CS 3360 - Design and Implementation of Programming Languages

PROJECT 1: WEB SCRIPTING WITH PHP
(File $Date: 2018/08/23 19:55:28 $)

Due: TBA

This assignment may be done individually or in pairs. Research has shown that students learn better when they do homework assignments in pairs. However, it does not necessarily mean a decrease in the time you will spend on your assignments. If you work in pair, you need to fill out the contribution form (see the course website).

The purpose of this assignment is to understand the concepts of web scripting languages and have a hands-on experience with web scripting by writing small web service code in PHP [1].

You are to write a lightweight web service in PHP for playing Connect Four games. The Connect Four game is a two-player connection game in which the players take turns dropping their discs from the top into a seven-column, six-row vertically suspended grid [Wikipedia]. The pieces fall straight down, occupying the next available space within the column. The objective of the game is to connect four of one's own discs next to each other vertically, horizontally, or diagonally before your opponent.

As a web service, your PHP code provides a few APIs implementing the
logic, data or process for playing Connect Four games, not a complete
Web application including a GUI. Do not include any GUI components in
your web service. The web service APIs are predefined and specified as
URLs (see below). Below are key requirements for your web service.

R1: The web service shall work with a provided Java client available
    and downloadble from the course website (c4Client.jar).

R2: The web service shall support multiple clients concurrently, each
    client playing against the server (see R5 below).

R3: A game board shall consist of six rows and seven columns, i.e.,
    6*7 places in which discs can be dropped.

R4: A column of the board shall be identified by a 0-based index,
    e.g., 2 for the third column.

R5: The web service shall provide at least two different move
    strategies for the computer, say Random and Smart. The Smart
    strategy shall be indeed "smart" to allow for a realistic
    play. Minimally, it should detect a winning/loosing row, i.e.,
    three consecutive discs with an open end.

R6: The web service shall determine the outcome: win, loss, or draw.

The web service APIs are implemented as a set of URLs as specified
below. All communications between the web service and a client shall
be done using the HTTP query string and the JavaScript Object Notation
(JSON), a lightweight data-interchange format (see www.json.org)
[3]. All inputs to the web service shall be obtained from HTTP query
strings, and all outputs to the client shall be written in JSON. Following the REST principles, the web service shall be
stateless and provide the following three URLs:

1. http://<c4-home>/info (short for .../info/ or .../info/index.php),
   where <c4-home> is the address of your Connect Four service
   typically consisting of a host name and a pathname. Provide game
   information, including the board size and available computer move
   strategies. Below is a sample JSON output.

   {"width":7,"height":6,"strategies":["Smart","Random"]}

   Hint: Use json_encode() function to create a JSON (string)
   representation of a PHP value or object.

2. http://<c4-home>/new?strategy=s
   Create a new game to play against the specified computer strategy.
   A normal response will be a JSON string like:

     {"response":true,"pid":"57cdc4815e1e5"}

   where pid is a unique play identifier generated by the web service.

It will be used to play the newly created game (see 3 below).

Upon an error, the web service will notify the client by providing an
an appropriate error response like:

{"response": false, "reason": "Strategy not specified"}
{"response": false, "reason": "Unknown strategy"}

Hint: use uniqid() function to generate a unique identifier based
on the current time in microseconds. Use the Strategy design pattern
to define different strategy classes, e.g., RandomStrategy and
SmartStrategy [Chapter 5 of 2].

3. http://<c4-home>/play?pid=p&move=x
Make a move by dropping a disc in the specified column, x, to play
the specified game, p. Example: .../play/?pid=57cdc4815e1e5&move=3.

A normal response will be a JSON string like:

```
{"response": true,
 "ack_move": {
   "slot": 3,
   "isWin": false,   // winning move?
"isDraw": false,  // draw?
"row": []},       // winning row if isWin is true
 "move": {
   "slot": 4,
   "isWin": false,
   "isDraw": false,
   "row": []}}
```

where "ack_move" is the acknowledgement and the outcome of the
requested move of the player, and "move" is the computer move made
right after the player's; there will be no computer move if the
player move is a game-ending (win or draw) move. For a winning
move, the value of "row" is an array of numbers denoting the
indices of the winning row [x1,y1,x2,y2,...,xn,yn], where x's and
y's are 0-based column and row indices of places, e.g.,

```
"row":[0,5,1,5,2,5,3,5]
```

If there is an error, it should be reported to the client by
providing an appropriate error message like:

```
{"response": false, "reason": "Pid not specified"}
{"response": false, "reason": "Move not specified"}
{"response": false, "reason": "Unknown pid"}
{"response": false, "reason": "Invalid slot, 10"}
```

Hint: Define several classes to model the Connect Four game, e.g.,
Play, Board, etc. The states of some of these classes need to be
stored externally, say in a file, because the web service sessions
are stateless and the game state should be preserved between
sessions. You may use JSON string to persist game states.

Hint: You may use the player identifier (pid) as a file name to
store the game state, and the game state may be stored as or

restored from a JSON string; use json_encode() and json_decode()
functions.

Hint: Use json_decode() function to convert a JSON encoded string
to a PHP value or object.

With above three web service URLs, a Connect Four game can be played
as follows:

Step 1: visit http://<c4-home>/info to find game info
Step 2: visit http://<c4-home>/new?strategy=s to create a new game
Step 3: repeatedly visit http://<c4-home>/play?pid=p&move=x
to drop a disc, e.g. in pseudo code:

```
while (true) {
   visit http://<c4-home>/play?pid=p&move=x
   if (ack_move.isWin) {
       break; // player won
   } else if (ack_move.isDraw) {
       break; // draw
   } else if (move.isWin) {
       break; // computer won
   } else if (move.isDraw) {
       break; // draw
   }
}
```

You can test run a sample implementation at http://www.cs.utep.edu/cheon/cs3360/project/c4/.

HINTS

It is strongly recommended to use the PHP Development Tools (PDT),
an Eclipse plugin for PHP (or PhpStorm for IntellJ IDEA).

Create in your src directory/folder three subdirectories/ folders
corresponding to the three URLs. Each subdirectory should have a PHP
file named index.php, the entry point of the provided API, e.g.,

info/index.php
new/index.php
play/index.php

If this is your first PHP project, start early. You need a bit of
time to get familiar with PHP scripting and to configure and get
used to its support tools and development environment.

TESTING

You code should be deployed and run correctly on the class website
at http://cs3360.cs.utep.edu. Refer to the FAQ page of the CS Tech
Support website for accessing the class website and uploading files
(http://csts.cs.utep.edu/faq/index.html; see the section "Web
Related").

WHAT AND HOW TO TURN IN

Submit your code along with any supporting documents through the
Assignment Submission page found in the Homework section of the
course website. The page will ask you to zip your code and support
files and upload a single archive file. The zip archive file should

include only a single directory named YourFirstNameLastName which
contains all your source code files and other support files needed
to run your scripts. You should submit your work by 11:59 pm on the
due date. If you work in pair, make just one submission by listing
both names in your submission.

You should also deploy your code to the class website located at
http://cs3360.cs.utep.edu. Make sure your code run correctly on
this particular web server (PHP version 5.3.3); TA will test your
your deployed code, not the submitted one.

If you work in pair, make only one submission through the
Assignment Submission page by specifying both names during the
submission; make sure to include the contribution form in your
submission. However, each member of your pair needs to deploy the
code to his/her own class website (cs3360.cs.utep.edu).

GRADING

You will be graded in part on the quality of the design and on how
clear your code is. Excessively long code will be penalized: don't
repeat code in multiple places. Your code should be reasonably
documented and sensibly indented so it is easy to read and
understand.

Be sure your name is in the comments in your code.

REFERENCES

[1] Rasmus Lerdorf and Kevin Tatroe, Programming PHP, 3rd edition,
O'Reilly, 2013. Ebook through UTEP library.
[2] Junade Ali, Mastering PHP Design Patterns, Packt Publishing,
2016. Ebook.
[3] Ben Smith, Beginning JSON, Apress, 2015. Ebook.