

# ASSIGNMENT 3

Version 21/FA

## MAPPINGS:

The following course objectives and/or outcomes are measured in this assignment:

### COURSE OBJECTIVES

- 3D: Declare and call/invoke JavaScript functions in applications that you develop.
- 3F: Use anonymous functions to handle events.

### COURSE OUTCOMES

- 1. Use JavaScript to create interactive web applications.
- 2. Write clean, consistent code.

## GRADING:

Task	Points	Criteria
<b>Part 1</b>		
3	5	Each line of the standard opening comment is worth 1 point. Subtract 1 point per missing item.
4	5	Pass/Fail
5.a.	10	If the arrow function is coded correctly and returns the required information, full credit. If the student wrote a function using function syntax instead, award a maximum of 5 points. Deduct points if errors were made with the rest of the requirement.
5.b.	5	Pass/Fail
5.b.i.	5	Pass/Fail
5.b.ii.	8	Pass/Fail – 2 points for each correctly calculated tax value. You may subtract 1 point for each tax calculation if the task was completed but with errors (instructor discretion).
5.c.	2	Pass/Fail
5.d.	2	Pass/Fail
<b>Part 2</b>		
2	5	Pass/Fail
7.a.	2	Pass/Fail
7.b.	2	Pass/Fail
<b>Total</b>	<b>51</b>	

Penalties
Deduct 50% from entire assignment for the use of the <b>var</b> keyword in variable declarations.
Deduct 60% from the entire assignment if JavaScript is inline/embedded instead of external.
Deduct a maximum of 5% for code that does not comply with the course <i>Style Guide</i> and/or which is messy/unorganized, uncommented, or missing semicolons.

## TASK:

### TEST THE APPLICATION

1. Review the course *JavaScript Style Guide* before starting this assignment. Part of the assignment will be graded on your adherence to the *Style Guide*.
2. Download **assignment03\_starter.zip** from the *Module 3: Assignment* drop box in Canvas. The file is located beneath the heading *Assignment Resources*.
3. Extract **assignment03\_starter.zip**. The file contains two files:
  - a. a single HTML document named *index.html*.
  - b. a single JavaScript file named *script.js*.
4. Double-click **index.html** to launch it in your browser; examine the application's behavior.
  - a. The page will open, and the content shown in Figure 1 should appear.
  - b. Click on **Calculate Taxes**. Nothing will happen.

# Tax Me!

Monthly Salary:

Figure 1: Starter version of index.html

*Continues...*

## MODIFY THE APPLICATION

### Part 1

1. **script.js** has already been linked to **index.html** for you.
2. Open **script.js**.
3. Add the **Standard Opening Comment** to the top of the script (**5 points**).
4. Add "use strict" to the very top of the JavaScript file (above the **//DO NOT MODIFY THIS CODE** comment. (**5 points**)
5. Beneath the comment **//YOUR CODE GOES BELOW** accomplish the following:
  - a. Use the **arrow** syntax to define a function named **\$** (**10 points**)
    - i. This function should accept one parameter, which represents a “selector” and it should return the return value of **document.querySelector(x)** – where *x* is the name of the parameter representing the “selector.”
    - ii. If you cannot use the **arrow** syntax, use the **function declaration** syntax (**-5 points**).
  - b. Define an event handler that responds to the **click** event generated by the **input** element with the **id** attribute value of **calculateTaxes**. (**5 points**)
    - i. The event handler should obtain the value in the text input field with the **id** attribute value of **monthlySalary**, which is defined in **index.html** (**5 points**)
    - ii. Then, the event handler should calculate the federal, state, social security, and Medicare tax rates by invoking the **calculateTax** function. Use the following tax rates for this exercise (these aren't real): (**8 points**)
      1. Federal Taxes: 12%
      2. State Taxes: 5%
      3. Social Security Taxes: 6%
      4. Medicare: 1.5%
  - c. Calculate the **net** salary by subtracting each of the resulting tax values from the original **monnthlySalary**. (**2 points**).
  - d. Invoke the **print** function. (**2 points**).

*Continues...*



- e. Save all open files. Then, launch **index.html** and your browser and test by inputting a salary value of **50** and clicking **Calculate Taxes**
- i. Figure 2 (below) shows sample output for the gross wage value of **\$50**

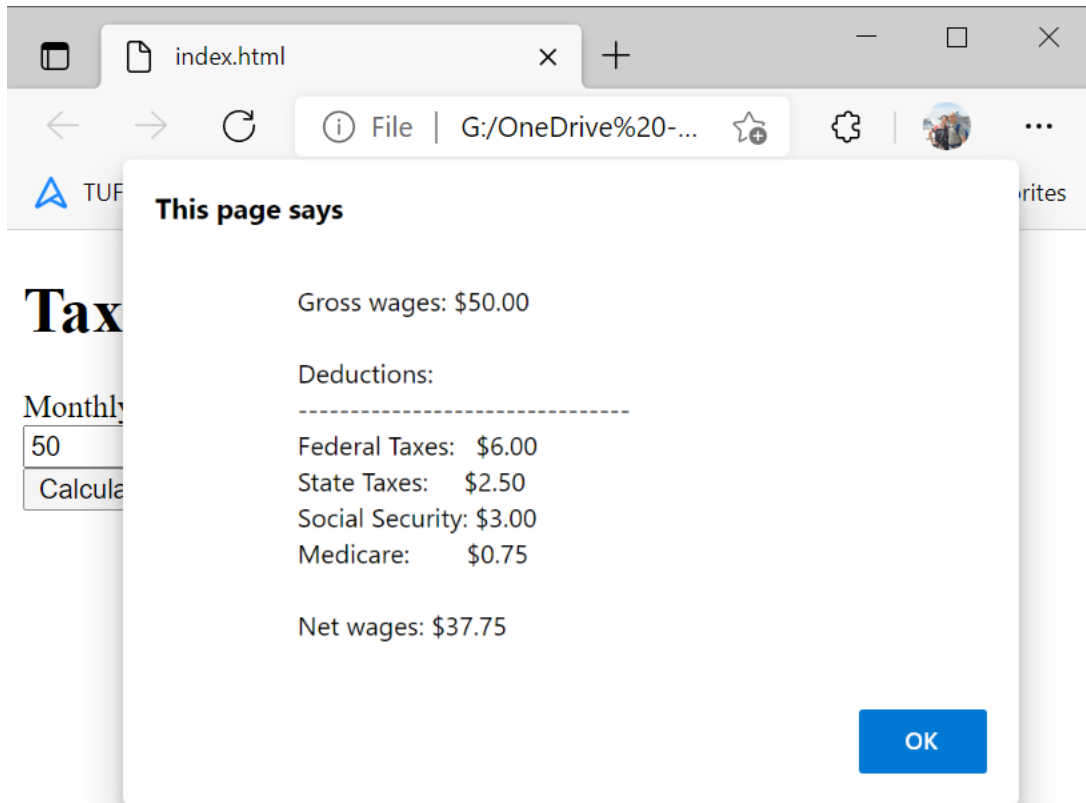


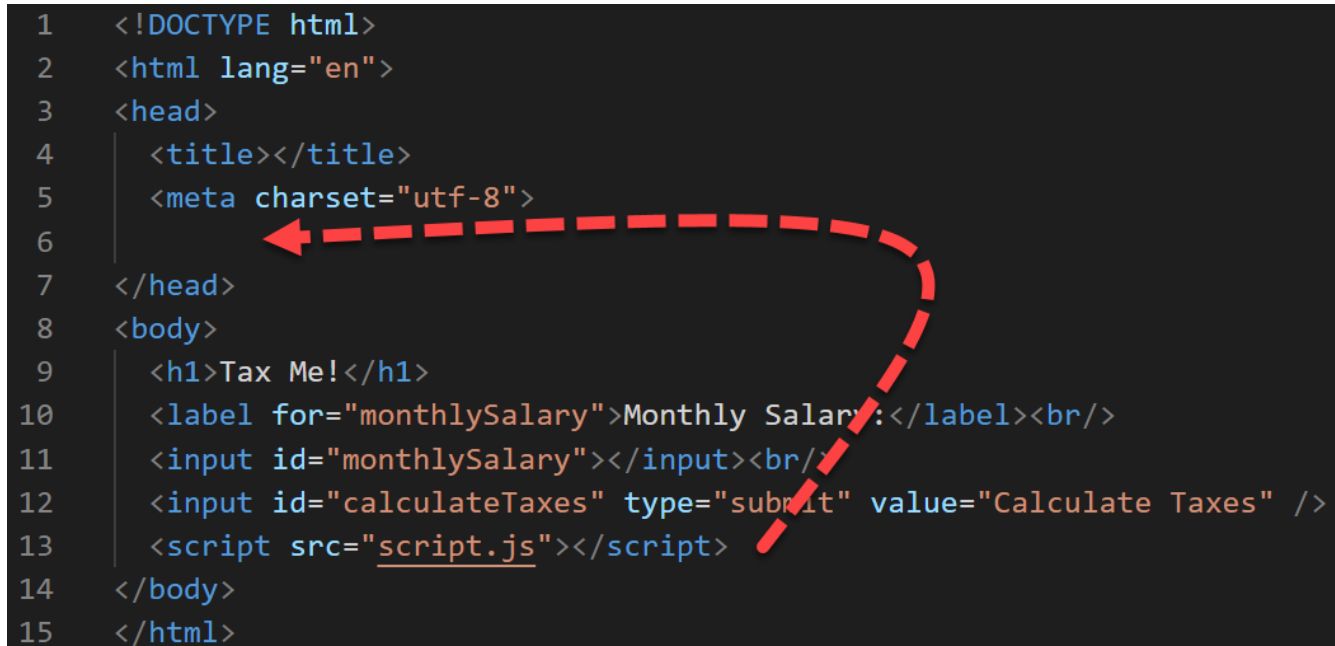
Figure 2

*Continues...*



**Part 2**

1. Open **index.html**.
2. **Move** the existing `<script>` tag from line 13 to line 6 (Figure 3). (5 points).



```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <title></title>
5    <meta charset="utf-8">
6
7  </head>
8  <body>
9    <h1>Tax Me!</h1>
10   <label for="monthlySalary">Monthly Salary:</label><br/>
11   <input id="monthlySalary"></input><br/>
12   <input id="calculateTaxes" type="submit" value="Calculate Taxes" />
13   <script src="script.js"></script>
14 </body>
15 </html>
```

Figure 3

3. The result should look like Figure 4.



```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <title></title>
5    <meta charset="utf-8">
6    <script src="script.js"></script>
7  </head>
8  <body>
9    <h1>Tax Me!</h1>
10   <label for="monthlySalary">Monthly Salary:</label><br/>
11   <input id="monthlySalary"></input><br/>
12   <input id="calculateTaxes" type="submit" value="Calculate Taxes" />
13 </body>
14 </html>
```

Figure 4



4. Remember, JavaScript will execute as soon as it's loaded into the browser. Up to this point, we have placed our JavaScript at the bottom of our HTML document to ensure all page elements were loaded before the JavaScript loaded and executed.
5. By moving our `<script>` tag from the bottom of the HTML document to the top of the HTML document, we're going to create problems, because our script will now execute before all page elements have loaded.
6. Save all open files (**index.html** and **script.js**). Then, launch **index.html** and your browser and test by inputting a salary value of **50** and clicking **Calculate Taxes**.
  - a. Nothing should happen (if the prompt appears like it did in Part 1, make sure all files have been saved).
  - b. Opening the browser console should show something like what's shown in Figure 5 (output may differ depending on your browser). The error in Figure 5 appeared because the **`addEventListener()`** method was called on an element *before* the element loaded in the browser (since we moved **`script.js`**).

## Tax Me!

Monthly Salary:


Figure 5

*Continues...*

7. Refactor **script.js** to accomplish the following:
  - a. Add an event listener to handle the **document** object's **DOMContentLoaded** event. (2 points)
  - b. Move code that must wait to execute until the HTML has been loaded into the browser and the DOM is ready into the event handler for the **DOMContentLoaded** event. (2 points)
8. Once done, save all open files (**index.html** and **script.js**). Then, launch **index.html** and your browser and test by inputting a salary value of **50** and clicking **Calculate Taxes**.
  - a. An example of this behavior is shown in Figure 6.

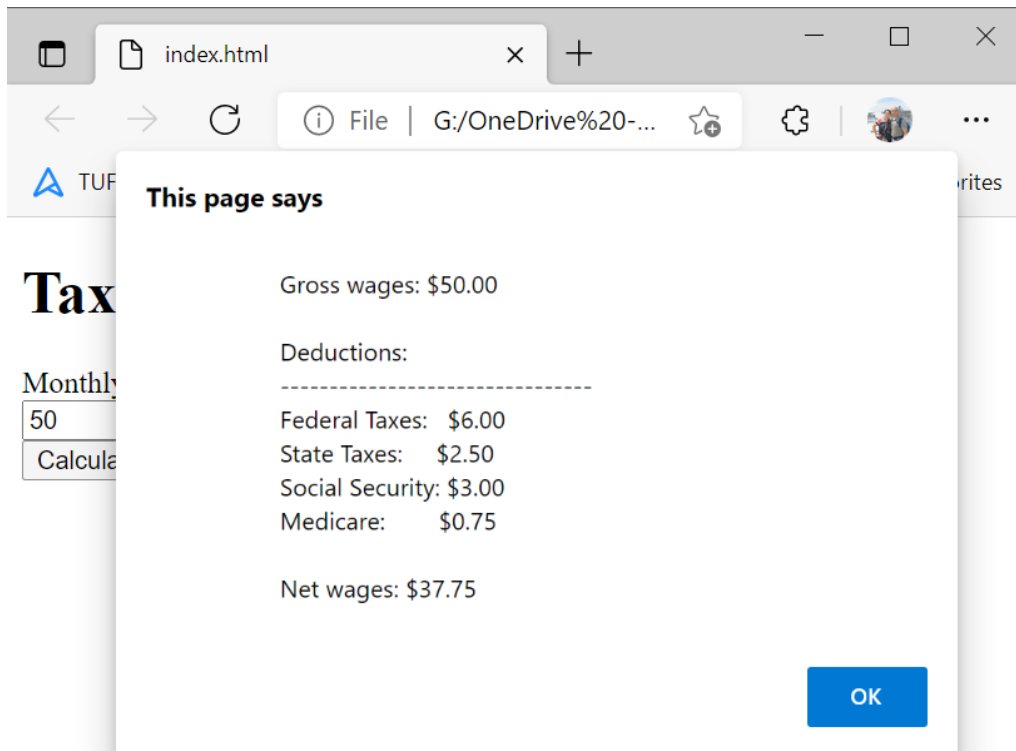


Figure 6

## SUBMISSION

When complete, create a single **ZIP** file containing your solution. The ZIP file should contain all files included in the original starter code or added as part of this assignment.

**NOTE: Canvas is configured to only accept ZIP files; it will not accept ZIPx, 7ZIP, pZip, RAR, etc.**

Upload and submit Assignment 3. Then, have some [Sanka!](#)

*End Assignment.*