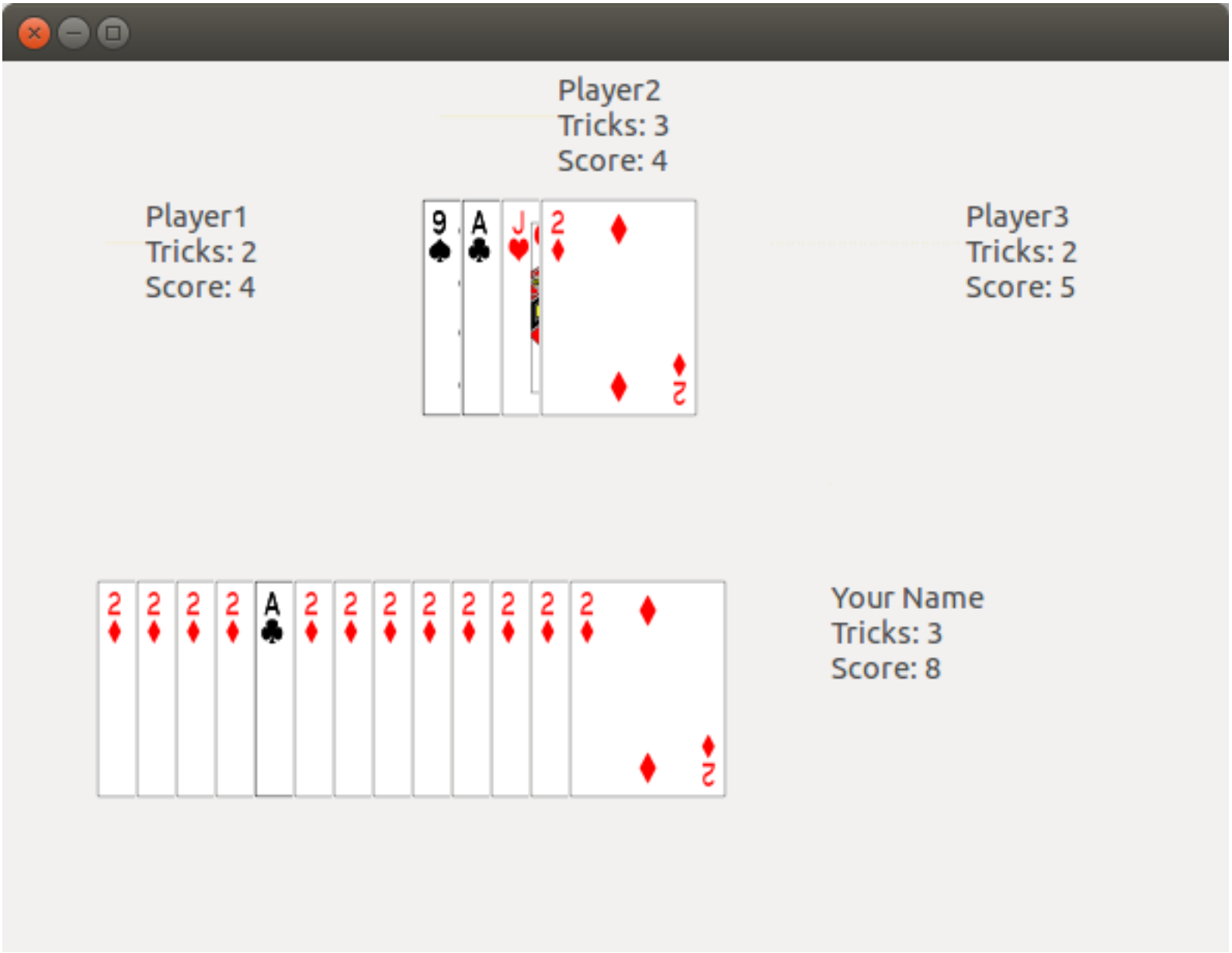
1.2 Main Menu

This screen is the main menu for the Game, it is divided in half with one half related to the game of Hearts, and the other half for the game of Spades. Each half includes three buttons, Join Private, Join Public and Create New. The Join Private brings up a screen which allows the user enter the name of the game they want to join. The Join Public will tell the server to assign the user to the first available public game, if no game is available the server will create a new game with the user and three AI players. Finally the Create New button will bring up a screen which asks for the users preferences to create a new game.



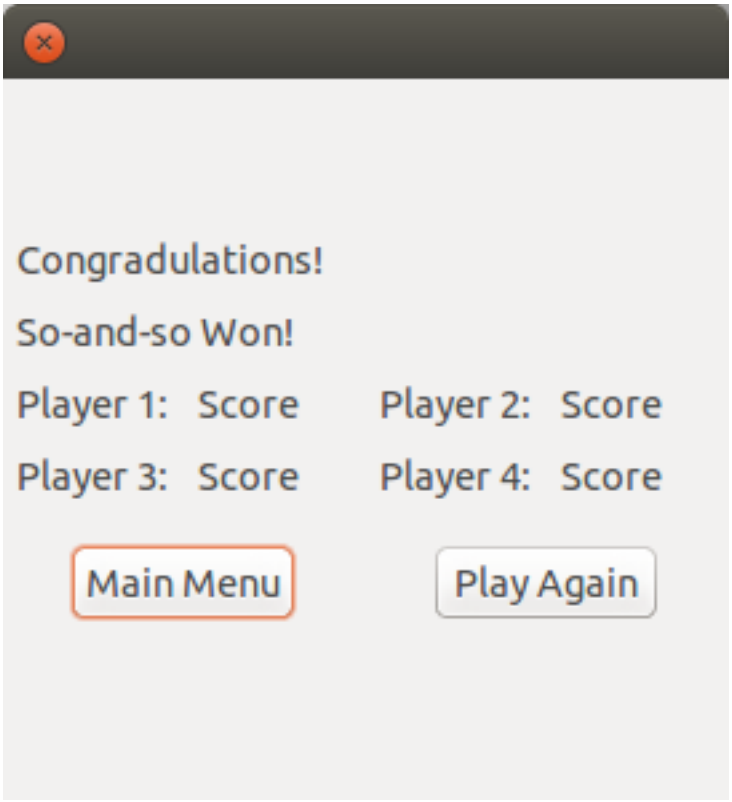
When connecting to the server, a menu option is available to specify the IP Address and Port number for the server.

1.3 Game Play



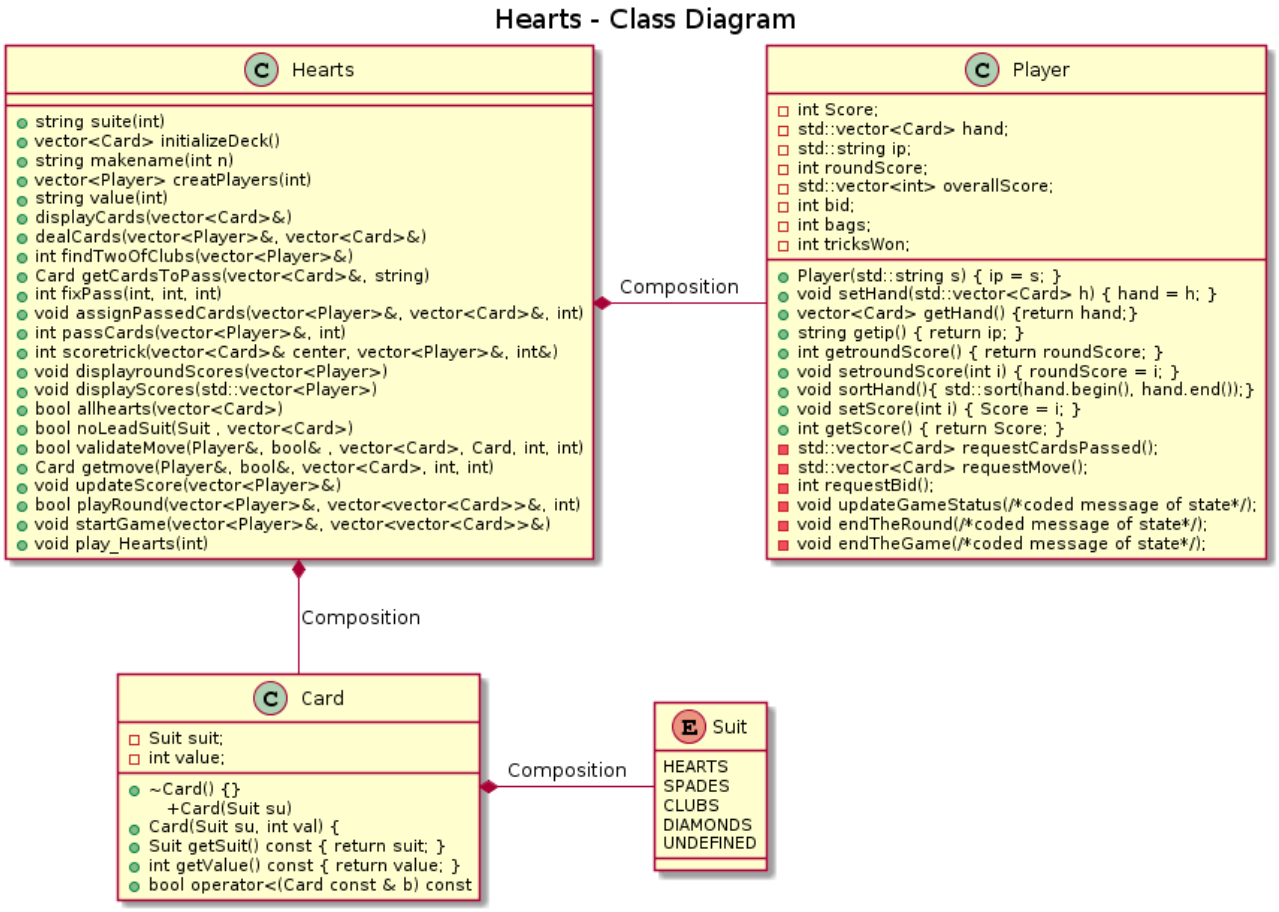
This screen will be the view that will be use for play of the game. Since Spades and Hearts have the same basic set-up we can use the same view for both. It shows the players hand along with the scores of the other players. Placing bids and passing cards fit within the gameplay screen.

1.4 End-of-Game



This the End-of-Game screen. It will display the winner of the game along with the final scores for the game.

There are two buttons, "Play Again" and "Main Menu". The "Play Again" button is used to play again with the same players. The "Main Menu" button will return the user to the main menu.

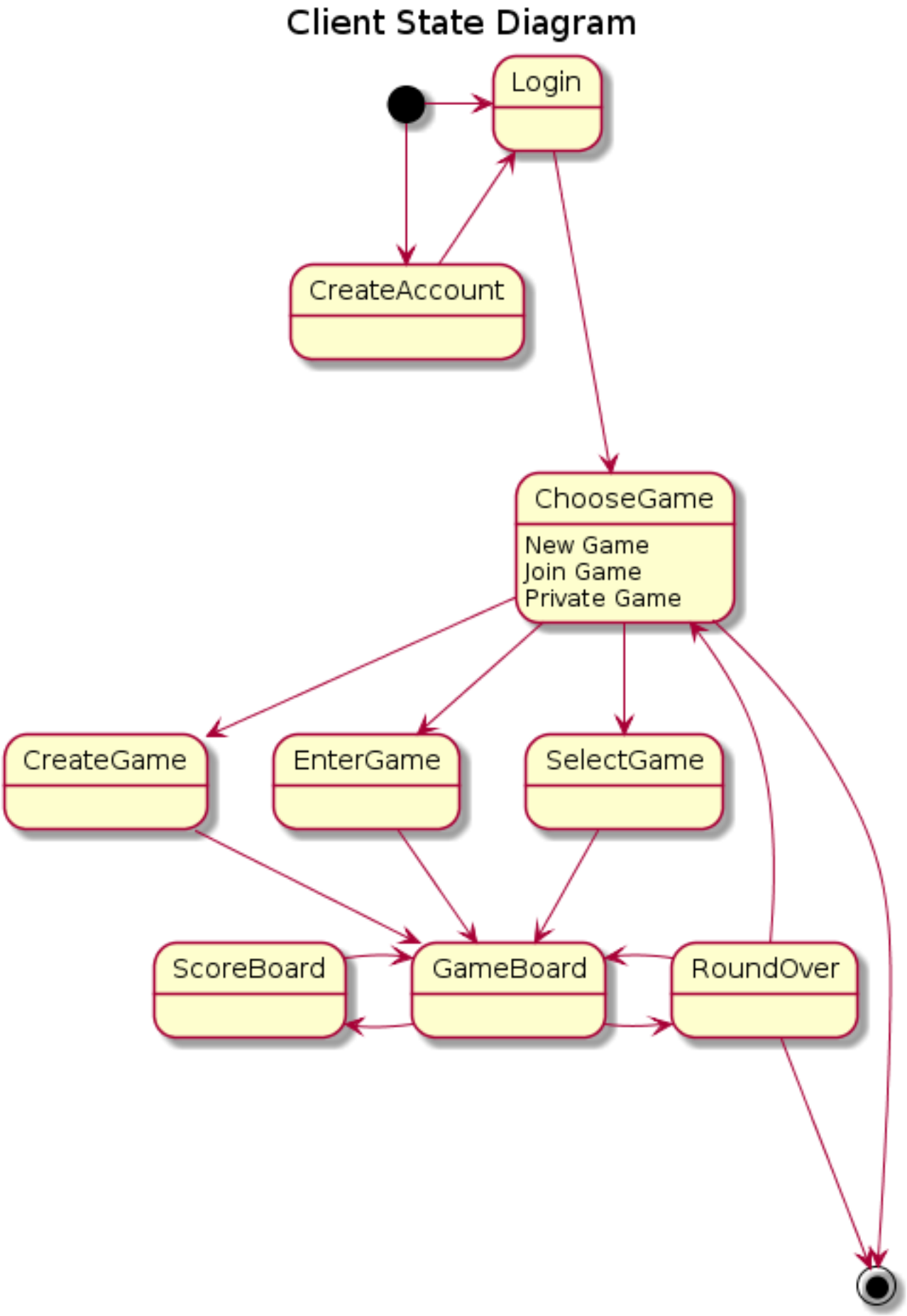


2 Server Low Level Design

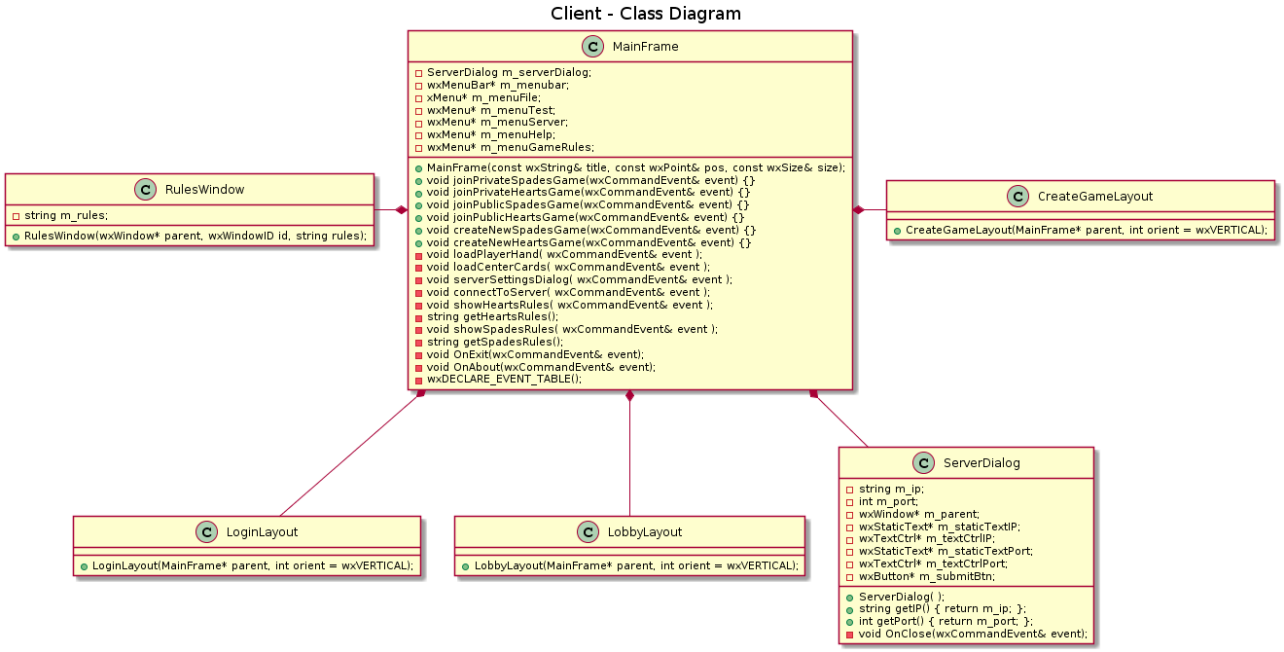
2.1 Hearts Class

- string suite(int i)
  - Converts enum of ints to string of card suit.
- vector ⟨Card⟩ initializeDeck()
  - Creates deck of cards taken from card class.
- string makename(int n)
  - Creates a player name.
- vector⟨Player⟩ creatPlayers(int p)
  - Creates a vector of Players to play the game.
- void displayCards(vector⟨Card⟩ & hand)
  - Displays the deck for screen purposes.
- void dealCards(vector⟨Player⟩ & players, vector⟨Card⟩ & Deck)
  - Deals cards to players.
- int findTwoOfClubs(vector⟨Player⟩ & p)
  - Looks through each hand to find the 2 of clubs to find starting player and hand.
- Card getCardsToPass(vector⟨Card⟩ & h, string p)
  - Gets and stores cards for passing at the beginning of each round.

- int fixPass(int r, int p, int c)
  - Ensures that cards are passed to the right players depending on the round.
- void assignPassedCards(vector⟨Player⟩ & p, vector⟨Card⟩ & h, int r)
  - Takes the passed cards and redistributes based on round.
- int passCards(vector⟨Player⟩ & p, int round)
  - Function for passing cards at beginging of round.
- int scoretrick(vector⟨Card⟩ & center, vector⟨Player⟩ & players, int& turn)
  - Holds the score for the current trick.
- void displayroundScores(vector⟨Player⟩ p)
  - Displays scores for the round.
- void displayScores(vector⟨Player⟩ p)
  - Display scores each turn.
- bool allhearts(vector⟨Card⟩ h)
  - Checks to see if a players hand is all hearts.
- string value(int i)
- bool noLeadSuit(Suit s, vector⟨Card⟩ h)
  - Compares hand against the lead suit
- bool validateMove(Player& p, bool& broken, vector⟨Card⟩ Center, Card move, int t, int i)
- Card getmove(Player& p, bool& b, vector⟨Card⟩ c, int t, int i)
- void updateScore(vector⟨Player⟩ & p)
  - Adds round score to Score.
- bool playRound(vector⟨Player⟩ & players, vector⟨vector⟨Card⟩ ⟩& history, int round)
- void startGame(vector⟨Player⟩ & players, vector⟨vector⟨Card⟩ ⟩& history)
  - Uses players and calls round until game is over
- void play_Hearts(int num)



State Diagram.png



3 Client Low-Level Design

3.1 MainFrame

This class is inherited publically from wxFrame.

- public:
  - MainFrame(const wxString& title, const wxPoint& pos, const wxSize& size);
    In the main frame, the main components of the software are held. Following are the functions included:
  - void joinPrivateSpadesGame(wxCommandEvent& event)
    This function allows a user to join a spades game that is closed to public use.
  - void joinPrivateHeartsGame(wxCommandEvent& event)
    This function allows a user to join a hearts game that is closed to public view.
  - void joinPublicSpadesGame(wxCommandEvent& event)
    This function will allow user to connect to first available spades game.
  - void joinPublicHeartsGame(wxCommandEvent& event)
    This function will allow user to connect to first available hearts game.
  - void createNewSpadesGame(wxCommandEvent& event)
    This function will create a new Spades Game.
  - void createNewHeartsGame(wxCommandEvent& event)
    This function will create a new Hearts Game.
- private:
  - ServerDialog m_serverDialog;
  - wxMenuBar* m_menubar;
  - xMenu* m_menuFile;
  - wxMenu* m_menuServer;
  - wxMenu* m_menuHelp;
  - wxMenu* m_menuGameRules;

- void loadPlayerHand(wxCommandEvent& event);
  Allows for players hand to be loaded on screen from server.
- void loadCenterCards(wxCommandEvent& event);
  Allows for cards to be placed in middle of screen after play.
- void serverSettingsDialog(wxCommandEvent& event);
- void connectToServer(wxCommandEvent& event);
  Allows player to connect to server to play game of choice.
- void showHeartsRules(wxCommandEvent& event);
  Allows user to get Hearts Rules.
- std::string getHeartsRules();
- void showSpadesRules(wxCommandEvent& event);
  Allows user to get Spades Rules.
- std::string getSpadesRules();
- void OnExit(wxCommandEvent& event);
  Allows user to exit program.
- void OnAbout(wxCommandEvent& event);
  Allows user to see details of program.
- wxDECLARE_EVENT_TABLE();

3.2 ServerDialog

This class is inherited publically from wxDialog.

- public:
  - ServerDialog(wxWindow* parent, wxWindowID id = wxID_ANY, const wxString& title = wxEmptyString, const wxPoint& pos = wxDefaultPosition, const wxSize& size = wxDefaultSize, long style = wxDEFAULT_DIALOG_STYLE);
  - std::string getIP() return m_ip; ;
  - int getPort() return m_port; ;
- private:
- protected:
  - std::string m_ip;
  - int m_port;
  - wxWindow* m_parent;
  - wxStaticText* m_staticTextIP;
  - wxTextCtrl* m_textCtrlIP;
  - wxStaticText* m_staticTextPort;
  - wxTextCtrl* m_textCtrlPort;
  - wxButton* m_submitBtn;
  - void OnClose(wxCommandEvent& event);

3.3 RulesWindow

This class is inherited publically from wxScrolledWindow

- public:
  - RulesWindow(wxWindow* parent, wxWindowID id, std::string rules);
    This allows user to read and learn rules for either Hearts or Spades.
- private:
  - std::string m_rules;

3.4 LoginLayout

This class is inherited publically from wxBoxSizer

- public:
  - LoginLayout(MainFrame* parent, int orient = wxVERTICAL);
    This function will bring up a screen where user may login to play game.

3.5 LobbyLayout

This class is inherited publically from wxBoxSizer

- public:
  - LobbyLayout(MainFrame* parent, int orient = wxVERTICAL);
    This function allows the user to access the game lobby.

3.6 CreateGameLayout

This class is inherited publically from wxBoxSizer

- public:
  - CreateGameLayout(MainFrame* parent, int orient = wxVERTICAL);
    When called, this function sets up a game layout for game play.