3 - Asynchronous Programming

1. Learning Outcomes

On completion of this lab you will have:

- Implement an interactive user interface in JS using event handling
- Process JS objects and arrays (from JSON) using the functional style
- Implement an AJAX request function
- Read and process remote JSON documents using promises

2. Organisation

Please complete the exercises individually.

3. Grading

This worksheet is worth up to 10% of your overall module grade. You must attend and sign in at the labs in order to obtain the credit for the associated worksheets. You may work on this worksheet during lab 4 and lab 5 with instructor assistance. You must also demonstrate your submission in order to receive credit - see below.

4. Submission

The deadline for submission is Wednesday Oct 26, 2016 @23:59 through Webcourses.

5. Demonstration

You will give a brief demonstration of your submission to the lab instructor in lab 5.

6. Requirements

For this lab you will need to

- Use your own laptop with local tools or,
- Sign up for a free account with JS Bin (http://jsbin.com). It is **strongly recommended** to also have free account with Github (http://github.com) which you can use to authenticate to JS Bin. This allows you to publish Gists directly from your JS Bin workspaces which can be useful if you want to post questions or have a discussion regarding a problem or exercise in your module labs.

7. Resources

You are free to research whatever you need to solve the problems in this lab. Some recommended resources include:

- https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model
- https://developer.mozilla.org/en-US/docs/Web/JavaScript
- https://www.codecademy.com/learn/javascript

8. Problem Sets

Provide Javascript ES6 code for the following problems using JS Bin or your own development environment as you prefer.

When submitting, clearly identify what code solutions are associated with which problems.

Note: Keep your JS Bin URLs private to you. Do not share these with class colleagues.

1	Add button and keyboard handling logic in Javascript for the calculator front-end provided in the following JS Bin. http://jsbin.com/gataru/edit Your solution should not use third-party libraries such as JQuery. Use only the native API - HTML5 or JS whichever you prefer. The final solution should implement a fully working calculator which will allow a user to use the mouse or keyboard to enter expressions and have them calculated and displayed in the front-end provided **Note**: You may edit the provided HTML/CSS code if you wish	
2	For this problem refer to the lecture on array operations and consult the documentation at: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array Create a JS data structure from JSON document here: http://jsonplaceholder.typicode.com/users Using this data structure and the functional programming style of array operations, provide code to solve for the following:	

	• Build an array of all user name strings
	• Extract an array of geo objects having the form
	`{ lat: -68.6102, -47.0653 }`
	• For each user, build an array having (in the order given) the following properties
	  ○ Name
	  ○ Id
	  ○ Company name
	  ○ Zipcode
	• Build an array of address street names for which the zipcode starts with the digit 5
	• Get the product of all of the user ids	

3	Implement a version of the `fetch` API call from first principles using the native JS APIs only.
	
	Your function signature should look like the following:
	
	`fetch(url, options)`
	
	where options is an object of the form:
	
	`{`
	`  method: string`
	`  headers: {`
	`  name: string,`
	`  ...`
	`  }`
	`}`
	
	Your function should return a native promise which implements the usual reject/resolve handling	

4	Using the fetch function you have developed in problem 3, read and parse the document from the JSON document here: http://jsonplaceholder.typicode.com/todos Using this data structure and the functional programming style of array operations, provide code to solve for the following: • How many todos are there for userId 10? • Build an array of userIds, sorted in order of most uncompleted todos	
5	Using the fetch function you have developed in problem 3, read from the following non-existent resource and provide an error handler which will be called	

	by your implementation of fetch http://www.example.com/todos	