# *Web Programming V (420-C50-HR)*

# *Lab 1 – Review*

Date assigned: Wednesday, August 30, 2017

Date due: **Wednesday, August 30, 6:00 p.m.**

**Learning Objectives**

Upon successful completion of this lab exercise, the student will have:

* Studied sample web pages to reverse engineer CSS using the browser’s “View Page Source”
* Build a web page with HTML and CSS

Lab Setup

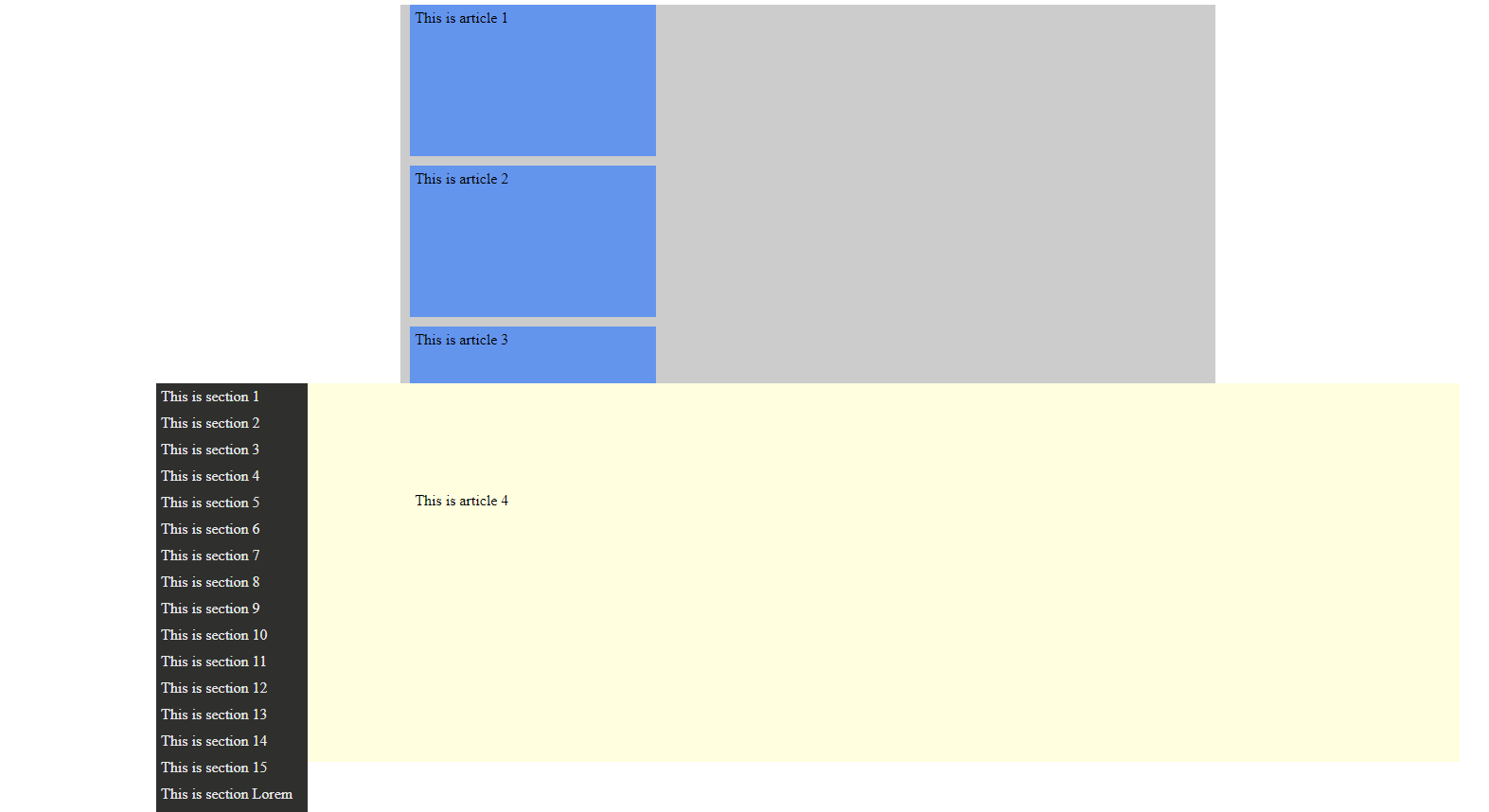
1. You are to use Chrome or FireFox for this exercise. Some of the features will NOT work in IE.
2. You can use whatever environment you want from Notepad++ to Dreamweaver to WebStorm to PHPStorm to Visual Studio. At this point you get to decide.

To do:

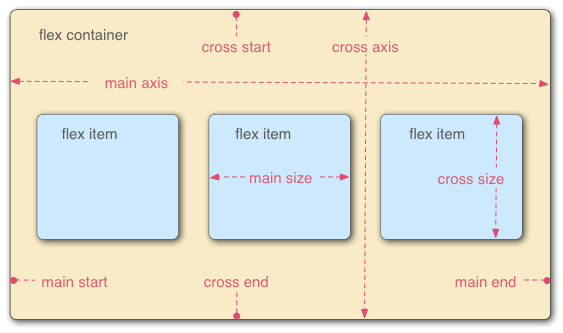
**Part A – CSS and HTML**

This section you will review HTML and CSS and maybe learn a couple of new things.

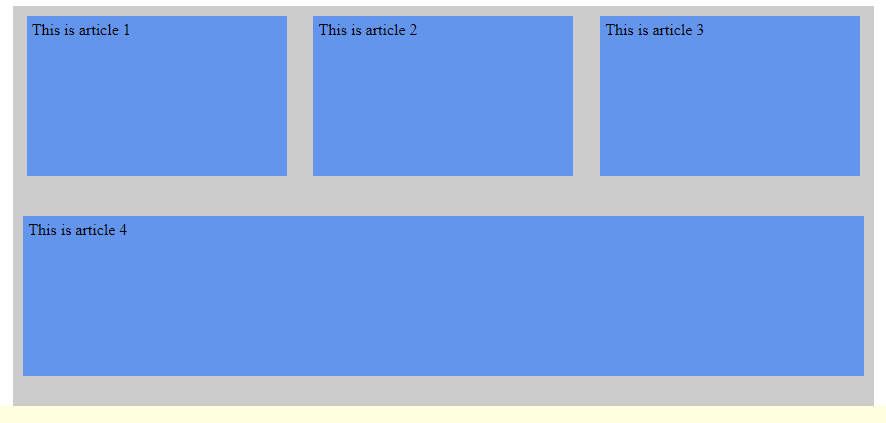
1. In the partA folder of the files you copied for the lab is a file called parta.html. Open it in your favourite editor if you like but YOU CANNOT MAKE CHANGES TO THE HTML FILE. Linked to it is a css file in the styles folder called parta.css. At this point the top part of the page looks like this when viewed in a browser:



1. While this is attractive, it’s lacking a certain something. We are going to change it around using flex boxes. Flex boxes have the following properties:



1. Start by changing the display of the container1 div to flex (display: flex). That makes this div a flex container and every child in it a potential flex item. Now if you display it, you should see that the articles are squished in a single row in the container.
2. Add a flex-wrap property with the value wrap to make this a little better, but here the articles in each row are still right together. This is fixed with the justify-content property. Try setting it to space-between and then space-around to see the difference. Choose one to use (my example below has space around). With space around you should end up with the 4th article in a separate row.
3. Almost, there. Now use the appropriate selector to choose the last article. In this selector, make the flex property 3 so that the box will expand to 3 times the size of the other boxes (if there is space). The top of the page looks like this now:



1. Use flex box properties to set the container2 div as flex container (display) that wraps (flex-wrap) by column (flex-direction) and align the items to the center (align-items).
2. Change the direction of the layout by adding a flex-direction property of column-reverse to the second container.
3. We want each section to take up 20% of the space if it can (it cannot by the way) so set the flex property of the section element to be 1 20%. The 1 keeps them all the same size and the 20% says to be that size if you can.
4. Lastly, select the first element of type section and have it align itself to the start of the flex item (align-self: flex-start) and choose the 11th item (nth-of-type(11)) and align it to the end of the flex item.
5. The final page looks like this:



**Part B – JavaScript**

After class yesterday it seemed like some JavaScript review might be in order. How can you not love this language???

1. Create a web page with the name review\_js.html which links a css file in the subfolder called styles. The CSS file is called review\_js.css. Also link a JavaScript file called review\_js.js Give the page the title “Review JavaScript”
   1. Create a form on the page
   2. Add the field firstName to the form. Give it the label “Name:”. Make the field 50 characters wide and able to accept 50 characters.
   3. Add a field myAge to the form. Give it the label “Age:”. Make the field 4 characters wide and able to accept 2 characters.
   4. Add a field emailAddr to the form. Give it the label “Email Address:”. Make the field 60 characters wide and able to accept 60 characters.
   5. Add a button to the form with the text of the button “Go For It”
   6. Add CSS to format the page and form so that:
      1. There is an container div called container that is 80% of the screen width and centred
      2. All of the labels are ri ght aligned (they line up on the right side).
      3. The text fields are left aligned
      4. The button is aligned with the text fields
   7. Add an event listener so that a function is called when the form is submitted (the button is pressed). This can be an anonymous or named function it is up to you. I think you will find it easier with an anonymous function.
   8. Add the JavaScript function called by the event listener which validates that all the fields are non-blank. If any field is blank, display a message in a div named “errorList” which is BELOW the form. The message should say something like, “Please fill in all the fields”. Best practices would be to put out a different message for each field in error, but only one error message in total. If there is an error, display the error message(s), change the background colour of the form to grey (#ccc) and prevent the form from submitting. If all fields have a value, allow the form to be submitted (the default action) to the file FormProcessor.html. FormProcessor.html can be found in the partb folder you downloaded at the start of the lab. FormProcessor.html must be called using a query string which includes the names and values of the variables.

**Part C – Visual Studio**

Review of .NET and Visual Studio. How can you not hate this environment???

1. Using Visual Studio, create an ASP .NET project using C# that creates a form
2. Create a page with a form with 3 fields: first name, age and email address. Also have a button labeled Register.
3. Add appropriate (the correct asp.net ones) validation controls as follows:
4. Each field is required.
5. The name can only contain upper and lower case letters.
6. The age must be a number less than 100.
7. The email address is a valid email address.

Remember: before adding any validators, there are a couple of lines to be added to the web.config file. This allows jQuery to run the validations on the query mode (you can copy these lines).

<appSettings>

<add key="ValidationSettings:UnobtrusiveValidationMode" value="None" />

</appSettings>

1. Use a validation summary to display all the error messages for all fields nicely formatted ABOVE the form. Add a red asterisk to the right of the field in error
2. When all fields are valid display the following message BELOW the form:

Hello *name* You are currently *age* years old. You will turn 100 in *calc* years. I will email this report to *email address*.

1. Use good design techniques for the form including field labels, sizes, alignment and colours. Add things like headers to the form and a title to the page.

**To submit**

When you have completed the lab exercise, call me over to check your work. Then create a zip file of the folder and load it to the Moodle page for this course. Your zip file should have the name format: *uname*\_C50L01Review.zip where *uname* is your short user name (i.e. amcdonald for me, jsmith for John Smith, etc)