For homework:

1) Add some more helper functions to the pcs tools dom helper (jQuery) like we wrote in class:

a) Add a function to cause the element to change colors (like the homework from a few classes ago where you flashed background colors on an interval). Allow the user to pass in the length of time to flash the element for, and maybe also an optional number to control the speed of the flashing

2) Create an HTML Clock.

a) Use setInterval to create an HTML "clock" that shows seconds, minutes, and hours. Doesn't have to keep correct time, it can start at 12:00:00 and start ticking at around once per second  - but if you do want it to be real you can try something like:

Let d = new Date();

hours = d.getHours();

minutes = d.getMinutes();

seconds = d.getSeconds();

or maybe simpler just use d.toLocaleTimeString()

(If not keeping real time, you can track hours, minutes, seconds like we did R, G, and B, in the colors example in class. That might be interesting to do as an exercise even if you ultimately make it so that the time is correct)

b) Turn it into a "module" that can be used simply by loading the script file. i.e. Do what we did with message box where instead of you having to have the html pre written, we created the elements for you.

One way to do this would be to make it so that user of our clock does nothing but load the script and magically gets a clock on the page. No code required. Just load the script and a self-invoking function runs and adds the clock.

Another approach would be that the user (developer user, not end user) has to request a clock and then gets a clock added, so they can ask for 3 clocks for example and get 3 clocks on the page (more like our message boxes, where you ask for one and it gets created). Maybe do it both ways :)

For now, clocks can get added at bottom of page (just like the message boxes do). In class we will look at ways to allow the caller to decide where in the page they want the clock to go.

Extra exercise for those that have the time to do it:

Add function(s) to pcs tools to allow the user to associate arbitrary data with an element. e.g myWrappedElement.data('foo", 5) will store the key value pair foo = 5 for this element, then later you can call myWrappedElement.data("foo") and get back 5. Calling myOtherWrappedElement.data('bar", 'something else') and then later myOtherWrappedElement.data("bar") would get back 'something else'.