Pokerface - v2024.06.10

Team 8: PretendGINEERS

Mharlo Borromeo

Jack Lu

Calvin Nguyen

Mervin Nguyen

Peter Nguyen

Derek Tang

Producer: PretendGINEERS Studios

Affiliation: PretendGINEERS Software Inc.

University of California, Irvine : EECS 22L S24

Table of Contents

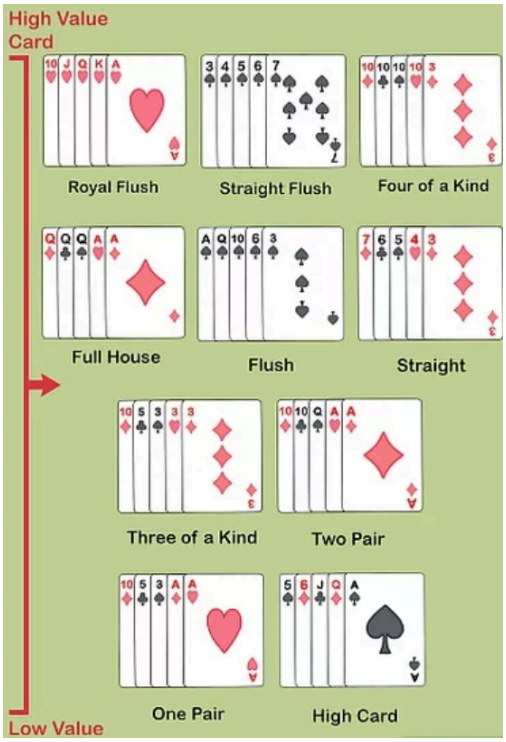
© 2004 - 2024 PretendGINEERS. All Rights Reserved.

Glossary

Poker Hand - consists of 5 cards in total per player, with each of them having a different ranking from low to high.

- **Royal Flush:**
  A, K, Q, J, 10, all of the same suit, highest ranking.
- **Straight Flush:**
  Five consecutive cards of the same suit.
- **Four of a Kind:**
  Four cards of the same rank and one side card or ‘kicker’.
- **Full House:**
  Three cards of one rank and two cards of another rank.
- **Flush:**
  Five cards of the same suit, not in sequence.
- **Straight:**
  Five consecutive cards of different suits.
- **Three of a Kind:**
  Three cards of the same rank and two unrelated side cards.
- **Two Pair:**
  Two cards of one rank, two cards of another rank, and one side card.
- **One Pair:**
  Two cards of the same rank and three unrelated side cards.
- **High Card:**
  Any hand that does not qualify under the categories listed, the highest card plays (this is the lowest ranking).

© 2004 - 2024 PretendGINEERS. All Rights Reserved.



- **Preflop:** Before any community cards are dealt, players receive two private cards (hole cards) and place initial bets through blinds or antes.
- **Flop:** Three community cards are dealt face up, allowing players to begin forming poker hands; betting follows, starting from the player left of the dealer.
- **Turn:** A fourth community card is added to the community board, providing more chances to build a hand, followed by another round of betting.
- **River:** The fifth and final community card is dealt, completing the possible hand combinations, with a final round of betting ensuing.
- Showdown: Remaining players reveal their hole cards, and the best five-card hand using any combination of their hole and community cards wins the pot.
- **Call:** To match the current bet made by another player to stay in the hand.
- All In: To bet all remaining chips. This move is made when a player puts their last chips into the pot.
- **Action:** Refers to a player's turn to act during a hand.
- **Backdoor:** Hitting needed cards on both the turn and the river to make a drawing hand.
- **Bad Beat:** Losing a hand where you were a strong favorite to win.
- **Bet:** To wager chips into the pot based on the strength of your hand, or to bluff other players into folding their hands.

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

C / GTK Technical Terminology

API - ‘Application programming interface’, refers to the way components/modules of a computer program are connected together.
Array - A collection of similar data type items stored contiguously in memory locations. Arrays can have different dimensions to them (such as 2D arrays, which will be used in this program).
Compile - The transformation of source code into machine code that can be executed by a computer, which is performed by a special program called a compiler.
Enum - A special kind of data type defined by the user where constant integers (starting from 0) are assigned to names.
GUI - ‘Graphical user interface’, refers to how the program will appear to the user and the environment that the user will interact with and within.
Widget - Graphical objects that are event driven and are resizable and are used to make the GTK GUI interactive
Graphics Library - A set or collection of functions and tools that can be used to perform tasks related to graphics and visual user interfaces.
Pointer - A variable that holds the memory address of a different variable of similar data types.
Struct - A composite data type that defines a grouped set of variables under one name in a single, continuous block of memory.

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

1. Poker Client Software Architecture Overview

1.1 Main Data Types and Structures

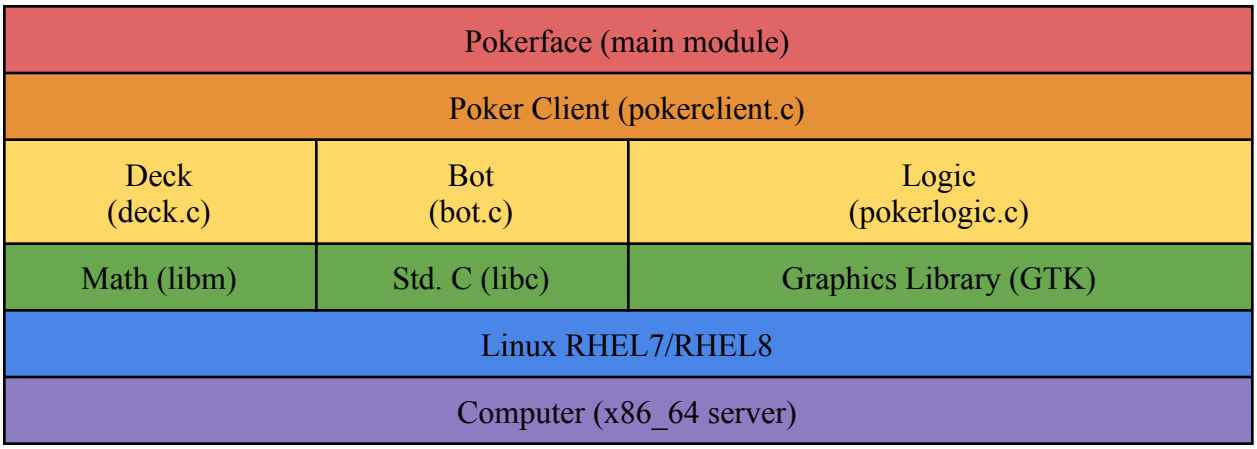
Data Types

- **CardSuit**
- **CardRank**
- **GameStage**
- **Role**
- **Actions**

Data Structures:

- **Card**
- **Deck**
- **Player**
- **GameState**

1.2 Major Software Components



© 2004 - 2024 PretendGINEERS. All Rights Reserved.

1.3 Module Interfaces

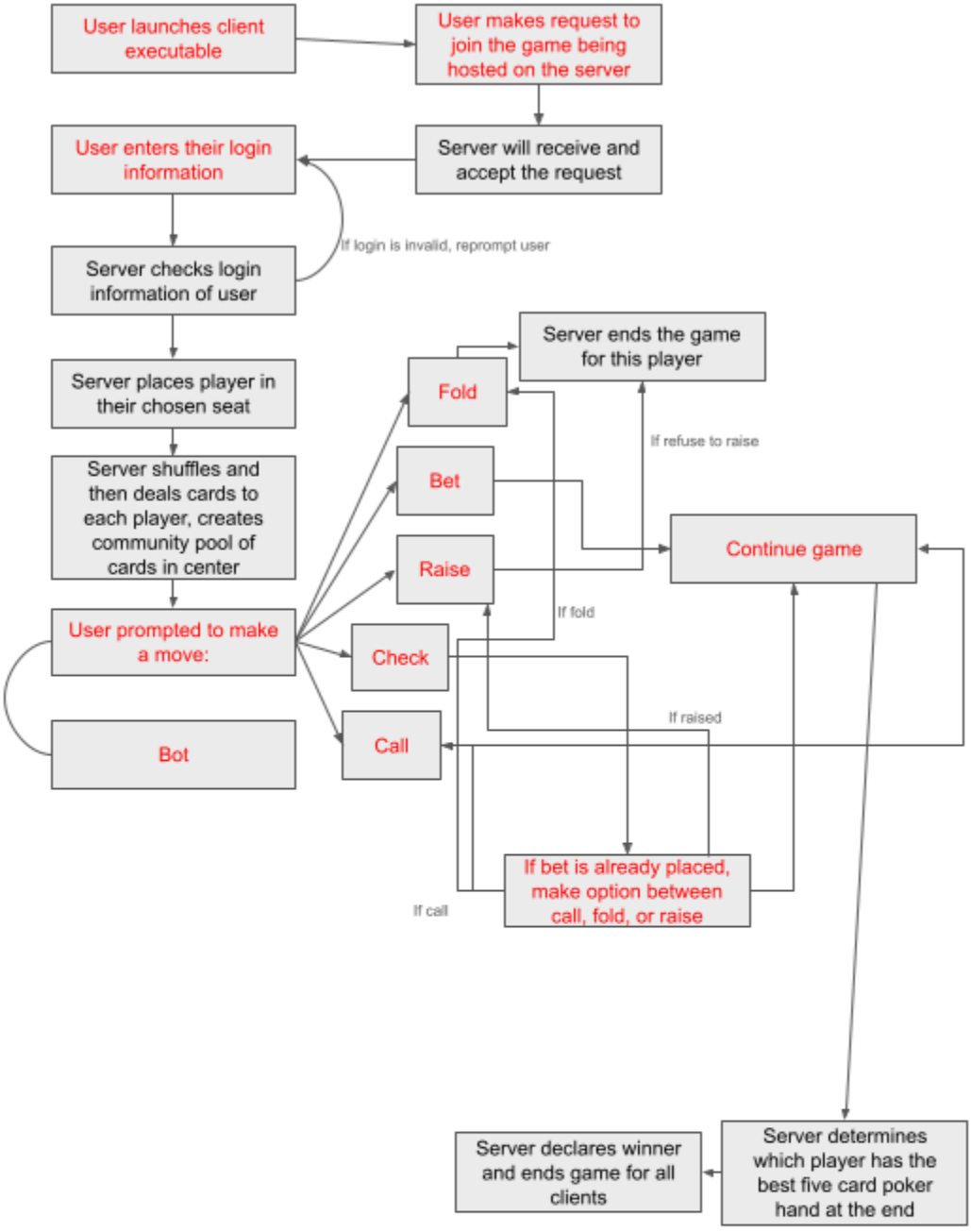
1. Pokerface (main module): The top-level module that integrates all the components of the poker software. It serves as the main application managing the overall game logic and user interface.
2. Poker Client (pokerclient.c): Manages the client-side operations, including the user interface and player interactions.
3. Pokerboard (pokerboard.c): Manages the state of the poker table, including the cards on the table, player positions, and the pot size. Handles the display and updates of the game board.
4. Logic (pokerlogic.c): Contains the core rules and logic of the poker game, managing the flow of the game, including dealing cards, betting rounds, and determining the winner.
5. GUI (gui.c): Creates an interactive GUI for both the client and server side. Takes in inputs from other modules and updates the GUI accordingly.
6. Bot (bot.c): A bot program that plays the game with the player and is able to make moves.

1.4 Overall Program Control Flow

1. Client program is started by the user in the main menu after launching the executable.
2. User chooses their seat, and logs in via a username and password.
3. Fill in the empty seat with a bot that plays with the players.
4. Server will deal cards to various players, after shuffling the deck and randomly selecting two cards for each player. The dealer will then deal five “community” cards in the center of the table that each player can use in conjunction with their own dealt cards to make their five card poker hand.
5. Players are able to choose from a variety of actions to perform, being ‘fold’, ‘check’, ‘bet’, ‘call’, or ‘raise’. Players are only able to check if no one else has made a bet yet. Once a player has bet, players that go afterwards must call, fold, or raise.
6. The server determined which of the players had the best five-card poker hand .
7. The game ends during the showdown, where the highest poker hand wins, unless all players except one have abandoned the game before the showdown round.

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

Figure 1: Diagram of program flow.



Control Flow Diagram with Red Highlighting for Client Interactions

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

1.5 Automated Client: Poker Bot

1. The poker bot should be able to interact with the poker game interface, such as reading the cards on the table and checking the players actions.
2. Evaluate the strength of its own hand and potential hands based on community cards.
3. Based on its own strength of its own hand, it makes decisions on what its next move is. (such as call or fold etc.)
4. If a new game starts, fill in all empty spots that are not chosen by the human players.

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

2. Poker Server Software Architecture Overview

2.1 Main Data Types and Structures

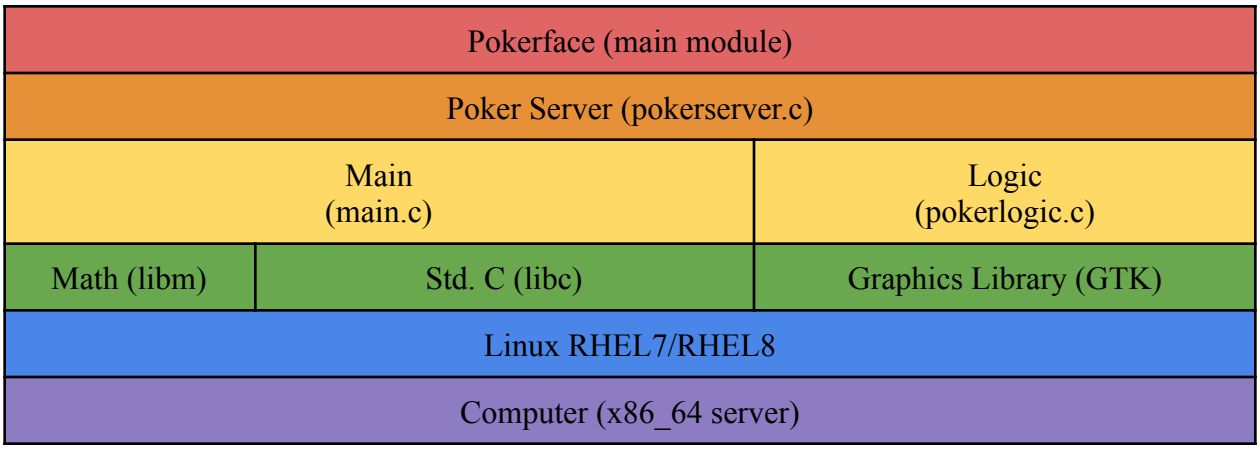
Data Types

- **CardSuit**
- **CardRank**
- **GameStage**
- **Role**
- **Actions**

Data Structures:

- **Card**
- **Deck**
- **Player**
- **GameState**

2.2 Major Software Components



2.3 Module Interfaces

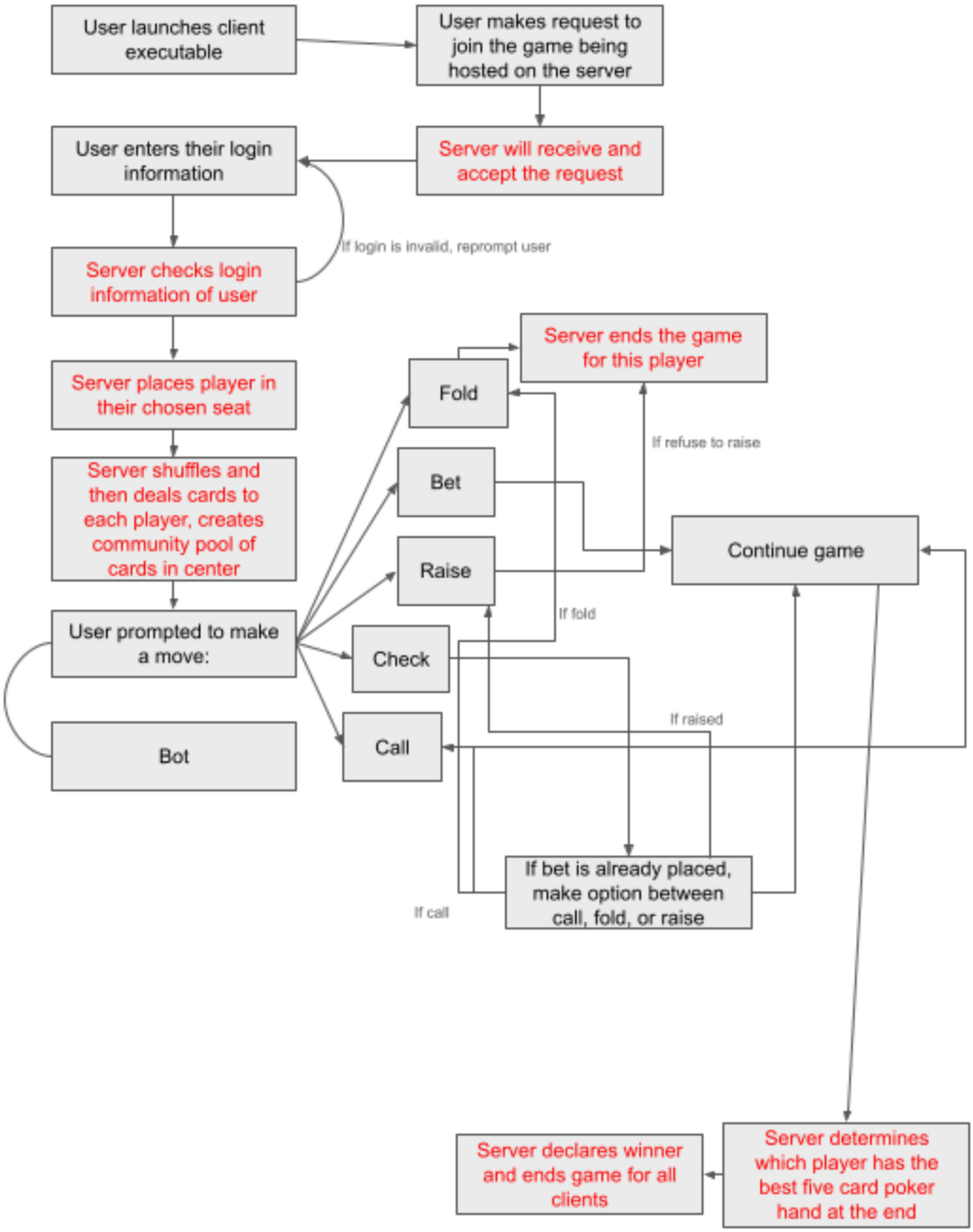
1. Logic (pokerlogic.c): Contains the core rules and logic of the poker game, managing the flow of the game, including dealing cards, betting rounds, and determining the winner.
2. Main (main.c): do things such as keep track of players, shuffle cards, check end game condition, will keep running for a certain number of rounds to create an entire poker game.
3. libm and libc C libraries: necessary in order to perform basic arithmetic functions as well as commands like printf() which are functions that are not included in the very basic C packages.

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

2.4 Overall Program Control Flow

1. Program runs and starts the server.
2. Server will wait for the user to send a request to join the server via a TCP/IP communication via sockets.
3. Once the server approves the user to connect to the server, begin to keep track of the user and check if their username and password are correct.
4. If the username/password is incorrect, ask the user to re-enter them, else allow the user to choose a seat and wait for the game to begin.
5. Fill in any open seat with a bot and begin the game.
6. The server will shuffle the deck and randomly select two cards for each player. The dealer will then deal five community cards in the center of the table that each player can use in conjunction with their own dealt cards to make their five card poker hand.
7. Continue the game with input from users (call, fold raise etc) via continue to keep track of each player's points and the current game state.
8. End the game and award the winning player with points based on whoever has the best five card poker hand.

© 2004 - 2024 PretendGINEERS. All Rights Reserved.



Control Flow Diagram with Red Highlighting for Server Interaction

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

3. Installation

3.1 System Requirements and Compatibility

Hardware	Minimum Specification
CPU	Any x86_64 processor with clock speed of at least 1GHz
RAM	At least 1 GB
Disk Space	At least 500 MB
Operating System	Linux (RHEL7/RHEL8)

Software

Xming downloaded to be able to display the poker GUI in separate window

3.2 Setup and Configuration

- The installation of the program requires some knowledge of navigating Linux via terminal interface.
- Prerequisite of having GTK 2.0 library installed.

3.3 Building, Compilation, Installation

- Installation (basic user, no building or compiling necessary)

- Download Poker.tar.gz compressed folder.
- Navigate to the directory where the Poker.tar.gz folder is located after downloading(for example, 'cd Downloads'), and then run command (in terminal) ‘gtar xvzf Poker.tar.gz’ in order to unpack/unzip the file.
- Navigate within the newly unzipped folder in the terminal ('cd poker), and then navigate to the ‘bin’ folder in order to run the program executable (‘./pokerserver [port number]’ to start the server, ‘./pokerclient [hostname] [port number]’ to start each client, use ports 10080-10089).

- Advanced Installation (advanced user, requires different folders in order to build and compile properly).

- Download Poker_src.tar.gz compressed folder.

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

- Navigate to the directory where the Poker_src.tar.gz folder is located after downloading(for example, 'cd Downloads'), and then run command (in terminal) ‘gtar xvzf Poker_src.tar.gz’.
- Navigate within the newly unzipped folder in the terminal ('cd poker), and then run the command ‘make all’ in order to build and compile the Poker executable within the ‘bin’ folder.
- Optionally, utilize commands ‘make clean’ in order to remove all object files and executable files created by the ‘make/make all’ command.
- Optionally, command ‘make test’ can also be used to create unit test executables for all major modules of the program, which will be placed within the ‘bin’ folder (commands ‘make test-gui’, ‘make test-comm’, ‘make test-server’, and ‘make test-client’ are also used to create individual unit test executables and run them).
- Optionally, use command ‘make gtar’ in order to create a tar of the source code package (compress the source code package) that will execute the macro ‘gtar cvzf Poker_src.tar.gz’ along with all bundled files.
- Navigate within the ‘bin’ folder and then run the executable by using the command ‘./pokerserver [port number]’ to start the server, ‘./pokerclient [hostname] [port number]’ to start each client, use ports 10080-10089.

- Uninstallation (for both basic and advanced users)

- Navigate to the folder where Poker.tar.gz was uncompressed within terminal ('cd Downloads', then 'cd poker), taking extra care that there are no other contents in the folder besides the files that were unzipped from the Chess.tar.gz file, then use 'rm -f *' to remove all files within the downloaded and unzipped folder without needing to authorize the deletion of each file at a time.
- Additionally, to remove the poker directory while starting inside of it, first cd to the directory right above (‘cd ..’), and then use the command ‘rm -rf poker/’ which will remove the poker directory which was created when unpacking the compressed folder.

- See installation instructions for more information.

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

4. Documentation of packages, modules, interfaces

4.1. Detailed description of data structures

typedef struct Card

Using typedef and struct to create “Card”. Rank is an integer representing rank of the card (2-10, jack, queen, king, & ace). Suit is an integer that represents the card (Hearts, Diamonds, Clubs, & Spades).

```
typedef struct {
    int rank;
    int suit;
}Card;
```

typedef struct Hand

Hand represents the data structure that will define the set of cards each player needs to make based on their own dealt cards, as well as the cards on the table. Each hand in poker needs to be 5 cards in total, and will be ranked amongst each player to determine which hand is the best.

```
typedef struct {
    struct Card hand[5];
}Hand;
```

typedef struct Deck

Using typedef + struct for Deck Structure. This represents the 52 cards in the playing deck. Cards is an array of “Card” structures. Top keeps track of the top card so it can be dealt.

```
typedef struct {
    struct Card cards[NUM_CARDS];
    int top;
}Deck;
```

typedef struct Player

Struct Player keeps track of the player - array of “Card” structures for each player

```
typedef struct {
    struct Card hand[CARDS_PER_PLAYER];
} Player;
```

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

GUI (gui.h)

In gui.h many data structures from the GTK 2.0 library are used in order to build and display the interactive GUI's used in the program. Some of these include GtkWidget which are used to create predefined GUI elements (widgets) and GdkPixbuf which is used to change the window's icon.

```
// GTK Variables
GtkWidget *window;                  // Main window
GtkWidget *vbox_main_menu;          // Allignment
GtkWidget *label_main_menu;         // Main Menu Label
GtkWidget *tablealign;              // Alignment for Username/Password table
GtkWidget *table_main_menu;         // Table Allignment
GtkWidget *label_username;          // Username Label
GtkWidget *label_password;          // Password Label
GtkWidget *entry_username;          // Entry textbox for username
GtkWidget *entry_password;          // Entry textbox for password
GtkWidget *comboalign;              // Alignment for dropdown selection
GtkWidget *combo_seat_dropdown;     // Dropdown text selection for seat number
GtkWidget *buttonalign;             // Alignment for the play button
GtkWidget *button_play;             // Play button
GdkPixbuf *pokerIcon;
```

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

4.2: Detailed description of functions and parameters.

- **`void initializeDeck(Deck *deck);`**
  - The primary purpose of initializeDeck is to initialize 52 cards to represent a deck, setting up rank and suits.
  - Parameters: initializeDeck sets deck as a pointer to Deck structure to be initialized

- **`void createHands(Hand *hand);`**
  - The primary purpose of createHands is to generate hands based on the player's choice of two cards (hole cards) as well as the ones they choose in the set of community cards. These hands will be completed once the entirety of the 5 community cards have been revealed (the flop, turn, and river).
  - Parameters: The parameters of createHands will be a pointer to the hand of a player, which will already consist of 2 dealt cards, and the dealer choosing which is the winning hand.

- **`void shuffleDeck(Deck *deck);`**
  - The primary purpose of the shuffleDeck is to randomize the order of the cards in the cards in the deck. This is to randomize the card order for every game to be different.
  - Parameters: deck is a pointer to the Deck structure so that it can be shuffled.

- **`void dealCards(Deck *deck, Player *players);`**
  - dealCards deals a fixed number of cards (CARDS_PER_PLAYER) from the deck to each player.
  - Parameters: deck is a pointer to Deck structure which deals the cards. player is a pointer to Player structures to receive dealt cards.

- **`void analyzeHands(Hand *hand)`**
  - The primary purpose of the analyzeHands function is to determine which player has the best hand after each community card has been revealed, as well the hand of each player who did not fold.
  - Parameters: Since the function needs to determine the winning hand, the parameter will be a pointer to each player’s hand accordingly.

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

- **`void printCard(const Card *card);`**
    - The primary purpose of printCard is to print the rank and suit of the card throughout various points of the game, such as after a card is pulled.
    - Parameters: Takes in the parameter, card is a pointer to the Card structure so that it can be printed.

- **`void showHands(Player *players);`**
    - The primary purpose of showHands is to display the hands of all current players at the time the function is called.
    - Parameters: players is an array of Player structures whose hands will be displayed.

- **`int makeBet()`**
    - The primary purpose of makeBet is to have the first two players of the poker game make a bet before their cards are revealed
    - Since this function is just asking for user input, there is no need for anything within the parameters.

- **`int makeCall()`**
    - The primary purpose of makeCall is to act as a move that a player can make, which essentially is a bet that matches the previous player's bet.
    - Parameter: A potential parameter for this function would be a way to access the previous player's bet and copy it over, should our game include each player to have their own particular bet as part of the struct.

- **`int makeRaise()`**
    - The primary purpose of makeRaise is to act as a move that a player can make, which essentially is a bet that exceeds the previous player's bet in value.
    - Parameter: A potential parameter for this function would be a way to access the previous player's bet and make sure that the currently made bet is higher than the previous one

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

- **`void makeFold(Hand *hand)`**
    - The primary function of makeFold is to act as a move that a player can make, which essentially has the player surrender their cards and their involvement in the round.
    - Parameter: A parameter for makeFold would be the current hand of the player that wants to fold, so that they can be removed from the round.

- **`void makeCheck()`**
    - The primary function of makeCheck is to allow the player to make the move by essentially skipping on their turn of betting or raising. However, this only works if the previous players haven't made a bet yet.
    - Parameter: There is no need for a parameter for makeCheck for the time being.

- **`void ShutdownClicked(GtkWidget *Widget, gpointer Data)`**
    - The purpose of this function shutdown the server when the shutdown button has been pressed
    - The function is a void function, as it can shutdown the server without needing to return a value
    - Parameters: The parameters of the function are the widget for which an event has occurred through GTK's g_signal_connect function as well as any additional data needed which is passed through the g_signal_connect function.

- **`GtkWidget *CreateWindow_Client(int *argc, char **argv[])`**
    - The purpose of this function is to create and maintain the GUI window for the client until the user exits the GUI.
    - The function returns a pointer of type GtkWidget which can be used in pokerclient.c to determine if the window has been successfully created.
    - Parameters: The parameters of the function are any command line inputs namely argc and argv of the main function in pokerclient.c.

- **`void UpdateWindow_Client(void);`**
    - The purpose of this function is to update the client side GUI to display the current cards if any change has occurred to the poker game.
    - The function is a void function, as it can update the client side GUI without needing to return a value
    - Parameters: The parameters of the function is void, as it can update the server side GUI without needing to return a value

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

- **`GtkWidget *CreateWindow_Server(int *argc, char **argv[])`**
    - The purpose of this function is to create and maintain the GUI window for the server until the user exits the GUI.
    - The function returns a pointer of type GtkWidget which can be used in pokerserver.c to determine if the window has been successfully created.
    - Parameters: The parameters of the function are any command line inputs namely argc and argv of the main function in pokerserver.c.

- **`void UpdateWindow_Server(void);`**
    - The purpose of this function is to update the server side GUI to display the current cards if any change has occurred to the poker game.
    - The function is a void function, as it can update the server side GUI without needing to return a value
    - Parameters: The parameters of the function is void, as it can update the server side GUI without needing to return a value

4.3: Detailed description of communication protocols

Communication Input/Output Formats

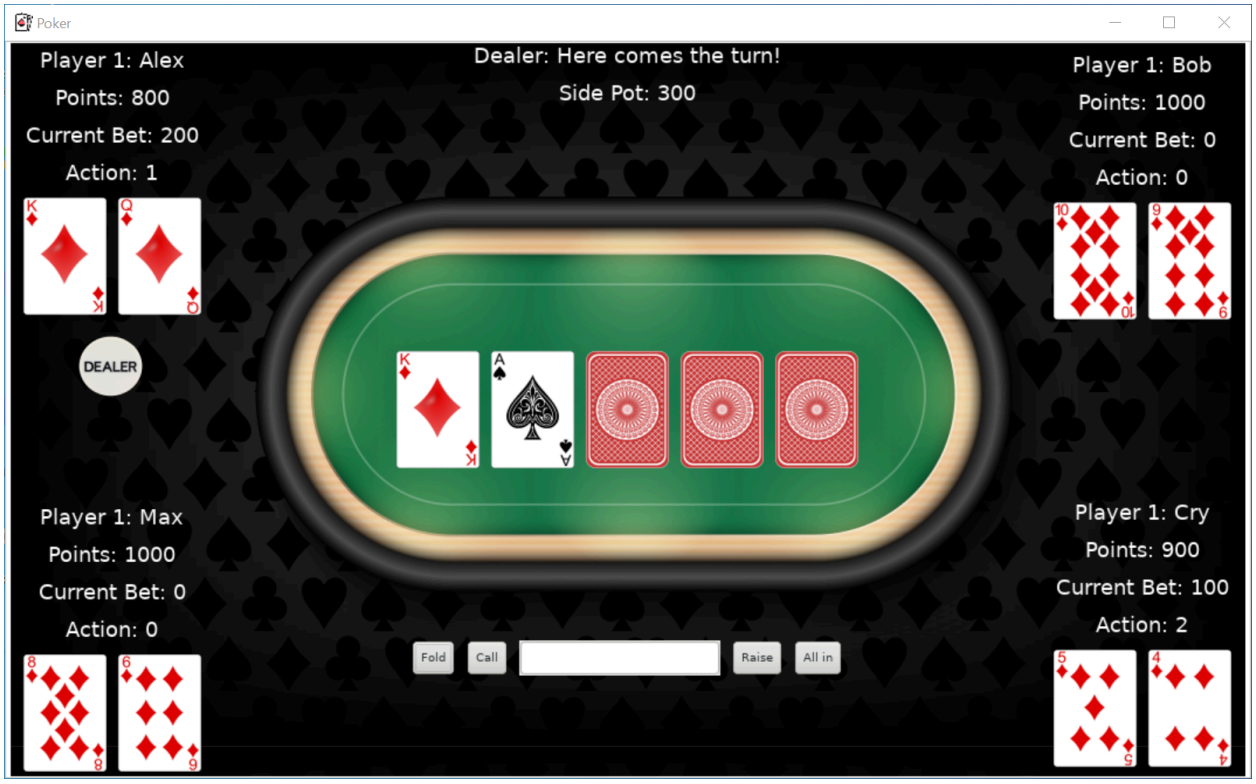
User Input:

Client Side:

The user will be prompted with this GUI at the start of the game:

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

After the game has begun:



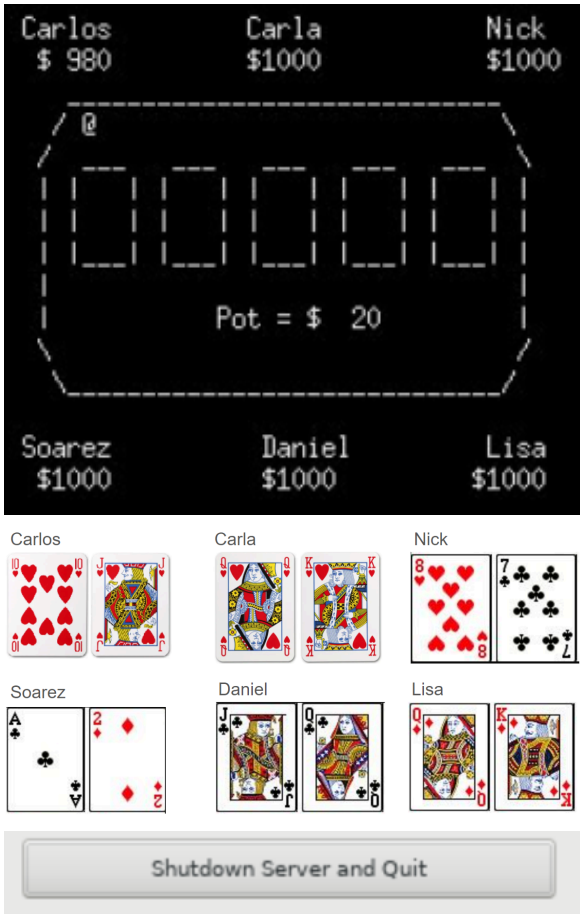
After starting the program the user will enter with their keyboard their username, password, and select their seat number with the dropdown menu. After entering all relevant info they will begin

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

the game and to play the game they will be able to use their mouse to click buttons in order to fold, call, raise, and to go all in.

Server Side:

The GUI after the server is connected:



The only input for the server side GUI is a button which will shutdown the server and quit the GUI.

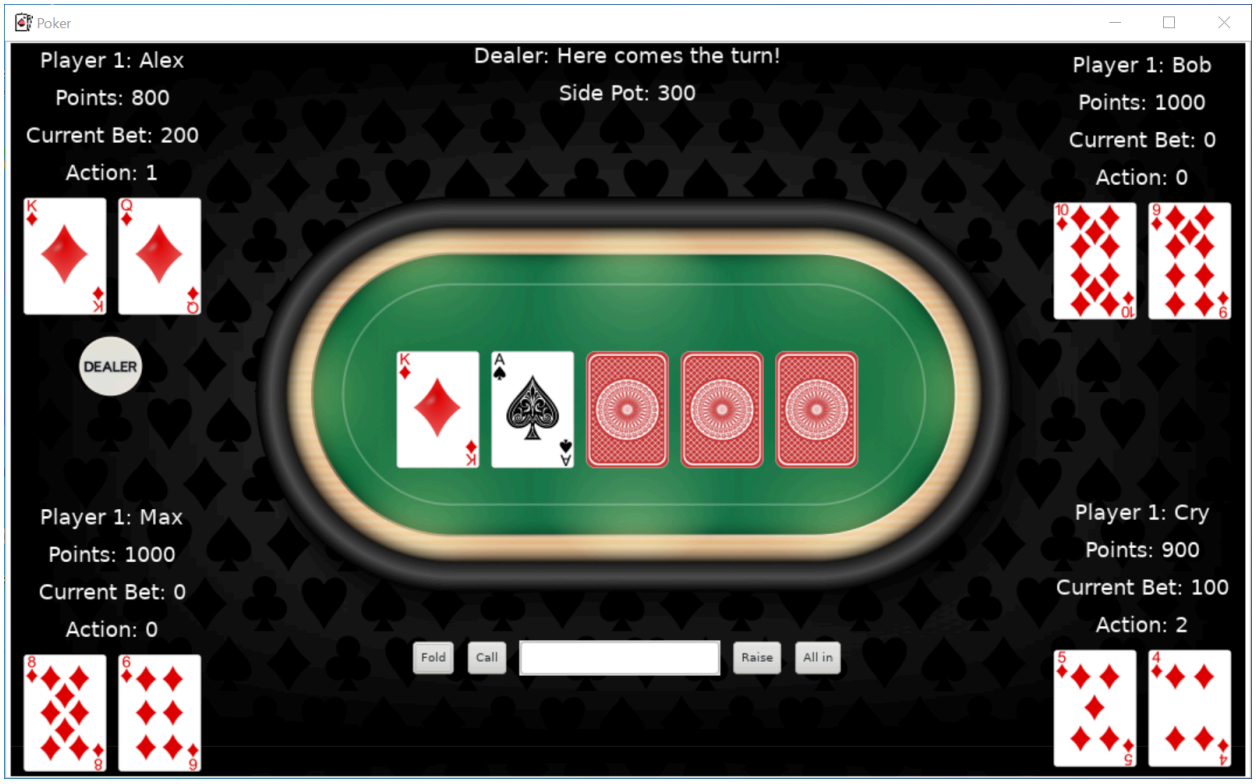
Output:

Client Side:

The user will be prompted with this GUI at the start of the game:

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

After the game has begun:

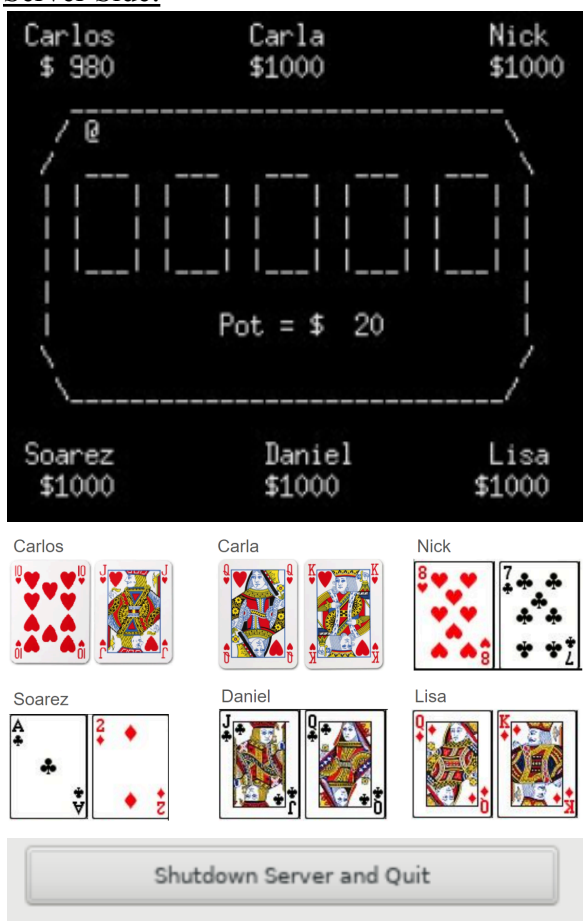


The client side GUI will display an interactive login screen at first. Then once the user is in the game it will display the points that other players currently have, any actions they may have

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

made, and the user's current hand. All outputs will be updated in real time on the client side GUI allowing the user to see and make decisions depending on their cards and the actions other players have made.

Server Side:



The server side GUI will display the current state of the board as well as the hand that each player has.

Communication Related Functions and Settings

Ports to be used: 10080-10089 (allows up to 10 clients to be connected, planned usage of 4 clients total including bot players)

gethostbyname(): Gets the host server address based on user input (will be prefilled in final release to only connect to UCI EECS servers, such as crystalcove@eecs.uci.edu)

connect(): Connects to designated server through designated port by establishing TCP connection

SendBuff/RecvBuff: Sending and receiving buffer, denotes the message buffer for sending a message to the server/client and receiving a response from the server/client.

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

5. Development plan and timeline

5.1: Partitioning of tasks

Weeks 1 and 2:

- User Manual
- Software Specification
- Begin GUI and server/client implementation
- Begin poker rule and poker bot logic implementation

Weeks 3 and 4:

- Continue GUI and server/client implementation
- Finish poker rule implementation
- Begin main function
- Begin unit testing for major modules

Week 5:

- Finish GUI and server/client implementation
- Finish poker bot integration
- Perform full bug testing and validation that all functionality is properly working
- Optional: add end scoreboard to see player rankings and all hands dealt to each player

5.2: Team member responsibilities

Mharlo Borromeo - user manual, software spec, structures.h (struct Card, struct Deck, struct Hand), gamelogic.c, (makeDeck), general debugging
Jack Lu - user manual, software spec, server/client code.
Calvin Nguyen - user manual/software spec, game functions (shuffleDeck, initGame, startGame, playerAction, displayPlayerCards, displayCommunityCards, game control flow
Mervin Nguyen - user manual, software spec, poker hand game logic(Royal flush, four of a kind, flush, straight, three of a kind, two pair, one pair, etc)
Peter Nguyen - user manual, software spec, documentation, server/client communication
Derek Tang - user manual, software spec, documentation, git setup, GUI (gui.c, gui.h, unit_test_gui.c)

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

Back Matter

Copyright Information

Permission to utilize, copy/make copies of, modify, and individually distribute the software packages for any purpose with or without fees is hereby granted.

THE SOFTWARE PACKAGE IS PROVIDED AS IS AND ALL AUTHORS DISCLAIM ALL WARRANTIES WITH REGARD TO THIS SOFTWARE PACKAGE INCLUDING ALL IMPLIED WARRANTIES. IN NO EVENT SHALL ANY AUTHORS BE LIABLE FOR ANY SPECIAL, DIRECT, OR INDIRECT DAMAGES WHATSOEVER RESULTING FROM LOSS OF DATA OR PROFITS WHETHER IN CONTRACTS OR IN NEGLIGENCE ARISING OUT OF OR IN CONNECTION WITH THE USAGE OR PERFORMANCE OF THIS SOFTWARE PACKAGE.

Index

© 2004 - 2024 PretendGINEERS. All Rights Reserved.

References

https://zone.msn.com/gameplayer/gameplayerHTML.aspx?game=texasholdem - for images of game buttons

https://www.pinterest.ca/pin/687010118138079233/ - for images of cards

© 2004 - 2024 PretendGINEERS. All Rights Reserved.