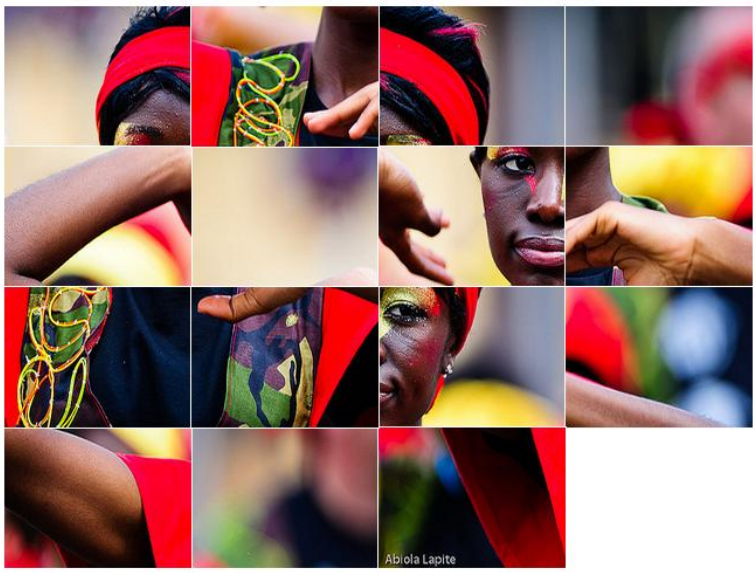
HCI-5/433D Web Lab
Project 1:
Sliding Block Puzzle
Due: Wednesday, September 21, 2011 by 11:59 pm



Goal
The goal of this project is to make a sliding block puzzle game similar to the game "15" (see http://www.proprofs.com/games/puzzle/sliding/ for an example of what we will make).

Files Provided
The following files are in the same .zip folder as this project description was in:

p1.js	JavaScript file that you will need to edit. Some skeleton code is provided for guidance, however you may implement this differently if you'd like.
p1.html	html file that will display your puzzle. NOTE: you MAY and SHOULD change this file if you want to make your puzzle look good.
img1.jpg	sample image that you can use for your puzzle.

Project Overview
For this project you will create a sliding block puzzle game similar to the popular game "15" using JavaScript. Your JavaScript app should do the following:

1. Make a grid of tiles, each of which shows a separate portion of an image. One tile from this grid should be missing. Your code should be written so that you can easily change the number of rows and columns in the grid (you may assume at least 2 rows/columns), as well as the entire width and height of the actual puzzle. These constants are provided for you in the JavaScript file; do not remove them.
2. Shuffle up the grid of tiles. Make sure that your shuffling results in a solvable puzzle state (hint: think of how you would shuffle a real puzzle like this).
3. If a valid tile is clicked, it should move to the empty square. If the tile is invalid nothing should happen.

4. To make your puzzle a little bit more interesting, you should randomize the puzzle a bit by randomly changing the number of rows and columns that the puzzle is made up of.

Bells and Whistles

Completing the above 4 requirements and having well-documented code with no errors will get you 45 out of 50 points, which is an A. You must complete one of the below 'bells and whistles' to get full points. You can complete as many of these as you want, but you will get no more than 10 points total. Different bonuses are worth different numbers of points, based on difficulty. You may receive up to 10 points for bells and whistles, bringing you up to up to 5 extra credit points. I hope these are fun for you!

1. **(2 points) Add variety by using different images in your puzzle:** Make your puzzle out of a variable number of images and randomly pick an image to show in your sliding puzzle.
2. **(1 points) Center your sliding block puzzle:** Make your puzzle look nice by centering it in the page.
3. **(5 points) Animate your tiles:** When you click on a tile, instead of moving it, figure out how to get your tiles to animate to their desired location. **NOTE:** You may NOT use any JavaScript libraries for this. I want you to figure out how to do this yourself. Hint: I used closures and setTimeout to accomplish this.
4. **(5 points) Auto solver:** Create a "solve me" button that shows the user how to solve the puzzle. Note you cannot just set the puzzle into its correct position as then the user won't know how to actually solve it. You must wait a bit before you move each block (would go well with the animating bell). You should also create a "hint" button that gives the user a hint.
5. **(up to 10 points) Make your page look awesome:** Right now the sliding puzzle is just on a white page and doesn't look very good. If you want you can design a web page to make your puzzle look very nice. I will assign a variable number of points here based on aesthetic appeal. If your page looks awesome you will get 10 points. Please make sure to include all files necessary in your turn in so I can see your puzzle correctly!
6. **(10 points) Generate a puzzle from an image pulled from Flickr:** You should be able to use the Flickr API to grab an image from some set and use this as your puzzle image.

Here are suggestions when implementing your code:

- Each tile in your puzzle should be a div. Remember, when you create divs in JavaScript you're just creating a JavaScript object, so this means you can add extra properties and functions to help track that state of the game.
- You shouldn't need to cut up your puzzle image into a bunch of smaller images and then manually reference each image. Instead, you should take advantage of the "CSS sprite trick" (http://css-tricks.com/158-css-sprites/) to show portions of the completed image in your divs.

Show and Tell

The course instructors will be picking the nicest puzzle implementations and showing them to the class. They may also be put on a public website. Please let me know if you do not want your solution shown in the README file in your assignment.

Turning Your Program In

The program is due Wednesday, September 21, 2011 at 11:59 pm. You should turn in your assignment on Blackboard by going to the assignment link, attaching a zip file with the contents below, and pressing SUBMIT. Please write a README.txt file which mentions any online sources you used to help with the project, as well as any notes about your project (i.e. if you couldn't get a particular part of the project to work). Create a zip file that contains p1.js, p1.html, any images you used, any other files necessary to fully view your puzzle, and the README.txt file. Then, name the file LASTNAME_FIRSTNAME_P#.zip. For example:

schwarz_julia_p1.zip

Once you have made this zip file, go to the assignment page on blackboard, attach this file, and submit.

Grading
Your assignment will be graded as follows:

Turn in is correct, code has no syntax errors	10 pts
Tiles are correctly created and positioned	10 pts
Shuffling is implemented correctly	5 pts
When tiles are clicked, they move to empty tile correctly	10 pts
Randomizing puzzle works correctly	5 pts
Bells and whistles	5 – 10 pts
Formatting, Comments and Coding style	5 pts