# **Web Dev ~ Full stack**

# **Module 1 ~ Introduction to web development**

What is web development?

Web development has two disciplines: front-end and back-end.

Front-end is visual and interactive aspects of a website. You will learn HTML, CSS and JavaScript to master front-end web development.

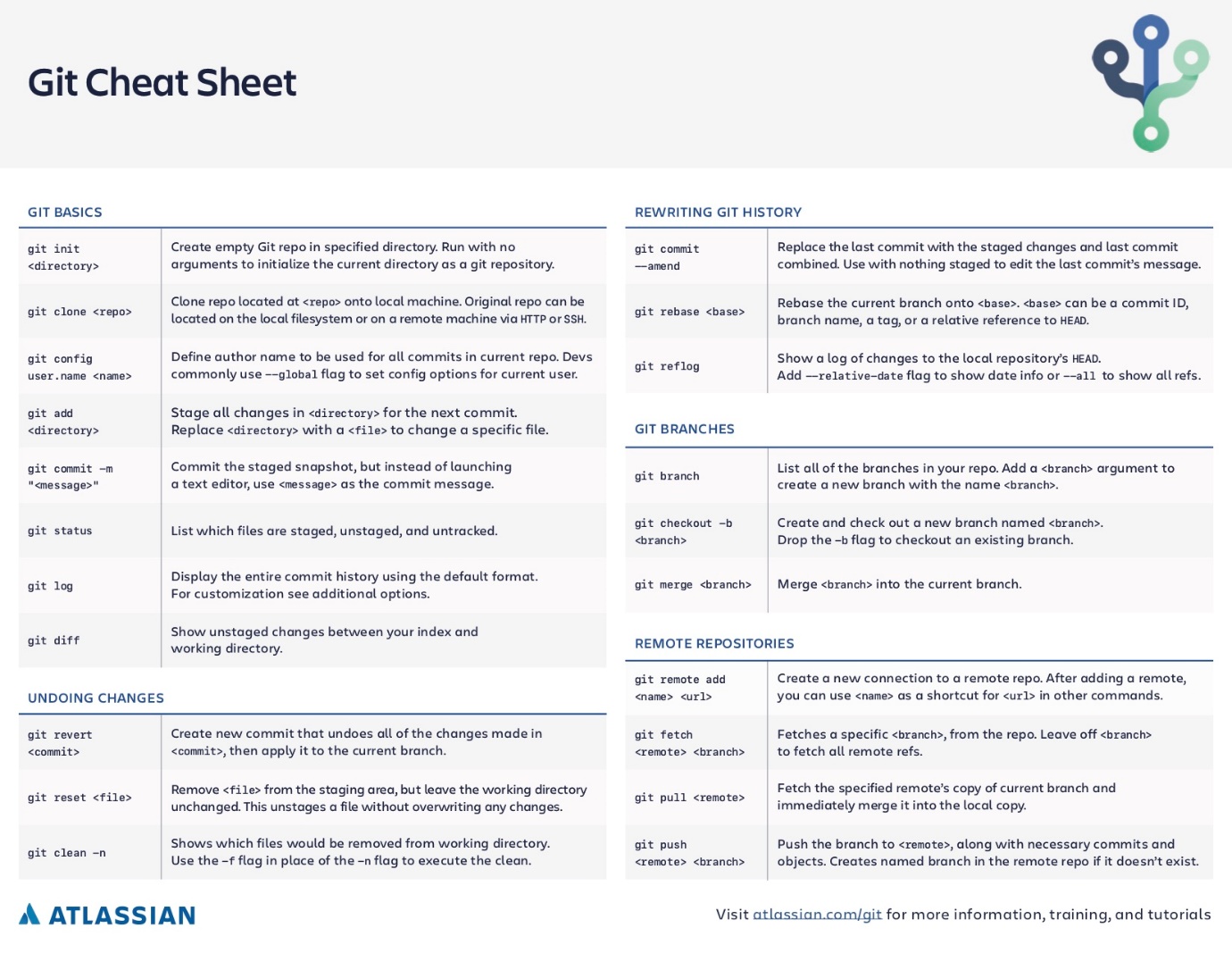
Back-end is all the logic behind the scenes that supports your website: databases, user management, etc. You will need to learn any one of the following back-end languages and frameworks:

* Ruby on Rails (used by many startups; taught in Full-stack Web Development)
* Node.js (JavaScript but for backend; used by many startups)
* Java (used by Google)
* PHP (used by Facebook)
* C# (used by Microsoft)
* Python (popular amongst solopreneurs)

And you need to learn about database, model-view-controller, testing.

You will build a unique full-stack project, Capstone project, and even learn how to deploy your project so it's live and accessible on the Internet

# **Module 2 ~ version control system**



<https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet>

1. **Command line interface** - Command line interface is a text based interface you can use to command your computer. Learn to navigate folder structures; create, move, delete files and folders. You need to know this well in order to use Git.
2. **Track versions with Git** - Git is a version control system. It means that if you are editing a project, you can save a version of the project as snapshot. It is also very useful for managing projects with multiple contributors. You will understand the Git lifecycle, from initiation to committing changes.
3. **Cloud storage with GitHub** - Just like backing up your photos to Google drive or iCloud, developers back up their project to third party storage services like GitHub. GitHub is also a project management tool that offers issue tracking, code reviews, and project collaboration.

# **Module 3 ~ Basics of HTML CSS JS**

You will learn how to write basic HTML and CSS. Install and use a code editor. Create and view a bare bone website locally on your computer.

## HTML

1. **HTML and CSS syntax** - Just like grammar in spoken languages, each computer language has their own syntax. Syntax is a set of rules that you must follow when you write code, otherwise the computer cannot understand your instructions. You will learn the syntax for HTML and CSS.
2. **HTML elements** - Elements are the basic building blocks of HTML. Each element represents a certain content type. Elements are defined using HTML tags. HTML elements give meaning to your content as well as help you organise it.
3. **HTML document structure** - HTML documents have a standard structure you need to follow otherwise browsers won’t be able to read, process, and render it properly. You will learn what this structure is and adhere to it.



1. **CSS selectors, properties, and values** - CSS can target specific elements in HTML and apply styles to it. In order to do this, you need to specify a selector, choose the correct property and define its property value. Selectors do the targeting, properties and values do the styling.
2. **Where to write CSS** - You can write inline CSS directly on HTML elements, inside a style element, or in a separate dedicated CSS file. You will learn know how to write CSS in all three ways and how to connect CSS files with HTML files.
3. **Code commenting in HTML and CSS** - Comments are not interpreted by the browser so they don't get rendered for the user to see when they are looking at your website. You can use comments to leave commentary in your source code and help you compartmentalize sections of code.
4. **Pick a code editor** - When working on HTML and CSS files locally on your computer, you will learn to use a text editor. Your efficiency can be greatly improved if you use a text editor designed for coding. We recommend Atom for beginners. See this Minimum setup on Atom (text editor) for front-end development guide to get started. Alternatives include Sublime Text, Brackets, Visual Studio Code.
5. **Build your first web page** - Create a simple webpage with a heading using one HTML file and one CSS file and view it with a web browser. You will learn to do this correctly and your HTML and CSS files are connected and syntax error free.
6. **Get familiar with browser DevTool** - Most modern web browsers will have developer tools that aids developers in website development. You should get familiar with the devtool of your browser. If you don’t know which browser to choose, we recommend Google Chrome or Mozilla Firefox.
7. **Semantics of HTML** - Most HTML elements carry semantic values. Semantic values are used by screen readers and search engines to make sense of what the content represents; such as using the <address> element for addresses, or the <h1> element for primary heading. Using correct semantic elements can potentially improve your search engine ranking.
8. **Block and inline elements** - Most HTML elements are either block or inline elements. Block elements always start on a new line and span the full width of its parent element. Inline elements can stack horizontally since they only take up the width of their content. You will understand thoroughly the difference between block and inline elements so you know when to use which.
9. **Nesting of elements** - You will learn how to nest elements within elements, it is often used to create complex layout structures.
10. **Text elements** - Text elements are used to contain text content. You will learn about the six heading elements, the paragraph element, and inline elements such as strong and emphasize. Text elements are the bread and butter of most informational websites.
11. **Structure elements** - Structure elements are used to break up an HTML page into compartments so search engines understand the structure better. Learn about header, nav, articles, sections, aside and footer. Those carry semantic values and should be used to layout your HTML file.
12. **Hyperlinks** - Hyperlink, or anchor tag, is an element that links to another file, page or website. You will use it to build connections between pages of your website. It is what search engines will use to map out the relationship between pages of your website, and even relationships between different websites.
13. **Lists** - Unordered and ordered lists are used to create lists in HTML. You will learn the structures.
14. **Image, video, audio elements** - You can add media content to a web page by using image, video, and audio HTML elements. Each of them have their own set of attributes you can use for fine tuning.
15. **Iframe elements** - Iframe elements are used to embed and render other websites. Think of it like screen within screen on TV. It is often used to embed services provided by third parties, such as map, forms, booking management, etc.
16. **Tables** - Tables are used for, well tables. This element requires the use of multiple HTML elements and can be a bit confusing at first.
17. **Forms** - Forms are the predominant way to retrieve information from users. You will learn about form, input, text area, select, and button element.
18. **HTML best practices** - Just getting the syntax right doesn’t mean your code will be clean and easy to read. Just like knowing how to spell and form sentences doesn’t mean your handwriting is legible or you can write wonderful novels. It is important to adapt a good set of best practices for writing HTML. These are usually called style guides and you can find many popular ones by searching.
19. **Create a basic multi page website** - At this point, you should try to create a multi-page website with multiple three to four HTML files. Use hyperlinks to connect each HTML file. Launch the website on your browser and make sure you can access all pages. Keep the content minimal.

## CSS

1. **Cascading effect** - CSS stands for Cascading Style Sheet. Cascading means CSS code is read and processed from top to bottom. You will learn how this affects your CSS code.
2. **Specificity** - CSS selectors, used for targeting HTML elements, each carry a specificity weight. Selectors with higher specificity will get to render their styles, while styles from selectors with lower specificity will be ignored.
3. **Combining and layering selectors** - You can combine different types of selectors, namely Type, Class, and Id, for HTML targeting. You will learn how to calculate the specificity from combined selectors. You can leverage this to write efficient, modularized CSS.
4. **Advanced selectors** - You will learn how to use pseudo-classes, such as :hover, :focus, :first, and attribute selectors to effectively target elements. As an example, the selector p:nth-child(odd) will let you target and apply styles to all odd numbered paragraph elements.
5. **Colours** - Colours are used in defining font Colours, border Colours, background Colours and box shadow colours. You will learn how to define colours using colour keywords, e.g. blue, white, black; hexadecimal code, e.g. #fff, #678998; rgb(), rgba() functions; and hsl(), hsla() functions.
6. **Lengths** - Length is used to define the size and position of things. It is used for font size, height, width, padding, margin and many more size and position properties. You will learn about absolute lengths, e.g. pixel, points, cm, mm, inch; and relative lengths, e.g. percentage (%), em, rem, vh, vw.
7. **Typography** - Most websites are for serving information, text styling is therefore a large part of front-end development. You will learn how to use common CSS typography properties. These properties deal with text and are categorized into font- and text-. Common ones include font-family, font-size, font-weight, text-align, text-decoration, text-transform, letter-spacing, word-spacing.
8. **Backgrounds and gradients** - Learn how to add background colours and background images to elements. Be able to position and repeat background images. Learn how to add gradient colour backgrounds.
9. **CSS resets** - Every browser has their own set of default styles for rendering HTML. To tackle this cross-browser consistency issue, we can use CSS resets. A CSS reset is a set of CSS rules that strips or tones down the styles of each HTML element, setting a consistent base for us to work on.
10. **CSS best practices** - Adapt a good style guide for writing clean CSS. Learn to name your class and id selectors effectively. This is very important for writing CSS that can be easily understood by fellow developers.

## JS

1. **Values and Types** - Programming is mostly for dealing with data. There are six basic types of data values in JavaScript. You will learn the most basic four first. Numbers, strings, booleans, and undefined values.
2. **Operators** - Operators are used to combine, compare, and manipulate data values. Learn to use arithmetic operators such as +, -, \*, /, %; comparison operators such as <, <=, ===, !==, >, >=; logical operators such as &&, ||, !. You also will learn the automatic type conversion that happens in JavaScript operators, which can lead to unintuitive outputs.
3. **Variables** - Variables are used to temporarily store values, so we can reuse them at different stages of a program. For example, when you are counting in your head, you keep check of the current count and continue to add numbers to it. In JavaScript, you create a variable to do this. Learn how to create new variables; assign and re-assign values to variables; use variable update shortcuts such as -=, +=, ++, --; and variable naming conventions in JavaScript.
4. **Where to write and test JavaScript** - To improve, you need to practice. The most accessible resource for you to write and test JavaScript programs is to use the browser devTool JavaScript console. Most browser’s that have a devTool suite will have a JavaScript console. It works when you don’t have internet connection too. If you want something more sophisticated, check out online coding environments such as repl.it. It offers a side by side view of a text editor and a JavaScript console for you to write and test your code.
5. **Statements, expressions and programs** - An expression is a fragment of code. While a statement is a line of expression(s) with a semicolon after it. A program is simply a collection of statements. Understand the difference between the three and learn about the straight control flow of JavaScript programs.
6. **Conditional statements** - When you want to execute some code only on certain conditions, you will need to use conditional statements. For example, the alarm should only sound on Sundays. Learn to use if, else, else if conditional statements with comparison expressions to create basic programs.
7. **Loops** - Loops are very common in programming. It is used when we want to repeat the execution of some tasks. There are three basic types of loops in JavaScript, while, do, and for. Each has their own syntax and quirks. You also will learn how to conditionally break out of a loop.
8. **Switch** - A switch statement is a special construct that lets you write easier to read conditional statements. You can use switch statements to replace lengthy and convoluted if/else statements.
9. **Syntax guide and commenting** - Following a good set of best practice for JavaScript syntax will keep your code consistent and easy to read. Which makes bug finding a lot easier. You also will learn how to add comments to your code so you can leave reminders or explanations.

# **Module 4 ~ Introduction to Bootstrap**

1. **Installation** - There are multiple ways to add Bootstrap in a project. The easiest way is to link to Bootstrap’s official CDN (content delivery network) and let the CDN handle serving Bootstrap’s CSS file to the clients. The other way is to install Bootstrap locally into your project. This is useful if you are working without internet. You will learn the pros and cons of each method.
2. **Responsive grid system** - Probably Bootstrap’s most popular feature is its grid system. You need to know it well enough to create complex nested grid structures with ease. This is used to create responsive layout for web content very quickly. Responsive layout means the layout will change based on the screen size of the viewing device.
3. **Bootstrap's style standards** - Bootstrap comes with a standardised set of styles for typography, colours, buttons, forms, tables, images and more. It also has a large selection of helper classes that lets you add predefined styles to elements. You will gain a good command of these styles and classes.
4. **Responsive CSS** - You will learn how to write CSS that is effective only on specific screen width ranges. This is called responsive CSS and it is how Bootstrap’s responsive grid is made. Knowing this will allow you to create custom responsive styles for web pages.
5. **Responsive development with browser devTools** - When working on responsive CSS, it would be useful to quickly view the result on screen sizes of different devices. Most web browser devtools will have a device mode that lets you do exactly this. Use this feature to increase your efficiency.
6. **Responsive website clone** - Equipped with Bootstrap 3, you should try to build a responsive webpage. Pick a couple simple responsive pages to clone.

# **Module 5 ~ Advanced CSS with Bootstrap 4**

1. **Z dimension** - There is a third dimension in web pages for depth, it’s the z-axis. You will learn about stacking context, its rules, and how to change the z-axis position of an element using the z-index property. This is a complex topic to master with many conditions that affect outcomes. But it is an important one. So I urge you to spend adequate time on it to fully understand it.
2. **Transform** - Transform is a CSS property capable of distorting and moving a HTML element. Learn how to use scale, skew, rotate, and translate.
3. **Transition** - The CSS transition property is used to add transition effects during element state changes. Elements change states when they are hovered, focused, or active. Such as making a link change colour when the mouse is hovering it. You will learn how to write basic transitions and understand the difference between each transition timing function.
4. **Animation** - You can create animations using the CSS animation property. Although animations are generally frowned upon, you will at least know how it works.
5. **Flexbox** - Flexbox is a CSS module that lets you create flexible layout structures. It is super convenient and a must learn for the modern developer. You will understand the relationship between a flex container and its flex items; flex container properties such as flex-direction, flex wrap, justify-content, align-items; flex item properties such as order, flex-grow, flex-shrink, flex-basis.

# **Module 6 ~ Introduction to PHP**

# **Module 7 ~ Database Development**

- MySqli

# **Module 8 ~ SEO**

# **Module 9 ~ Project**

To further solidify all the skills, you learned, you will come up with a simple website idea and build it out. If you don’t know what to build, consider creating your personal portfolio page. Use a CSS framework and track your project with Git. Try to achieve the following in your website.

# **Module 10 ~ Deploying**

You should know how to host your simple HTML CSS websites on third party services so people can access it. For a basic website to work, you simply need a dedicated computer that will listen for requests to serve the corresponding HTML CSS files.

Many web hosting services offer free hosting for static file websites, that is websites that only consist of static files, such as HTML, CSS, JavaScript and images. But you will be restricted to a subdomain. Getting a custom domain will usually cost you 8 to 10 USD a month. Below are some free services to host your HTML CSS projects.

1. GitHub Pages (https://pages.github.com/) - GitHub offers one free site for each project. You just need to activate it in your project settings.
2. Netlify (https://www.netlify.com/) - Netlify is a simple to use static web project hosting service. You just need to drag your project folder onto the browser to upload the deploy your projects.
3. Neocities (https://neocities.org/) - Also an easy to use web hosting site that offers free static file hosting with a subdomain of your choice.