Professional Skills

Agile Fundamentals

Jira

Git

Databases Introduction

Java Beginner

Maven

Testing (Foundation)

Java Intermediate

HTML

CSS

Javascript

- What is JavaScript
- Getting started with JS
- Variables
- Data types
- ASI
- Strict mode
- Iteration
- Conditionals with Truthy / Falsey
- Objects, Arrays + JSON
- **Structuring JS Code**
- Destructuring
- Scope
- Functions, function expressions and arrow functions
- The ECMAScript 6 Specification
- OOP in JavaScript
- Best Practices
- Closures
- Callbacks and Promises
- Cookies
- Hoisting
- Prototypes
- Query Parameters
- Higher Order Functions

Structuring JS Code

Contents

Overview

Stucturing one's code in a reasonable manner saves time, headaches and processing power in every language - JavaScript is no exception to the rule.

Tutorial

Structure within a script

With any new script that is created, ensure that it is kept out of the global scope - to do this, wrap it in a function.

If it is something that we can explicitly run later or at a specific time, then wrap it in a *named function* that can be called later:

```
let myScript = function () {
    // insert fantastic code
}

// Later on...
myScript();
```

Else, wrap it in an *IIFE (Immediately Invoked Function Expression)*.

```
(function() {
    // insert fantastic code
})();
```

Within scripts, it is important to break up code into three sections:

1. Variables
2. Methods
3. Initialisations and event listeners

This helps keep code predictable and organised.

○ Web Storage

○ DOM Manipulation

○ Handling Events and Timed Events

○ Asynchronous Programming

○ HTTP-Requests

○ XMLHttpRequests

○ Fetch API

Spring Boot

Selenium

Sonarqube

Advanced Testing (Theory)

Cucumber

MongoDB

Express

NodeJS

React

Express-Testing

Networking

Security

Cloud Fundamentals

AWS Foundations

AWS Intermediate

Linux

DevOps

Jenkins Introduction

Jenkins Pipeline

Markdown

IDE Cheatsheet

```
(function(){
    /*
    *
    * Variables
    *
    */

    let element = document.querySelector('#app');

    /*
    *
    * Methods
    *
    */
    let handleClicks = function(event){
        console.log(event.target);
    }

    /*
    *
    * Inits & Event Listeners
    *
    */

    element.addEventListener('click',handleClicks);
})();
```

Modular Code

Do not keep all of your code in one giant JavaScript file. That would be madness!

Ensure you create a `/js` directory in your project where you keep all the individual script files.

There are module bundlers such as `gulp` which you can use to combine fines (called concatenation) and minify them.

Exercises

Revisit any previous JS work, and ensure it follows the correct structure.