Cody Farris

Tic-Tac-Toe Restful Web Service Documentation

**Setup:**

This program requires a connection MySQL database. Once you have an instance of MySQL running in your environment then create a schema and use the file tictactoe\_tablesCreate.sql to build the required tables.

Once your database is running correctly you’ll now need to edit the game.yml file with the correct configuration. You’ll need to place your MySQL username, password and JDBC URL. The last portion of the JDBC URL is the name of the schema that you placed the tables created by tictactoe\_tablesCreate.sql.

Lastly you’ll need to open your command line and change your directory to the one containing game.yml and the rest of the project. Now to start the web service run the following command

java –jar target/my-project-0.0.1-SNAPSHOT.jar server game.yml

That should complete the setup for the Restful web service.

**Playing Tic-Tac-Toe Part 1: Creating Players**

The general workflow of using this web service to play Tic-Tac-Toe is a series of POSTS to different locations. First you must create your Player by posting to <http://localhost:8080/players>. The following is an example cURL POST command.

curl -i -H "Accept: application/json" -H "Content-Type:application/json" -X POST <http://localhost:8080/players>

If successful you’ll receive a 201 response with the contents of your player. Please remember your id number because you’ll need it to initiate a game. You’ll also need a second player so use the same cURL POST command to create them.

**Playing Tic-Tac-Toe Part 2: Creating a Game**

To create a game you’ll have to have two pieces of information. The id number of your intended player one and the id number of your intended player two. Player One always places X and Player Two always places O so remember this as you are assigning players. If you wish for player two to go first you must indicate this by including a third piece of information nextMove. NextMove should have player twos id number as its value. The JSON sent to the web service can therefore be two different formats when posting to games.

{

"playerOne": 1,

"playerTwo": 2

}

Or

{

"playerOne": 1,

"playerTwo": 3,

“nextMove”: 3

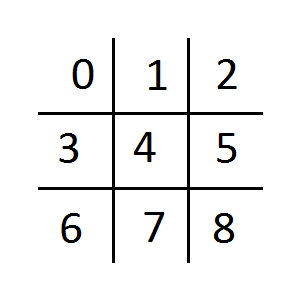
}

curl –i –H “Accept: application/json” –H “Content-Type:application/json” –X POST –d @game.json <http://localhost:8080/games>

Returned from the POST request will be a JSON game object. This will contain the game id, nextMove, playerOne, playerTwo, winner, and links to useful objects. Winner is initially set to -1 to represent no winner of that game. The last link is to the first gamestate object for the current game. The first gamestate denoted by moveNumber 0 has an empty Tic-Tac-Toe board represented by “EEEEEEEEE”.

**Playing Tic-Tac-Toe Part 3: Making a Move**

To make a move you’ll need to post to the gamestate url for the game your trying to make a move in. The board is represented by a 9 character string which starts as “EEEEEEEEE”. When choosing which ‘E’ to replace with your move ‘X’ or ‘O’ you can think of the board as “012345678” where each number is assigned like the image below.



As in all games of Tic-Tac-Toe you cannot overwrite a previous move and only one move may be made at a time. Each player must take a turn, if an ‘X’ was placed previously then an ‘O’ must be next and vice versa. Valid win scenarios are three in a row vertically, three in a row horizontally, or three in a row diagonally. Once a game is won neither player is able to post another move to gamestate. After the win the game’s winner variable sets the player id who won the game and the player’s winCount is incremented by 1.

To post using cURL issue the following command where {gameId} is the game you are making a move on and slotValues.json is the file with your json content.

curl –i –H “Accept: application/json” –H “Content-Type:application/json” –X POST –d @slotValues.json [http://localhost:8080/games/{gameId}/gamestates](http://localhost:8080/games/%7bgameId%7d/gamestates)

Example of json conent for slotValues.json

{

"slotValues": "EEEEEXEEE"

}

**Additional Information:**

While links are provided with each object return in this restful web service you may want additional information at any point. The follow get commands also work for cURL

Full list of games: curl <http://localhost:8080/games>

Specific game: curl [http://localhost:8080/games/{gameId}](http://localhost:8080/games/%7bgameId%7d)

Full list of players: curl <http://localhost:8080/players>

Specific player: curl [http://localhost:8080/players/{playerId}](http://localhost:8080/players/%7bplayerId%7d)

Full list of gameStates for a single game: curl [http://localhost:8080/games/{gameId}/gamestates](http://localhost:8080/games/%7bgameId%7d/gamestates)

Specific gameState: curl [http://localhost:8080/games/{gameId}/gamestates/{moveNumber}](http://localhost:8080/games/%7bgameId%7d/gamestates/%7bmoveNumber%7d)

Examples of the submitted json POST files are also provided in the exampleJSON folder.