CS 1520 / CoE 1520: Programming Languages for Web Applications (Spring 2012)

Department of Computer Science, University of Pittsburgh

Assignment #2: PHP

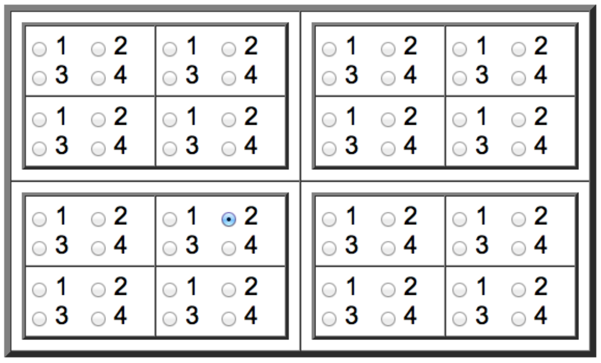
Released: February 28th, 2012 **Due:** 11:59pm, Tuesday, March 20th, 2012

Goal

Gain familiarity with PHP.

Description

In this assignment, you will implement in PHP a variant of the popular Sudoku game. Our version of Sudoku is played on a 4×4 grid, which is divided into four 2×2 regions (top left, top right, bottom left, bottom right). The goal of the game is to place the numbers 1-4 into the grid such that every number appears exactly once in each row, each column, and each region. The game begins with several numbers that cannot be moved already in place; the player must reason logically to determine how to fill the remaining squares. There is a lot of information to be found at Wikipedia's article on Sudoku[1], and there are many online versions available, on 9×9 grids.


(a) Sample board setup

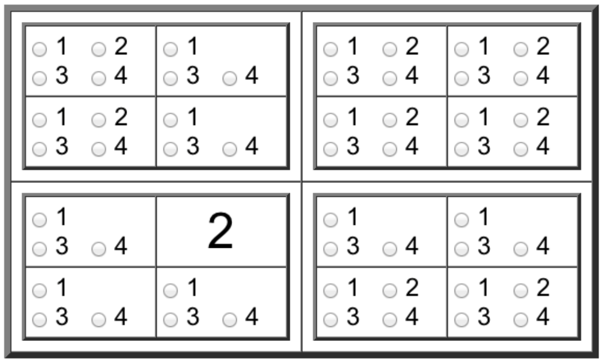

(b) Board after one move

Figure 1: Sudoku Game Board

Web Page components

The web page that your program `sudoku.php` creates should include the following components:

1. **Sudoku board** – a 2×2 grid of regions, each of which has 2×2 cells, each of which has (in the beginning) the numbers 1, 2, 3, 4 arranged in a 2×2 pattern (Figure **??**a). Each number is essentially a single form input element of type radio (only one button at a time could be selected from the entire board).

2. **Sudoku initialization form** – a text input field that can be used to initialize the board to a certain configuration. A configuration will be a series of 19 characters as follows: `ZZZZ:ZZZZ:ZZZZ:ZZZZ`, where each `Z` can be either a single digit 1, 2, 3, 4, or an `x` (lower-case) to denote no number in that location. A configuration maps to the Sudoku board in a straightforward way: the first four digits

[1]http://en.wikipedia.org/wiki/Sudoku

correspond to the four cells of the top-left region, the second set of four digits corresponds to the four cells of the top-right region, the third set of four digits corresponds to the four cells of the bottom-left region and the last four digits correspond to the four cells of the bottom-right region. Within each set, the order is again top-left, top-right, bottom-left, and bottom-right. For example, the board in Figure **??**b (with the 2 selected) can be encoded as `xxxx:xxxx:x2xx:xxxx`.

3. **Action buttons** – there are five action buttons:
   - **Initialize board** (next to the initialization form) – will initialize the Sudoku board to the configuration submitted in the initialization form. All past moves will be lost. A sanity check should be made to see if the submitted configuration is a valid one (both in terms of string format and in terms of valid Sudoku moves).
   - **Clear board** (next to the initialization form) – this is equivalent to call the Initialize board, with the configuration `xxxx:xxxx:xxxx:xxxx`.
   - **Submit move** (at the bottom of the board) – this "executes" the move, after the user has selected one of the currently available numbers on the board. A sanity check should be made to see if the submitted move is a valid one.
   - **Back** (at the bottom of the board) – this button is only displayed if the submitted configuration or submitted move is not a valid one, in which case the option to return to the previous state is offered to the user. In such a case, there is no Undo move and no Submit move buttons displayed.
   - **Undo move** (at the bottom of the board) – assuming a valid move, this returns the board to the previous state before the last move was executed. You should keep all the moves in some sort of history (either as a hidden form element, as a cookie, or through a session variable). This should allow the user to go all the way back to the last board initialization (one move at a time). Once a move is undone, it cannot be redone (unless the user reselects the same number).

You are also encouraged to add some form of personalization to your assignment (e.g., a header on the web page with your name and other information) and utilize CSS to simplify your HTML page design (although these components of your program will receive minimal attention during grading).

Please note that you should NOT use Javascript in your program.

Game integrity

It is crucial that the Sudoku board is valid at all times according to the three rules of (4×4) Sudoku:

- there is no more than one of 1, one of 2, one of 3, and one of 4 appearing in each row
- there is no more than one of 1, one of 2, one of 3, and one of 4 appearing in each column
- there is no more than one of 1, one of 2, one of 3, and one of 4 appearing in each region

The above rules essentially form a set of 12 constraints (4 rows + 4 columns + 4 regions), which must not be violated by a move or by a new board configuration.

If a new board configuration or a submitted move leads to any of the 12 constraints being violated, then the board should be displayed with the erroneous configuration and the background color of the appropriate cells should be turned red. If a certain choice violates more than one rules, all should be highlighted appropriately.

In case of a rule violation, a Back button should be displayed (no Undo move or Submit move buttons), which will revert back to the previous version of the board, when clicked.

In cases of a new board configuration that violates one or more rules, the board should be reset to the previous state (after the Back button is pressed) and the previously submitted configuration (e.g., xxxx:2343:xxx1:xxx2) should be displayed in the text box.

HINT: You are highly encouraged to write a function that checks a board configuration against the different constraints and use it both for checking new configurations and new moves.

Game hints

During the execution of the game, your program should identify what are the remaining valid choices and only enable those numbers as options for the next move. For example, the board after a move where the number 2 was selected (as in Figure **??**a) should have all the 2's from the same column, same row, and same region eliminated as options (as in Figure **??**b).

What to submit

Although you will be testing your PHP code with the PHP setup provided and with the PHP files being in your own accounts, you must submit all the required files so that your program can be executed at a different web server. Name the top-level file `sudoku.php`. Make sure that all paths are **relative**, if you have different files to include in your program. Submit all files as a single zip file.

Academic Honesty

The work in this assignment is to be done *independently*, by you and only you. Discussions with other students on the assignment should be limited to understanding the statement of the problem. **Cheating in any way, including giving your work to someone else, will result in an F for the course and a report to the appropriate University authority for further disciplinary action.**

How to submit your assignment

We will use a Web-based assignment submission interface. To submit your assignment:

- Go to the class web page `http://db.cs.pitt.edu/courses/cs1520/spring2012` and click the Submit button.
- Use your pittID as the username and the password you specified at the contact information form for authentication. There is a reminder service via email if you forgot your password. You must have already submitted your contact information, if you have not yet you need to do so now.
- Upload your assignment file to the appropriate assignment (from the drop-down list).
- Check (through the web interface) to verify what is the file size that has been uploaded and make sure it has been submitted in full. **It is your responsibility to make sure the assignment was properly submitted.**

You must submit your assignment before the due date (11:59pm, Tuesday, March 20th, 2012) to avoid getting any late penalty. The timestamp of the electronic submission will determine if you have met the deadline. There will be no late submissions allowed after 11:59pm, Thursday, March 22nd, 2012.

[Last updated on February 28, 2012 at 1:33am EST]