2016

OMI ONLINE!

CO324 – PROJECT 2

E/12/162, E/12/190

Group 20

Content

1. Brief Introduction about Game
2. Requirements
3. Designing
4. How to use Application
5. How to setup the Server
6. References

1. Brief Introduction about Game

The card game Whist better known locally as “Omi’ is a popular Sri Lanka. But the game that was implemented for this project is bit difference. There are no teams in this game. Each player plays individually. 52 cards in the deck must get for the game. The power of cards are in descending order as **A K Q J 10 8 7 6 5 4 3 2 1**. Each player get 13 cards for a round. Trump will be the last card of distribution after shuffling.

The first player may begin the game by playing any card. The rest of the players must play the same suit as the card that leads the trick, if they can. Otherwise they may play any card. If a trump suit card is played, the highest trump wins. Otherwise the high card of the suit wins the trick. The winner of a trick leads the next turn.

The player who wins 10 trick at first will be the winner.

2. Requirements

- **Iteration 1**
  Connecting 4 players to the game and dealing cards to each player is completed

- **Iteration 2**
  Game logic and wining player is decided

3. Designing

- **Communication**
  - The updates from instance such as
    - player played a card
    - current player
    - trick winner
    - game winner

    are sending to the browser page via **Server send events**. The system is implemented in such a way that server can send messages separately to each player.

    The SSE (Server Sent Events) are handled by doGet method in servlet.

  - The request from player: the cards played by the player are send to server by **AJAX**.
    All AJAX requests are POST requests and handled by doPost method in servlet.

- **The JSON object receive by player**
  - Parameters
    - cards : the player hand
    - showCards : view the cards on the desk
    - showHand : view the player hand
    - showNewGame : view NewGame button
    - shoeLeaveGame : view LeaveGame button
    - currentScore : player's trick wins
    - trump : trump card
    - message : display message (Application message)
    - player : name assigned to player
    - card1 : player 1 card
    - card2 : player 2 card
    - card3 : player 4 card
    - mycard : players card if player is player 3

- **Classes**
  - **BackBone**
    - BackBone handle the requests and responses. The connections of 4 players are saved in this method as a HashMap<String,AsyncContext> where string is the player id and AsyncContext is the connection to the player.
    - Each player is identified by a session id. So with same session, two players cannot be login.

  - **HandleWhist**
    - HandleWhist handle this game. The main items needed for the game like
      - Game logic
      - Message api (API to make messages for each instances)

      are implemented in here.

    - This class implemented to serve more than one game. But its not completed properly by now.

  - **SendingMessage**
    - The message structure which needed to build above JSON object is defined in here.
    - This have few constructors to serve various message types.

  - **AccessCheck** (enum)
    - This helps to move in state when each player is connected to the game.
    - This have five stages as each player connects.

  - **CardPlay** (enum)
    - The playing game logic and states are implemented in here. It changes state from player to player when round is playing and wining.

  - **GameFlow** (enum)
    - The total game flow should implement in here. But currently its not completed. The game flow is implemented in HandleWhist class by now.
  - **Deck**
    - This class implement universally to support any card game. Few main functions needed by card games are implemented in here.

4. How to use the Application

The players should connect to the server via a web browser. Lets assume server is in local server with private ip 192.168.1.3
To connect to the game, user should access **192.168.1.3/whist** from the browser.
The currently connected players will display in the window. Currently only four players can play by connecting to the game. After all connected, players should play according to the rules.

5. How to setup the server

This is a web application, so it will run on servlet supported servers.
The zip file will contain the **source code** and **war** packaging file.

How to run

- The **whist.war** file can be copy to **webapps** folder in server and restart server. Then this game will deploy automatically by the server.
- The project is build using maven. So maven packaging can be used to make the war file and game app folder (whist). The created folder can directly copy to webapp folder in servlet server. Restarting server will automatically deploy the game server.

For recent updates in the game : - https://github.com/DeshanKTD/whist

6. References

a. TutorialsPoint
b. StackOverFlow
c. Java Official Documentation
d. Java EE documentation