A Project Report On

FIVE IN A ROW

For the course

DATA COMMUNICATION AND COMPUTER NETWORKS

Project done by

Shah Riddhi G.
0003553764
ridshah@iupui.edu

in partnership with

Kavya Shankar
0003575005
kavshank@iupui.edu

Date:
12/01/2015

Abstract

Five in a row is an ancient game played with stones on a checkered board. It is also known as Gomoku, Gobang. It was traditionally played with black and white stones on a go board. Here we have implemented this game as a two player online game on a 19x19 checkered board with ‘O’ and ‘X’ similar to tic tac toe. The main idea of the game still remains the same. Any player who manages to get five consecutive ‘X’ or ‘O’ in a horizontal, vertical or diagonal checks wins. We have used HTTP Protocols to implement this.

Table of Contents

CHAPTER-1 INTRODUCTION

Five in a row is an abstract strategy board game. It is also known as Gobang or Gomoku. Traditionally it is played on 15x15 board but we are implementing it on a 19x19 board. It is normally played with black and white stones but we have implemented it with 'X' and 'O'. The player who makes a row, column or diagonal of five consecutive same pieces wins the game.

Five in a row is implemented as an online game to be played between two players. The player who clicks the start button first goes first. The first player gets the 'X' and the second player gets the 'O'. The game can be reset at any time by pressing the reset button.

The technology used to implement this online game is HTML, JavaScript and Node.js 'http' module to establish the connection and implement the HTTP protocol. Simple 'GET' and 'POST' methods are used to connect the players with each other.

The five in a row online game is implemented using the concept of Peer to Peer architecture. "Peer-to-peer (P2P) computing or networking is a distributed application architecture that partitions tasks or workloads between peers. Peers are equally privileged, equipotent participants in the application. They are said to form a peer-to-peer network of nodes."- Wikipedia

Player 2 connects to player 1. The connection is established using the node.js http module. The board appears on both the player's computer. The player who starts the game first will always get the first turn. When the player clicks on the board the turn is taken and the appropriate piece is placed. A post request is made which sends the information about which square is clicked on the board. To send this information on the board of other player, the board is refreshed after every click and reloaded. Also, after every click the status of the game is checked to see if the move caused the player to win. The game is won in a case where there are 5 consecutive 'X' or 'O' in a row, column or diagonal. If the game is won, alert message is displayed stating that the game is won else the game continues.

The front end of the game is built using HTML and jquery in JavaScript. JavaScript is also used in the back end along with node.js.

CHAPTER-2 IMPLEMENTATION

Connection is established between two machines using HTTP protocol using node.js. After the connection is made dispatcher is implemented on the backend side to handle multiple post requests from peers.

There are different functions like checkstatus, reset, start, and update and broadcast which implements different post requests. These functions are implemented using the dispatcher. Each function creates a URL based on its function. The dispatcher listens to the URL and the appropriate function is implemented.

Below is a list of functions and their explanation:-

Back end functions:

GET request "/game":
When the client loads the game with the url ending with "/game" a GET request is sent to the server and response is in the form of HTML data which gets loaded on the browser.

POST request "/start":
When the player clicks on START button , a POST request is sent to the dispatcher with the regular expression "/start" and the game starts and status of the current player is set to Y and the opponent to N.

POST request "/checkstatus":
When the player clicks on a box on the board a POST request is sent to the dispatcher containing the regular expression "/checkstatus". A check is made to see if the current player is allowed to play or not. If not he cannot make a move before the opponent player.

POST request "/update":
Once the player has clicked on a box and the response from the server is Y, a POST request is sent to the server where the dispatcher with the regular expression "/update" is matched to update the status of the current player to N so that he is not allowed to play again and the opponent player's status is changed to Y so it's his turn to play the game.

POST request "/broadcaste":
The server with dispatcher matching the regular expression "/broadcaste" receives the POST request from the client periodically every 2 seconds to update the board with the Xs and Os

POST request "/reset":
When the player clicks on RESET button, a POST request is sent to the server where the dispatcher with the regular expression "/reset" is matched and the entire game is reset by clearing the board of both the players.

Front End functions:

$("#start").click ():
When the player clicks on the START button a POST request is sent to the server using an AJAX call XMLHttpRequest(). The status of the player who clicked on START is set to Y and the other player to N.

$("#table_tag td").click ():
When the player clicks on the box in the board a POST request is sent to the server using an AJAX call XMLHttpRequest(). The status of the player is checked to see if he is allowed to play. If he is not an alert message is displayed to tell him that it's not his turn.

updateBox ():
When the player successfully places the X or O in the box, his status is changed to N and the opponent's status is changed to Y so it's his turn to play.

setImage ():
This function is used to place the Xs and Os in the right position where the players have clicked.

checkforwin ():
This function consists of the winning logic for 5 Xs or 5 Os placed horizontally, vertically, diagonally or anti-diagonally as well as draw.

broadcasteData ():
This function is called every time there is a change to the board contents and it makes sure both players' browser displays the same board.

drawboard ():
Contains the logic to draw the gomuku board.

$("#reset").click ():
When the player clicks on the RESET button a POST request is sent to the server using an AJAX call XMLHttpRequest(). The board is cleared and the status of the players is reset.

CHAPTER-3 MY ROLE IN THE IMPLEMENTATION

Back-end implementation:-

- Dispatcher handler "/checkstatus" checks whether the player I allowed to make the move or not. This handler is important so that one player does not get more than one turn until the second player has taken its turn. For this two variables are used isFirst and isSecond and the possible values for these variables is Y and N. Using this it is decided if it is the turn of player 1 or player 2 and accordingly the value is passed in the response.
- Dispatcher handler "/start" is called when the player presses the start button. The player who presses the start button first is considered the first player and other is considered the second player. Accordingly the variables isFirst and isSecond are set.
- Dispatcher handler "/broadcaste" gets the response every 2 seconds from the machines and updates the board accordingly so that both the machines are in sync with each other's move. It sends a post message containing the moves made and the isFirst and isSecond value.

Front-end implementation:-

- JavaScript jQuery is used to design the HTML 19x19 boards. All the td elements are given a id "[row]_[column]". This id is used to know where the move is made. Two buttons start and reset are used to start and reset the game respectively. Start calls the start dispatcher and reset calls the reset dispatcher.
- The checkforwin () function checks whether the move made by the player results in winning of the game. This function will be called every time a move is made. This checks for five consecutive 'X' or 'O' in rows, column or diagonal. Also it checks if all the checks are filled and if the game is drawn.
- The setimage() function sets the 'X' or 'O' on the box clicked. It won't allow the turn to be taken on the same box.
- The broadcasteData () function calls the broadcaste dispatcher.

CHAPTER-4 WIRESHARK ANALYSIS

When the player opens the game a GET message is received from the source. A HTTP response with status code 200 is sent to the source along with the HTML data in the line based text data. The internet protocol version used is TCP. So TCP packets are sent to the source containing the line based text data. These TCP packets are reassembled on the other side and HTML data is created. An acknowledgement is then sent with status code 200.

After the game is opened the player presses the start button which sends a POST message with start parameter. In response the state whether two players are connected or not is sent using the 'Y' and 'N' parameter. First parameter is for the first player and second parameter is for the second player. An acknowledgement is received from the source which means the parameters are successfully passed.

After both the players are connected and one player makes the move a POST message is sent. The response to this message is sent which contains the position of the move made. ACK acknowledgement is received indicating that the response has been received. An acknowledgement is sent to indicate that the data has been received. The connection is not closed anywhere. The connection remains open until the game is finished. So for every new request to be sent a new connection is not established.

After every move is made checkstatus will be executed. Checkstatus checks if the player is allowed to make the move or not. So a POST message is sent. The message will have parameter of 'Y' and 'N' for the first and the second player. The response will contain the parameter based on whose turn it is. An acknowledgement is received ACK and a counter ACK is sent.

An update message is sent using the POST message to give the position of the move made by the other player. The response to this message will contain a list of all the moves made and information about whose turn it is next. This continues until the game is finished.

The wireshark trace gives an insight to what calls are made to establish the connection between the two systems. Initially when the game is loaded the HTML content is passed in the line based data and reassembled on the other side. Also, the GET and POST messages are sent and received containing the parameters about player turn and moves made.

REFERENCES

https://en.wikipedia.org/wiki/Application_layer
https://en.wikipedia.org/wiki/Peer-to-peer
https://en.wikipedia.org/wiki/Gomoku
https://nodejs.org/api/http.html
http://www.html5rocks.com/en/tutorials/websockets/basics/#toc-introduction-sockets
https://docs.nodejitsu.com/articles/HTTP
https://strongloop.com/strongblog/real-time-engines-in-node-js/
http://blog.frankgrimm.net/2010/11/howto-access-http-message-body-post-data-in-node-js/
http://www.htmlgoodies.com/html5/javascript/an-intro-to-node.js.html
http://blog.modulus.io/build-your-first-http-server-in-nodejs