CS 1520 / CoE 1520: Programming Languages for Web Applications (Spring 2013)

Department of Computer Science, University of Pittsburgh

Assignment #3: Javascript

Released: April 4th, 2013 **Due:** 11:59pm, Friday, April 19th, 2013

Goal

Gain familiarity with Javascript and the DOM.

Game Description

Our version of Sudoku is played on a 4x4 grid, which is divided into four 2x2 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 great deal more information to be found at Wikipedia's article on Sudoku, and there are many online versions available, though usually on larger 9x9 grids.

Description

For this assignment, you will create a Javascript-based, 4x4 Sudoku game with hints. The game will consist of a grid of 16 cells in which the user can place the numbers 1 through 4 by clicking a corresponding hyperlink displayed in each cell. The goal of the game is to arrange the numbers such that each column of the grid, each row of the grid, and each quadrant contain each of the numbers in some order.

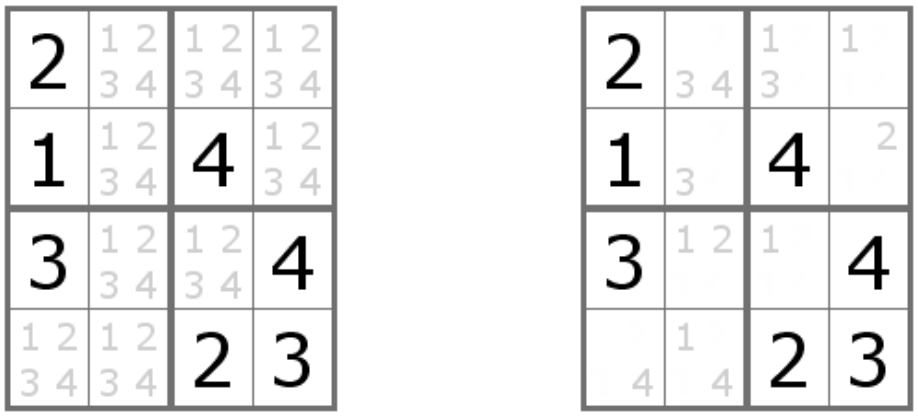
1. **Game Initialization**.
   The initial board information, containing the spaces that are already filled, will be imported into the game by linking to an external javascript file that defines a multidimensional array variable called *initBoard*. Each element of the array will be the number for the corresponding cell of the game or zero if the space should be initially empty. The file is located at:

   `http://db.cs.pitt.edu/courses/cs1520/spring2013/assign/sudoku.js`

   However you should also test your program with your own files. As an example, the file could contain the following statement, which should result in the initial table displayed in Figure 1:

```
var initBoard = new Array(new Array(2, 0, 0, 0),
                          new Array(1, 0, 4, 0),
                          new Array(3, 0, 0, 4),
                          new Array(0, 0, 2, 3));
```

2. **Cell Display**.
   For empty cells, up to four hyperlinks should display each of the possible numbers in small text and a faded color. When one of the hyperlinks is clicked, all the hyperlinks should be "removed" from the cell (i.e., made non-visible) and the corresponding number should be displayed in a font size that fills the cell and is a darker color. Borders for the cells should accurately define rows, columns, and quadrants such that a thicker line divides quadrants and an even thicker line surrounds the board (e.g., Figure 1).



(a) Inital Board (b) Board with Hints

Figure 1: Game Board

3. **Hints**.
   When the user selects a number in a given box, that number cannot be used again in the same row, column, or quadrant. You application should help the user by enforcing this rule. Each of the hyperlinks for that number in the same row, same column, and same quadrant should be "removed" (i.e., made non-visible).

4. **Undo**.
   The game should provide an undo button (or hyperlink) outside of the main board that will undo the last move the user made (i.e., revert to the state of the board just before his/her move). Undoing a move should bring the board to its previous state, before that move was made. A user can click undo repeatedly and should be able to keep clicking on the undo button until he/she reaches the initial state of the board. At that point the undo button (or hyperlink) should become inactive and only be made active again after the user makes a new move.

5. **Move Counter**.
   You should provide a counter outside of the main board that will keep track of how many moves have been played. The counter should start at 0, and increase by one with any move the user makes or decrease by one every time the undo button is pressed.

6. **Winning**.
   When the user wins by filling in all the cells, an alert box should be displayed with a congratulatory message, and, once closed, the game should be reinitialized.

Notes

- Construct your board using the simplest HTML possible (e.g., tables within tables), then figure out the necessary CSS to display and style it as needed. Very few points (if any) will be given to HTML style, so please focus on the Javascript aspect of the assignment.

What to submit

One HTML file (called `4x4sudoku.html`), along with your javascript libraries that should be in the same directory (i.e., using relative paths). There is no standard naming scheme for the libraries, although the expectation for javascript files is that they end in `.js`.

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:

- If you have more than one file to submit, prepare your assignment for uploading, by generating a single zip file with all the files.
- Go to the class web page `http://db.cs.pitt.edu/courses/cs1520/spring2013` 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, Friday, April 19th, 2013) 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, Saturday, April 20th, 2013.

[Last updated on April 4, 2013 at 9:51pm EST]