

Name: _____

ID: _____

Midterm

CS193C, Summer 2010, Young

Please print out this page, sign the Honor Code Statement. Turn in the Honor Code statement at next Tuesday's Lecture, or slip it under my office door (Gates 194). SCPD students may FAX it to (650) 723-6092.

As we've discussed, you have four hours to complete this exam and get it submitted. Please get it submitted by 11pm. Zip up your files and submit them using the dropbox on Coursework by the deadline.

This test is open book, open note. You may use the Internet to access reference material, but may **not** use it to communicate with anyone other than the CS193C teaching staff.

The Stanford Honor Code

1. The Honor Code is an undertaking of the students, individually and collectively:
 - a. that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;
 - b. that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.
2. The faculty on its part manifests its confidence in the honor of its students by refraining from proctoring examinations and from taking unusual and unreasonable precautions to prevent the forms of dishonesty mentioned above. The faculty will also avoid, as far as practicable, academic procedures that create temptations to violate the Honor Code.
3. While the faculty alone has the right and obligation to set academic requirements, the students and faculty will work together to establish optimal conditions for honorable academic work

I accept the letter and the spirit of the Stanford University Honor Code. I swear that I have not given or received unpermitted aid on this exam, and I have taken an active part in stopping any violations of the Honor Code which I see on this exam.

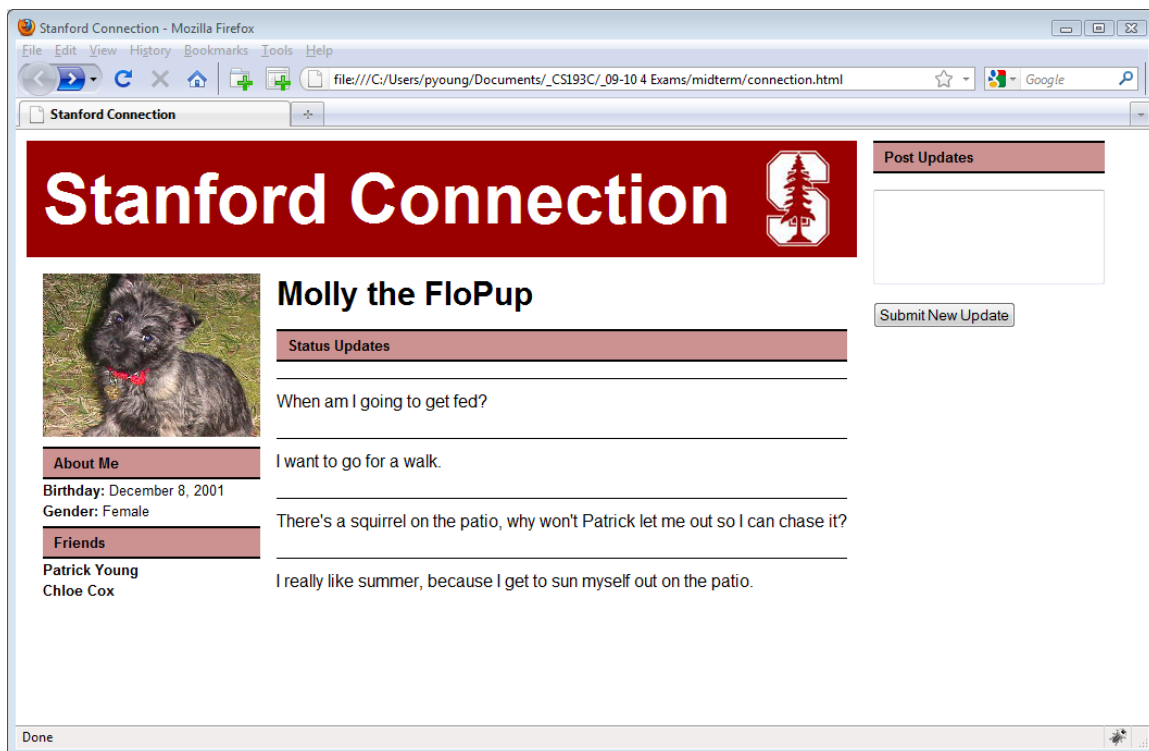
Signed: _____

Instructions

- **Make absolutely sure you turn everything in on time. Do not turn in your exam late.**
- Check the midterm directory for a zip file of images that you will need for this exam.
- **All problems should work on both Internet Explorer and Firefox.**
- If desired you may use the convenience innerHTML property along with any other techniques which are supported by both web browsers, even if they are not part of the official W3C standard.
- Make sure everything validates. You may use either XHTML Strict or Transitional.
- You may not use Dreamweaver or other WYSIWYG editors.
- As we discussed in class. All code must be developed from scratch – this means you may not copy directly from code you find on the Internet.
- Check your e-mail. If someone sends me something that I think everyone should know, I'll send you all an e-mail – hopefully this won't be necessary.
- There are only two problems on this exam since the first problem requires use of HTML/CSS layout anyway.
- If we do not provide values for items you need, any reasonable value is acceptable. For example you may make up your own alt values for images or set widths and heights as desired when they are not specified in the problem statement.
- Both these problems were inspired by suggestions from our TA Alex Bain. However, the final problem formulation, the writeup, and any errors or incorrect estimations on difficulty are entirely my doing.

Stanford Connection

For our first problem we will develop a “Facebook-like” webpage. For a real Facebook implementation you would need web server programming, so we are going to fake it a bit. Here's what the webpage should look like:



As you can see, the webpage consists of a photo, some information about the person, and a set of status updates. On the right, using form elements, the user is able to enter in new updates, which will be added to the status updates shown in the center.

Your first task is to reproduce the webpage as shown. You don't have to make a page for Molly, you can substitute your own information, but don't use any time thinking about this, we are only interested in the webpage layout and the accompanying JavaScript. Also if you enter your own material, make sure it stays at the PG-13 level or below.

While you do not need a pixel-perfect implementation you should include the following:

- The Stanford Connection banner which runs across the top of the left two columns (but not across the "Post Updates" column).
- As seen in the screenshot the banner should have the Stanford Logo, which is included with the midterm's downloads. Note that the actual "Stanford Connection" is text (not part of a graphic image) and it should remain text. You should not create any new graphics files for either problem in this midterm. The color in the logo is #9A0000.
- A column on the left consisting of a picture, followed by information about the user.
- Two sections below the picture, one for "About Me" and one for "Friends".
- Titles in the "About Me" section like "Birthday" should be bolded, while actual values such as December 8, 2001 are not bolded. Friends' names are bolded.
- Each of the sections "About Me", "Friends", "Status Updates", and "Post Updates" should have a colored background – I used the color #CC9191. They should also have a 2-pixel black border on the top and bottom but no border on the sides.
- Status updates should be separated by thin lines as shown.
- The right column should include a "Post Updates" heading, a text area, and a submission button.
- Leave a reasonable amount of space between columns.

The webpage should be laid out using CSS. Do not use HTML tables for layout on this problem.

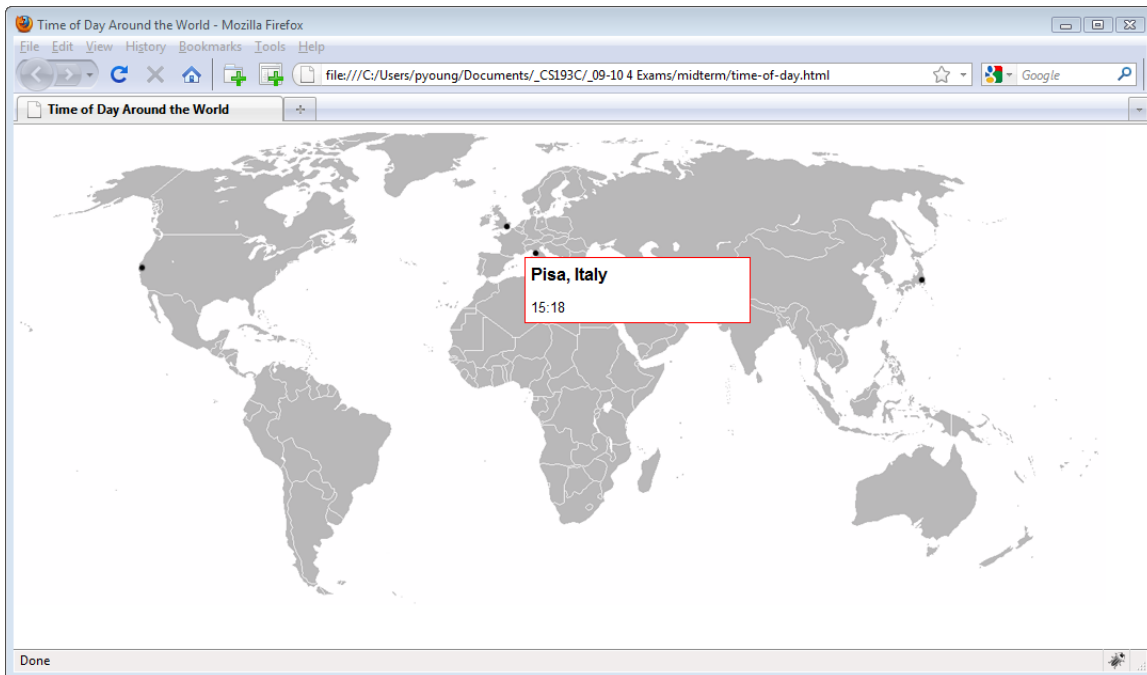
Once you've got the webpage appropriately laid out, write some JavaScript so that if you enter text into the text area in the right column and click on the "Submit New Update" button, the text gets added as another status update in the middle column.

Warning: the next problem is probably quite a bit harder – don't use all your time on this problem. If you're having trouble with the CSS layout, put something together that's mostly right, then move on to the next problem. You can come back and fix your CSS layout on this one if you still have time after completing the next problem.

Time of Day

For this problem you will develop a map which displays the time of day in different cities around the world. The time of day will be displayed in a popup. To keep things simple you may display time using a 24-hour clock format (i.e., 20:15 instead of 8:15pm) – you can use the traditional method if you're more comfortable with it, but it will probably be a bit more work.

Here is a screenshot of the webpage in action:



As you can see the webpage simply consists of a big map image. This image is provided for you as the `world.png` file.

The program has the following features:

- When the x and y coordinates of the cursor are both within 10 pixels of one of the cities on the map a popup will display, showing the name of the city and the current time in that city.
- The popup should appear next to the city. You can use a preset location for each city's popup, or you can place the popup so that its top-left corner is at or near the mouse location which triggers the popup.
- The popup will remain visible until the cursor moves off of the popup.
- The popup should not move while it is visible. After the cursor moves off of the popup and it disappears, it may of course appear in a different location – for example if the user moves the cursor on top of a different city.
- You do not have to update the time while the popup is visible. However, the time should be updated to the current time the next time it appears.

We will only support four cities for this problem. However, store the city information in an array. We should be able to add more cities by simply adding more dots on the map in the image file and adding additional data items to your array, without further changes to either your JavaScript or the HTML

Here is the city information you will need:

Stanford, California: located at (121,135) on the map image, time difference from UTC¹ = -7

London, United Kingdom: (463, 96) on the map image, time difference from UTC = +1

Pisa, Italy: (490,121) on the map image, time difference from UTC = +2

Tokyo, Japan: (852, 145) on the map image, time difference from UTC = +9

To determine the current time, you can create a Date object. If you do not provide any parameters when constructing it, you will get a Date object corresponding to the current data and time. To learn more about the JavaScript Date object do a Google search or try visiting this webpage:

http://www.w3schools.com/jsref/jsref_obj_date.asp

If you are at a loss on how to start out, I recommend building this problem in stages. Here are the steps I took in creating my solution:

- First I made sure I was getting coordinates properly in my event handlers.
- Next I had the event handler display the popup with static information (i.e., the same information regardless of the city, and without looking up the correct time).
- Then I had the event handler display the popup only when the mouse coordinates were in the appropriate coordinate range.
- I then modified the popup to display the city name.
- I added the current time in UTC to the popup.
- I modified the time in UTC to the time appropriate for the city.
- I next changed the location of the popup to appear over the city.
- Finally I set the code so that the popup would go away when the cursor moved off of the popup.

¹ Times zones are relative to UTC or Coordinated Universal Time (also known as Greenwich Mean Time). This is the time system the JavaScript Date object will use. If you're curious as to why London is not UTC = 0, it's because London is currently in BST (British Summer Time) similar to our own Daylight Savings Time.