CODESPACE
imagine • learn • create

Interactive Web Apps (IWA) - Full Time

Course Expectations

Welcome to Interactive Web Apps and your introduction to the world of JavaScript!

In this course you will explore some of the most fundamental concepts in JavaScript. Starting with a step-by-step video where you will build a very simple web-app inspired by https://tallycount.app. After completing the first module: **Practical Example**, the course will retroactively explore the concepts covered in the initial video in more detail.

Course Outline

The course will take you through the following modules:

WEEK	MODULE	TASK CODE	DESCRIPTION
01	**Practical Example**	IWA1	Follow along as we add JavaScript to HTML and CSS in order to turn our page into a fully-working, interactive web app. This lesson does not have any challenges, but requires you to submit all HTML, CSS and JavaScript files as reflected at the end of the video to prove that you did indeed follow along. **You will be expected to follow along to the video, and submit your final outcome.** After completing the first video the course will retroactively explore some concepts covered in the initial video in more detail.
	Comments and Logging	IWA2	Commenting and logging of values, while not influencing the user-facing behaviour of our app are important tools in a developers toolkit. It not only aids in the dissection, debugging and rewriting of existing code, but can also be used effectively to help others work with your code.
	Script Element and Modules	IWA3	In order to connect JavaScript to a page we need to add a `<script>` element to our HTML. However, there are several ways that this `<script>` element can be configured. It is important to understand how JavaScript files can be loaded not only from the HTML document, but also from other JavaScript files.
	Variables and Lexical Scope	IWA4	Variables allow you to store information in your machine's memory. This is helpful if you want to use a piece of information later, or want to re-use it several times across your app. However, there is a set of rules determining how variables can be created and how they can be accessed after the fact.
02	**Branching Logic**	IWA5	Central to any imperative programming language is the ability to change user-facing behaviour based on specific internal conditions. Imagine how different an app looks if you are logged in, verses when you are logged out. Within this section we will explore the various ways that JavaScript allows us to influence behaviour based on specific conditions.
	Type Coercion	IWA6	Up until this point we've explored three different primitive data types, namely strings, numbers and booleans. JavaScript is a weakly typed language, which effectively means that it automatically converts values from one data type to another. Due to this it can be hard working with different types of data unless you understand the underlying coercion mechanisms.
	Strings and Logical Operators	IWA7	In the previous module we explored the different operations that can manually or automatically be performed on different primitive data types. However, it is worth diving a bit deeper into strings specifically, since they are somewhat unique in how they are used and manipulated, specifically by means of logical operators.
	Object Literal Syntax	IWA8	JavaScript not only supports primitive data types like strings, numbers, booleans, null and undefined. It also supports various composite data types. These composite types are used to group related primitive values. Within this lesson we will look at the most pervasive composite type, the JavaScript object literal.
03	**Using Objects**	IWA9	Now that we are familiar with grammar used by JavaScript objects, we will explore ways in which you are able to meaningfully use objects in JavaScript. First and foremost is understanding how the values inside objects relate to one another. We will look at ways in which you can clearly communicate the nature and purpose of objects.

	Built-in Objects	IWA10	While JavaScript allows you to create any custom object you wish, it does come with a massive range of built-in objects. While you will never fully understand, nor memorise, all these objects it is worthwhile familiarising yourself with some objects that are common in day-to-day JavaScript development. In addition, we will also look at how you can go about expanding your knowledge of built-in objects.
	Document Object Mode	IWA11	If you are working in the browser you will spend the majority of your time using a specific built-in object. This object, called the Document Object Model (often, more simply the "*DOM*") is the means by which you can modify, in real-time, what is currently being shown to the user in the browser.
	Style Object	IIWA12	While the DOM allows the real-time updating of the content shown to the user, it also allows updating content styling based on JavaScript internals. There are several ways in which this can be accomplished, however we will start by simply using the nested `style` object directly on DOM elements themselves.
04	**Intro to Functions**	IWA13	Up until now we've exclusively explored how variables can be used to save data for reuse. However, we are also able to store behaviour in variables for reuse. These are called functions, and allow us to run (and re-run) a set grouped together actions
	Types of Functions	IWA14	Functions, similar to the variables, can be created in a variety of different ways. The different ways that we can create functions actually have a big impact on how they behave. In this lesson we will explore the various ways to create and use functions.
	Arrays and Destructuring	IWA15	Javascript arrays provide a means to store data in a list-like order. Due to their linear organisation of information by means of indices they are great when you want to loop over a set of data.
05	**Using Arrays**	IWA16	Working with arrays in JavaScript is tricky because they are mutable and can be modified in place. This makes it difficult to keep track of their state and changes, and can lead to unexpected behaviour. In this lesson we will explore some pitfalls when working with arrays in JavaScript.
	Loops	IWA17	JavaScript loops are super powerful when used in conjunction with arrays. It allows the iterative execution of a specific piece for each item inside an array. In addition, it is possible to run arrays on specific conditions as well without any array present.
	Events and Listeners	IWA18	Javascript events are messages sent by the browser or user when a certain action is performed, such as clicking on a button. These events allow developers to execute certain functions in response to user or browser actions, such as when a user clicks on a button, a popup window appears.
06	**Capstone Week**		During this week, you will have time to work on, complete and submit your final project. **The deadline for the capstone project will be on Friday @ 17:00.**
07	**Live Assessment Week**		1. The final assessment will require the student to upload their project to Github and submit the link via the **[Projects]** tab by the due date. 2. Students will then be required to do a **live presentation of their project and code** to a panel of coaches and subject matter experts. 3. The minimum required mark to pass a course and proceed to the next course, is 50%. **4. If a student achieves a 40-49% mark for their first submission, they will be given the opportunity to resubmit within 7 days. Failing that, a student will be required to redo the course.** 5. If a student achieves 39% or less, they will be required to redo the course.
08	**Resubmission Week**		**If a student achieves a 40-49% mark for their first submission, they will be given the opportunity to resubmit within 7 days. Failing that, a student will be required to redo the course.**

Each module will begin with a pre-recorded lecture covering all concepts needed to complete the module exercises. Some of the concept videos will be hard to understand. It is therefore assumed that you will pause the videos quite often, rewind and perhaps watch them several times. Go over each timestamp to ensure you understand all the concepts covered.

After watching each concept video you will be required to complete several code challenges, to confirm that you correctly understood all concepts covered in a specific video/lesson.

It is essential that you don't skip over any content. Ensure you understand the concepts before moving to the next module. Each module builds on top of the previous one and by completing every module, lesson and exercise you are less likely to become confused.

PLEASE NOTE: submissions for this course will be validated during a contact session with your designated coach. **If you are unable to validate your understanding of the code, your submission will be marked as unsuccessful.**

During the code validation session you will be asked to take the coach through your code challenges, not only explaining **what you did** - but also **why you did it**, and what other approaches you could have taken.

If a coach feels that you have not mastered the specific content of a lesson by means of the code challenges, they will ask you to return to the challenges and resubmit. Please manage your time wisely and work with your coach to ensure you meet your deadlines successfully.

Course Outcome

As you progress through the software development program, you will not only develop the knowledge and skill to code, but our teaching and learning approach also aims to foster the necessary skills and scenarios you will face in a career in tech.

The skills include:

- Presentation skills
- Planning and research
- Learning how to learn, adapt and onboard new tools, tech stacks or coding languages
- Talking confidently about your code
- Identifying and implementing bug fixes

As a junior developer, you will very likely be added to a cross functional and multi skilled team of developers, where you will find yourself working on small features, retrofitting older projects or refactoring code. Understanding how to read, explain and debug code is a vital skillset.

Explaining how and why you resolved the bugs and why you chose a specific solution is critical. The entire team, your stakeholders and clients need to understand that your solution is or was the appropriate way forward to secure the longevity of an application or product.

For the Interactive Web Application (IWA) Capstone Project you will be presented with an existing codebase for a web application, You will be required to audit and fix the web app to function as it is intended to work. However you will also be expected to provide recommendations on how code can be improved (even the parts that are working, but can be done better).

Once you have reworked the code. You will be required to submit the code for review and then present your code in a live panel review, where your understanding of the code, why you chose certain solutions, and made specific code improvements, will be tested.

To successfully meet the outcomes of IWA, you will need to demonstrate appropriate knowledge and understanding of concepts learned, in order to proceed to the next course in the software development program.