ITEC 136 Lab 2 – 2010 Winter

Purpose

To assess your ability to apply the knowledge and skills developed in Module 3 and Module 4. Emphasis will be placed on the following learning outcomes:

1. Decompose a problem into modularized components.
2. Write and call functions that utilize parameters and return values.
3. Properly pass HTML id values to JavaScript to allow read/write assess of HTML elements.
4. Write correct conditional statements to solve a given problem.

Assignment

Develop a tax program in HTML and JavaScript for a given number of tax filers. The program should accept the input from 4 HTML input tags of type="text" each containing the taxable income of a person or family. Presume all input income is in whole dollars. All calculated/displayed tax should fixed to 2 decimal places of precision. All output should be below the calculate button and the horizontal rule (<hr />) either using approach outlined at the end of this lab or a HTML <textarea> tag. I recommend the using the output system as I have posted several examples to make this easier.

- **Use at least two functions. The first function should be called from the HTML file in the onclick attribute of the button <input> using four parameters. The parameters should be the HTML ID values for each of the four income text <input> tags. The second function should calculate and return the tax using one parameter - a given taxable income. Call the second function four times within the first function, once for each of the four values input by the user.** Hint: Don't forget to use the document.getElementById() function to resolve the ID parameters into the user entered values.
- **No significant JavaScript embedded in the HTML.** Only a call to your main function should be in the onclick attribute of the button should be needed. All other JavaScript should be in an external file with a ".js" extension. Remember to put the HTML and JS file in the same folder (directory) so that the JavaScript is found without additional efforts.
- **Do not use global variables to avoid using parameters.** Remember that any variable declared using *var* outside of a function is a global variable. Occasionally global values are necessary, but not for this lab and are forbidden. Many times global variables are misused to avoid parameters and returning values from functions.
- Try to make your solution look as close to the screen shots below as possible.
- Put your name, class and assignment number in all text based files either visible or in comments, including HTML and JavaScript.
- Put all files in a single ZIP file for the dropbox.
- No need for any Word file as the first lab required. Your *well-commented* HTM and JS file should be sufficient.
- Review code against the documentation and style requirements Word document available on the course web site for the assignment. You will find that both documentation and style requirements are facilitated by the Aptana Integrated Development Environment (IDE). For example, when you create a new untitled JavaScript file, by default a comment block is generated, possibly with your log in name. You should document all your functions with the purpose and each parameter(s) usage.
- No need to validate your code with the W3C validator, http://validator.w3.org/.

Use the information below to determine the tax based upon the based upon the taxable income.

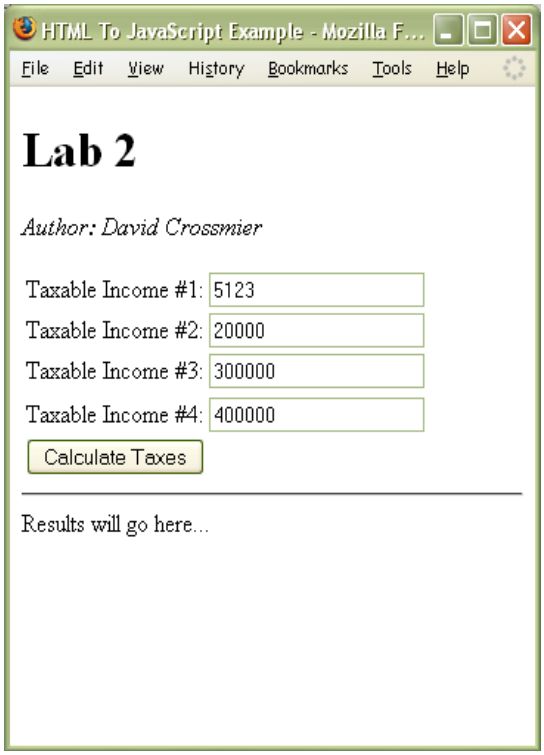
If taxable income is over--	**But not over--**	**The tax is:**
$0	$7,825	10% of the amount over $0
$7,825	$31,850	$782.50 plus 15% of the amount over 7,825
$31,850	$77,100	$4,386.25 plus 25% of the amount over 31,850
$77,100	$160,850	$15,698.75 plus 28% of the amount over 77,100
$160,850	$349,700	$39,148.75 plus 33% of the amount over 160,850
$349,700	no limit	$101,469.25 plus 35% of the amount over 349,700

As an example, the table below shows the taxes for the several taxable incomes. **Use the values in the first column as the default values in your HTM page.**

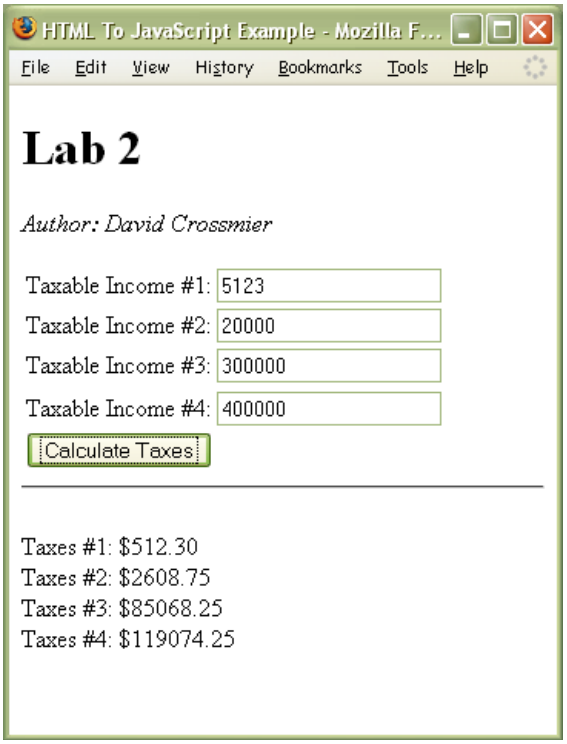
Taxable Income	Taxes to calculate in JavaScript
5123	$512.30
20000	$2608.75
300000	$85068.25
400000	$119074.25

Screen shots of my application in action

Initial Load



Pushed Calculate Taxes button.



Helpful Hints

The following hints may help you to solve the problem:

1. Consider the example framework I posted as a starting point for this lab. Several files are posted to the FTP site at: http://cs.franklin.edu/~crossmid/2010-Wi-ITEC136/Examples/

    - framework.htm – sample framework HTML file to fetch/set values from HTML. HTML display of all output; everything starts here.
    - code.js – sample framework of JavaScript to get/set HTML values.
    - get.js – utility code to make access to the document.getElementById function easier.

2. Use the HTML div manipulation functions to allow you to easily create your output for your lab. Use an HTML layer to your HTML. This will allow you a re-writable "surface" on the HTML document other than document.writeln(). There are 3 helper functions in div.js on the FTP site to allow easy manipulation of the HTML text in the . I have also posted two test files (divTest.htm and divTest.js) to show how to use div.js properly.

    - divTest.htm – test/example HTML file to manipulate HTML div tags. The onclick event attributes call functions in divTest.js.
    - divTest.js – test functions called from the HTML file. Demonstrates how to use the functions in "div.js".
    - div.js – utility functions to manipulate HTML div tags in JavaScript. This is the file to include in the <head> section of your HTML page. Then you can call any function in div.js in your JS file(s).

3. Although default values are provided, make sure you test with other values, especially the **"edge" values** between tax brackets. For example, $77,100 is an edge value for taxable income. The correct answer for this value is $15,698.75.

4. The toFixed() method of any Number object to convert a number into a string with a fixed number of decimal places. For example:

```
var myPI = 3.1415926535;
var shortPI = myPI.toFixed(2);  // puts 3.14 into shortPI
```