COMP 2537 – Web Development 2

Assignment 1

BCIT CST — Due: May 2nd, 11:59 PM

Late submissions result in zero

Introduction

You are going to take advantage of some more JavaScript features – based on what we've covered this week. You will use JS6 classes in your mini-component.

Details

You will create a client-side script and a server-side script – both in JS. The server-side will be run within the Node.js runtime. Client-side JS will, as usual, run in the browser. The details are:

1. All JS must be in separate JS files that are references by one HTML document
2. All CSS must be in separate CSS style sheet files and referenced by one HTML document
3. The HTML document is to be updated via AJAX

Overall Details

As far as context goes, you will create a component that provides a timeline updater. The content and context is up to you to choose. Perhaps it should be based upon what you are doing in your COMP 2800 projects course … 🤔.

This is a newsfeed component. It's meant to be able to be plugged into any app that you are developing on, so make it tightly cohesive and loosely coupled. That said, as web developers we don't want to re-invent the wheel. So no requiring any CSS or JS libraries for this other than what is being presented in this document.

This component will sit inside of a div element and will be resizable (both width and height – so liquid all the way for this). This newsfeed component allows for up to 10 items to be in it. Once it is full and a new item is added to it, the oldest item is removed.

For each item in this timeline, you must include the following pieces of information:

1. A title
2. The body of text
3. A date when the item was posted

You are free to lay out these sub-content pieces any way you wish, but they must be found within each newsfeed item.

Client-side Details

The client will use setInterval in JS to contact the Node.js server and request a new newsfeed item and add it to the newsfeed list. Again, the newsfeed can only hold up to 10 items so if there are already 10 news items, the oldest item must be removed. We could write this ourselves. But why when, chances are, there's someone out there who has done it for us! Just remember to give appropriate credit for such things. We'll be using some code from Ben Nadel, Co-founder of @InVisionApp, Inc. InVision is a prototyping app designed to help UX designers quickly prototype. His code is found at:

https://gist.github.com/bennadel/9760671

Download it and look at how it works. The two most important lines of code to call this code up are:

```
let friends = FixedQueue( 3, [ "Sarah", "Kit" ] );
// Add 2 items to the head.
friends.unshift( "Tricia", "Anna" );
```

Use this for your queue on the client side and update when you get a new newsfeed item from the Node server.

Server-side Details

The Node server will have two paths that serve up content:

- The HTML document, at path '/'
- A random newsfeed item that will be sent from a pool of hard coded newsfeed items, at path '/newsfeed-update'

Both are HTTP GETs. Your Node.js server app must have at least 30 newsfeed items. Each time that path is accessed, the server will use a random number generator to figure out which item from the pool of at least 30 newsfeed items to choose from.

Submission Requirements

Zip up your submission and upload to the COMP 2537 learning hub "Assignment 1" folder.

Grading Scheme

Your assignment will be marked out of 20:

- 10 marks for your client:
    - 1 marks for the ajax call
    - 1 marks for the CSS that styles it and makes it look pretty (yes, you are now actually getting marked on making things look pretty!)
    - 2 marks for calling the server and retrieving one class/object
    - 2 marks for displaying the news items
    - 4 marks for correctly updating the news items (10 item queue)
- 10 marks for your server:
    - 1 mark for returning the index.html file
    - 1 mark for static path mappings
    - 4 marks for 30+ newsfeed items
    - 2 marks for randomly choosing one of the 30+ items and sending it as an class/object

Create a class that represents your title, body text, and date. Classes/objects can be stringified easily:

```
class Customer {
  constructor() {
    this.name = “”;
  }
}

let c1 = new Customer()

console.log(JSON.stringify(myClass));
```

You will lose marks for:

- **Not using strict**
- Invalid code of any type
- Runtime errors
- Not following the requirements of the assignment
- Using variable names or file names that are very clearly example code or example files (e.g., lab7-starter.js)
- Content that is boilerplate (e.g., dummyimage images, lorem ipsum)

If you have questions pertaining to any design choices, please see your lab instructor or lecture instructor.

Submit naming format:

> Your zip file containing your source code files in a zip named COMP2537-assignment1-<lastname>-<firstname>.zip (only one zip file for all source code)