Server based law general essay

Law

1. Introduction

This document defines how to play a Tic-tac-toe game over TMPP (TMPP is temporary Protocol). It is quite common to play games over Servers. Since Tic-tac-toe is a well-known and simple game, it's a good example for a server-based game protocol.

2. Requirements

This document addresses the requirements for a game protocol as defined by Multi-User Gaming. In particular this consists of: Game rolesMatch stateTurnsMatch configurationConditions for interrupting and terminating a match

3. Discovering Support

An entity implementing this protocol MUST also support Multi-User Gaming and answer to disco requests accordingly including both the Multi-User Gaming and the game elements in the response.

Example 1. Service Informs ABC That It is Capable of Hosting Tic-tac-toe

from='www. website. org’id='disco1'to='ABC@website. com/garden'>

...

...

4. Game Roles

A Tic-tac-toe game uses two game roles, " x" and " o". Both roles have to be assigned to exactly one player to start a match. If one role gets unassigned or a player gets unavailable the match has to be paused.

5. Match Configuration

A service SHOULD offer a configuration form with the following options: Starting role (" mug/tictactoe#config_first")Size of the board (" mug/tictactoe#config_rows" and " mug/tictactoe#config_cols")The number of marks in a row needed to win (" mug/tictactoe#config_first")An implementation MUST be able to handle the board with three cols and rows and three respective marks to win. Everything beyond that is OPTIONAL.

Example 2. Service Sends Configuration Form

id='config1'to='ABC@pqrs. com/garden'type='result'>

...

Below you can see the default configuration. To accept the default configuration, click OK. To select a different configuration, please complete this form. Submitting a strike option value bigger than the number of rows or columns SHOULD result in error.

6. Match State

The state in a Tic-tac-toe match represents the match configuration, the player who makes the next turn and the current state of the board. Every state is distributed to all occupants.

Example 3. Service Sends Start State

from='tictactoe@games. shakespeare. lit'to='ABC@pqrs. com/garden'>

After a valid turn, the state gets updated with the corresponding mark.

Example 4. Service Sends Mid-Game State

from='tictactoe@games. shakespeare. lit'to='ABC@pqrs. com/garden'>

Example 5. Service Sends Draw State

from='tictactoe@games. shakespeare. lit'to='ABC@pqrs. com/garden'>

7. Turns

During the game, player’s change in turn, each of them MUST send only one move at a time. It MUST possess these attributes:

Table 1: attributes

Name

Type

Description

'id'REQUIREDThe number of the move. First move is 1.'row'REQUIREDThe horizontal position of the mark.'col'REQUIREDThe vertical position of the mark.

Example 6. ABC Sends a Move

8. Security Considerations

The author is not aware of any security issues introduced by this protocol extension.

9. FLOW CHART

ServerCreate a server socket. Accept connection from the first player and notify the player is Player 1 with token X. Accept connection from the second player and notify the player is Player 2 with token O. Start a thread for the session. Player 11. Initialize user interface. 2. Request connection to the server and know which token to use from the server. 3. Get the start signal from the server. 4. Wait for the player to mark a cell, send the cell's row and

column index to the server. 5. Receive status from the server. 6. If WIN, display the winner; if player 2 wins, receive the last move from player 2. Break the loop7. If DRAW, display game is over; break the loop. 8. If CONTINUE, receive player 2's selected row and column index and mark the cell for player 2. Player 21. Initialize user interface. 2. Request connection to the server and know which token to use from the server. 3. Receive status from the server. 4. If WIN, display the winner. If player 1 wins, receive player 1's last move, and break the loop. 5. If DRAW, display game is over, and receive player 1's last move, and break the loop. 6. If CONTINUE, receive player 1's selected row and index and mark the cell for player 1. 7. Wait for the player to move, and send the selected row and column to the server. Handle a session: 1. Tell player 1 to start. 2. Receive row and column of the selected cell from Player 1. 3. Determine the game status (WIN, DRAW, CONTINUE). If player 1 wins, or drawn, send the status (PLAYER1_WON, DRAW) to both players and send player 1's move to player 2. Exit.

•

4. If CONTINUE, notify player 2 to take the turn, and send player 1's newly selected row and column index to player 2. 5. Receive row and column of the selected cell from player 2. 6. If player 2 wins, send the status (PLAYER2_WON) to both players, and send player 2's move to player 1. Exit. 7. If CONTINUE, send the status, and send player 2's newly selected row and column index to Player 1.