Week 5 – How is your team functioning? And are you participating?

* Practice, Practice, Practice
* Portfolios
* Notes
* Let me help with examples you would like to see
* Debugging

# Practice, Practice, Practice

The scene a busy street in New York: A guy gets in a taxi with a violin case and says to the taxi driver, ‘What’s the fastest way to get to Carnegie Hall’? The taxi driver responds, ‘practice, practice, practice’!

If you are older than 50 maybe you’ll get the reference to the famous music hall in NY where the greats came to be honored and perform. If not, then just remember: ‘practice, practice, practice’.

I see that many are starting to enter in to new territory here. Especially with fat arrow functions, events, and import/export. Start with simple examples and build up from there. If you try and leap frog to too advanced code samples you’ll get lost trying to unwind 100 lines of code to find the 2 lines that are bad. Both of these code topics can be distilled down to very simple examples. Make sure you are good with them before climbing Mount Everest.

# Portfolios

The portfolios are developing well. I have some more examples for you this week. If you are stuck, review what your peers have done. Most examples are pretty straight forward.

I was very pleased to see the examples of scoring done with the tic tac toe game. Some of you did some clever work. I like to see the code announce a winner, I like it even more if I cannot play after a winner is declared and I am especially happy when row or column or a diagonal winner is declared.

If you are posting a URL for your assignment, make sure you click on the URL tab to submit the URL. This will allow me to just click on your link to see your great work in action.

# Notes

Over the last few weeks, I have shared student portfolios. Many show excellent note examples. Go back and look at their efforts. Here are a few more examples. Some submissions I get the notes are pages long. Others are 6 or 8 lines. Please do your future self a BIG favor and learn to take good notes. Create your own reference document for JavaScript and CSS. Minimal effort will be rewarded with minimal points.

|  |  |  |
| --- | --- | --- |
| Saman Tavasoli | <https://sam796.github.io/> | |
| Emily Wall | https://wallylime.github.io/web\_frontend\_development\_2/week4/week4\_notes.html | |
| Jesse Wilson | https://jesse-wilson88.github.io/wdd330/week04/ | |
| Ashley Zufelt | https://azufelt.github.io/wdd330/week4/index.html | |
| Daphne Museruka | https://daphneavril2604.github.io/WDD-330.github.io/week04/index.html | |
| Eugene Olsen | https://eolsenbyui.github.io/wdd330/week04/ | |
| Kimi Weldon | <https://kiadawel.github.io/WDD330KW/> |
| Amy Baker | <https://shakerbaker78.github.io/amy_baker_portfolio/> |

# Let me HELP!

What code samples would you like to see? Did the repos from last week help? I am happy to develop something new that might be of assistance. I can leave general notices each week, but I really would like to generate something for you if you have specific questions.

# Debugging

This week you will spend time on debugging. There ought to be a Bachelor of Science in Debugging. You could write your own ticket for a job. It is about the most important skill you will develop. It is easy to write code. It is not easy to write, safe, defensive, supportable code. If you need to be methodical and analytical designing and developing code you need that and more to get your code clean, tested and hardened. If you do not know how to use a debugger spend time on it this week and in the months and years to come. I can’t emphasize enough how critical this skill is.

Last week I had office hours for those interested in debugging. Here is the link to one of the sessions. We covered Debugging in VS Code, Safari and Chrome. I hope you find the video helpful.

<https://byui.zoom.us/rec/share/DSJ4IIBax2clAbnWI8NI8NL4Xo6XlpyGDioWP4y984_EN77iqXl8IBLR-U4wBGRG.vxHGygPy3Pyas9ZK>

Passcode: P\*Ft=+9S

# Imports and modules

How has your experience been with modules and imports? This might help. I love minimalist examples. This is about as few lines of code possible to demonstrate imports

modulesExample.html

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <title>Basic Math</title>  
</head>  
  
<body>  
<script type="module">  
 // this code works great  
 // first of all the script block specifies that it is a module.  
 // Hence it knows about cool stuff like import  
 import { sqRoot, getUrlVar } from './basicMath.js';  
 let ***num*** = getUrlVar('num'); // get the value of the URL param: **num**

if (typeof num === 'undefined') num = 137; // default to 137  
 let ***ans*** = sqRoot(***num***); // calculate the Square Root  
 ***console***.log(***ans***); // log to console  
 // push to the web page for fun

***document***.body.innerText = `The square root of ${***num***} is ${***ans***}`;

</script>  
<script>

// after you have seen that the sqRoot code below does not work.  
 // console.log(**num**); Uncomment this line  
 // you will see that the variable num is also not visible between script blocks either  
 // it is as if the script tag inserts an opening and closing set of curly braces around the code,  
 // thus scope blocking everything in the tag  
  
 // looking in the console you will see that this code does not work

// What are your thoughts on why that is?

// it seems like if we imported the file from above it would still be visible  
 // but that is not how modules work. The SCOPE of the imports is just in that script tag  
 let ***ans*** = sqRoot(441); // sqRoot is undefined. We cannot get past this point  
 ***console***.log(***ans***);  
 ***document***.body.innerText = `The square root of 441 is ${***ans***}`;  
</script>  
</body>  
  
</html>

basicMath.js

// basic utility function  
// return the square root of a number  
export function sqRoot (num) {  
 return ***Math***.sqrt(num);  
}  
  
// this is something fun I just picked up  
// My hate for alert messages knows no bounds  
// now I can add some input variables to the path URL to pass in to my code  
// ?num=100&name=Brigham Young University-Idaho&message=today is a good day!  
export function getUrlVar(pathVar) {  
 let vars = {};  
 ***window***.location.href.replace(/[?&]+([^=&]+)=([^&]\*)/gi,(m,key,value) => vars[key] = value);  
 return vars[pathVar];  
}

Above you see two script tags**. In the first script block** you have a working example of the import statement. The tag specifies the required **type=”module”**. Inside the tag you have the import of a function from basicMath.js and we use the sqRoot function as expected.

**But in the second script block** it all begins to fall apart. Your editor MIGHT show the color of the sqRoot function is a sad gray where it is a happy bright color in the top script block. (last semester that was the case with WebStorm, but I see now it shows a regular function color). Already the editor is communicating to you that it does not recognize the sqRoot function. That is because we are in a new block of code and the scope has changed. The import is limited to the block it is imported into. While your editor might not be showing this syntax color change the browser console will inform you that sqrRoot is not defined.

I added a fun little piece of code I found this week, getUrlVar. It will extract the URL parameters and let me use them in my app. I detest!! Alert messages, this allows me to add params to the URL and use them in my code. Add **?num=42** to your url and you’ll see the square of 42 displayed on the page.

I hope that helps. I had problems myself with import until I understood this concept of Scope as implemented with the script tag.

# In closing

How are your teams working out? I hope everyone is attending. Several of you have afternoon meetings. I am going to do my best to catch them this week.

Office hours are closed but I am always open to individual conversations. Please contact me, I am more than happy to look at your code and talk through whatever your challenge might be.

Hope to see you in your team meetings soon

Bro. James