CS290 Summer 2017
Activity: Practice
Daniel Stoyer

Activity: Practice

Notes on extending the to-do list with Open Weather queries and dynamically changing content.

I copied the original to-do list code by hand to be sure that I understood each line and what it does.

I used an existing JavaScript file and modified it with the to-do js code.

I forgot to add the body-parser middleware and received an error when trying to post a name. The to-do list worked properly after adding the body-parser middleware.

Now that I have the to-do list web app working (confirmed adding and removing items works properly), I want to extend it to adding a city and minimum temperature for accomplishing the to-do, looking the city up in Open Weather, getting the city temp, checking against the provided minimum temp, then highlight the to-do in the list with a red or green background depending on if the temperature is warm enough for the to-do.

An aside, I really like using markdown for note-taking, it is a really simple formatting scheme. So, I found a markdown to pdf converter through npm. It can be installed from the command line with "npm install -g markdown-pdf".

Problem and Solution

Problem Statement:

- Make sure you can create sessions, add and update different kinds of data to sessions and end sessions.
- Make sure you can make HTTP requests, including POSTs, from the server and deal with the data you get back.
- Make sure that you can set up pages to accept data from forms.
- Make sure you can handle nested asynchronous requests and process errors if they crop up (force errors to happen by doing things like entering in the wrong URL).

Solution

- Extend the "to-do" web app by allowing users to:
    - add cities to-do items need to happen in
    - add min temp for to-do items
- Make HTTP calls to Open Weather Map to check if the current weather is good enough for the to-do.

    - if temp is too low, to-do should be red
    - if temp is right, to-do should be green.

Steps for approaching the problem:

1. Allow the user to add cities to the to-do list.

    - input field for the city name
      First I want to look at how the to-do items are currents entered. Then I want to see what the best format will be to enter a city name. After I have the basic UI in place, I will test entering the data on the client and getting the data on the server.
    - display the city name in the to-do list.
    - Should the city and temp be optional?

        - Need to think about how to handle "yes" or "no" to optional.

2. Allow the user to add min temp.

    - input field for the temp.
    - display the temp in the to-do list.

3. After getting the UI part down for city and temp, implement making the call to Open Weather Map.

Problems Encountered

- The input boxes in the forms are not aligned. This looks really bad. I would like to find out how to align the boxes, if I have time.

- The biggest challenge here is thinking through how to set up the asynchronous call to Open Weather.

    - Initially I'm trying to do the request like from the AJAX homework.

    - I've found that, while concepts are similar (asynchronous calls are needed), I need to use the Express/Handlebars way of doing things to make this work in the to-do app.

    - I've been circling around getting asynchronous calls to work properly. I've tried doing a request call in a helper function, but found that the page does not update with a new background color. I also tried putting the request in app.post(...) in the req.body['Add Item'] block, but received an error when trying to render.

    - I think that I might have to change the script to make a GET call with the Open Weather request first, then do a POST to the page.

        - Further introspection has me convinced that it shouldn't make a difference whether the call to Open Weather is in a GET or POST function and

- I have tried using node modules: async, then-request, sync-request (not good to use for client/server apps). I think I might have to go back to trying a hybrid of handlebars and a conventional ajax script for updating the css values.
- Instead of trying to use some sort of hybrid. I re-tried the approach of adding a request to the req.body['Add Item'] block of the app.post(...) function. My first attempt did not work because I was trying to render in both the Add Item block and at the end of the post function.
    - This was resolved by adding a return to the end of the Add Item block which allowed the request to get the information from Open Weather, render the page, and not continue in the post function to try to render again. I used a helper function to compare temperature values and set the style background color to 'red' or 'green'.

Snippet of app.post(...):

```
expressApp.post('/', function(req, res) {
    var context = {};

    if (req.body['New List']) {
        req.session.name = req.body.name;
        req.session.city = req.body.city;
        req.session.temp = req.body.temp;
        req.session.toDoList = [];
        req.session.curId = 0;
    }

    //If there is no session, go to the main page.
    if (!req.session.name) {
        res.render('newSession', context);
        return;
    }
    if (req.body['Add Item']) {
        request('http://api.openweathermap.org/data/2.5/weather?
q='+req.body.city+'&units=imperial&APPID=aa224681db2a563756dd2041bc0eb5ca',
                function(err, response, body) {
                    if (!err && response.statusCode < 400) {
                        var owmData = JSON.parse(body);
                        req.session.toDoList.push({
                            "name" : req.body.name,
                            "city" : req.body.city,
                            "temp" : req.body.temp,
                            "owmTemp" : owmData.main.temp,
                            "id" : req.session.curId
                        });
                        // Increase the id for the next to-do
                        req.session.curId++;
                        // the context needs to be updated with the session
information here because
                        // we want to render the page in this request block.
                        context.name = req.session.name;
                        context.city = req.session.city;
                        context.temp = req.session.temp;
                        context.toDoCount = req.session.toDoList.length;
                        context.toDoList = req.session.toDoList;
                        console.log(context.toDoList);
                        res.render('toDo', context);
                    } else {
                        if (response) {
                            console.log(response.statusCode);
                        }
                    next(err);
                }
            });
        // we make sure that the post does not continue because the page is
rendered in the request block.
        return;
    }

    if (req.body['Done']) {
        req.session.toDoList = req.session.toDoList.filter(function(e) {
            return e.id != req.body.id;
        })
    }

    // the POST was either for a new list, or to remove a to-do
    // we need to update the context and render the page.
    context.name = req.session.name;
    context.city = req.session.city;
    context.temp = req.session.temp;
    context.toDoCount = req.session.toDoList.length;
    context.toDoList = req.session.toDoList;
    console.log(context.toDoList);
    res.render('toDo', context);
});
```

I had to duplicate some code, which I would probably put into a function, but decided to leave as-is for now.

Snippet of the helper function in context:

```
var handleBars = require('express-handlebars').create({
    defaultLayout : 'main',
    helpers: {'getColor': function(minTemp, toDoTemp){
        if (minTemp < toDoTemp) {
            return 'green';
        } else {
            return 'red';
        }
    }}
});
```

Snippet of how the helper function is used:

```
<li style="background-color:{{getColor this.temp this.owmTemp}}">
```

To-Dos now highlight green or red, based on the minimum required to-do temperature compared to the city's current temperature.