**CHAPTER 1**

**INTRODUCTION**

**1.1 Web Development**

Web development is a specific field of software engineering that focuses on building web pages. Web pages, or web apps, are codebases that are downloaded and run in our web browser (e.g., Google Chrome) each time a user navigates to the website address.

Web development is the work involved in developing a [Web site](https://en.wikipedia.org/wiki/Web_site) for the [Internet](https://en.wikipedia.org/wiki/Internet) or an [internet](https://en.wikipedia.org/wiki/Intranet) (a private network).Web development can range from developing a simple single [static page](https://en.wikipedia.org/wiki/Static_Web_page) of [plain text](https://en.wikipedia.org/wiki/Plain_text) to complex Web-based [Internet applications](https://en.wikipedia.org/wiki/Internet_application) (Web apps), [electronic businesses](https://en.wikipedia.org/wiki/Electronic_business), and [social network services](https://en.wikipedia.org/wiki/Social_network_service).

There are two broad divisions of web development – front-end development (also called client-side development) and back-end development (also called server-side development).

Front-end development refers to constructing what a user sees when they load a web application – the content, design and how you interact with it. This is done with three codes – HTML, CSS and JavaScript.

Responsive design allows software developers to build a Web page that can dynamically adapt to the size of the devices. This development philosophy enables the rendering of Web pages in a fast and optimized way, ensuring a good user experience on mobile devices, tablet and desktop.

Websites are files stored on servers, which are computers that host (fancy term for “store files for”) websites. These servers are connected to a giant network called the internet … or the World Wide Web (if we’re sticking with 90s terminology). We talk more about servers in the next section.

Browsers are computer programs that load the websites via your internet connection, such as Google Chrome or Internet Explorer. Your computer is also known as the client.

Front-end (or client-side) is the side of a website or software that you see and interact with as an internet user. When website information is transferred from a server to a browser, front-end coding languages allow the website to function without having to continually “communicate” with the internet.

Front-end code allows users like you and me to interact with a website and play videos, expand or minimize images, highlight text, and more. Web developers who work on front-end coding work on client-side development.

A front-end dev takes care of layout, design and interactivity using HTML, CSS and JavaScript. They take an idea from the drawing board and turn it into reality. What you see and what you use, such as the visual aspect of the website, the drop-down menus and the text, are all brought together by the front-end dev, who writes a series of programmes to bind and structure the elements, make them look good and add interactivity.

**CHAPTER 2**

**HTML**

**2.1 Introduction to HTML**

HTML, otherwise known as Hypertext Markup Language, is the language used to create Web pages. Using HTML, you can create a Web page with text, graphics, sound, and video. It’s the work that happens behind the scenes to make a website look great, work fast and perform well with a seamless user experience. HTML describes the structure of a Web page.HTML consists of a series of elements. When you save the document, it creates an .html file. Markup language is used to define the text document within tag which defines the structure of web pages. This language is used to annotate (make notes for the computer) text so that a machine can understand it and manipulate text accordingly. Most markup languages (e.g., HTML) are human readable.

**2.2 HTML TAGS**

HTML is a markup language and makes use of various tags to format the content. These tags are enclosed within angle braces.

The opening and closing tags use the same command except the closing tag contains and additional forward slash /

Any document starts with a heading. You can use different sizes for your headings. HTML also has six levels of headings, which use the elements <h1>, <h2>, <h3>, <h4>, <h5>, and <h6>. While displaying any heading, browser adds one line before and one line after that heading.

Various tags are:

* The <!DOCTYPE html> declaration defines that this document is an HTML5 document
* The <html> element is the root element of an HTML page
* The <head> element contains meta information about the HTML page
* The <title> element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
* The <body> element defines the document's body, and container for visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
* The <h1> element defines a large heading
* The <p> element defines a paragraph
* The <br> element defines a line break
* The <div> element defines a division content

**2.3 Structure of a Web Page**

<!DOCTYPE html>

<HTML>

<HEAD>

<TITLE> Example </TITLE>

</HEAD>

<BODY>

This is where you would include the text and images on your Web page.

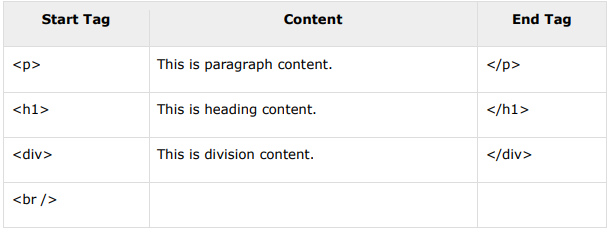
</BODY>

</HTML>

**Table 2.1: Structure of a web page**

**2.4 HTML Elements**

An HTML element is defined by a starting tag. If the element contains other content, it ends with a closing tag, where the element name is preceded by a forward slash as shown below with few tags:



**Table 2.2: HTML Elements**

**2.5 HTML Attributes**

An attribute is used to define the characteristics of an HTML element and is placed inside the element's opening tag. All attributes are made up of two parts: a name and a value:

**2.5.1 Name**

Name is the property you want to set. For example, the paragraph <p> element in the example carries an attribute whose name is align, which you can use to indicate the alignment of paragraph on the page.

**2.5.2 Value**

Value is what you want the value of the property to be set and always put within quotations. The below example shows three possible values of align attribute: left, center and right.

**2.6 HTML Formatting**

Manipulating text in HTML can be tricky; Oftentimes, what you see is not what

you gate.

For instance, special HTML tags are needed to create paragraphs, move to the next line, and create headings

<B> Bold Face </B>

<strong> Strong </strong>

<I> *Italics* </I>

<em> *emphasized* </em>

<U> Underline </U>

<P> New Paragraph </P>

<BR> Next Line

**2.7 Uses of HTML**

**2.7.1 Web pages development**

#### [HTML is heavily used for creating pages](https://www.educba.com/html-style-sheets/) that are displayed on the world wide web.

**2.7.2 Web document Creation**

Document creation on the internet is dominated by HTML and its basic concept via and DOM i.e., document object model. HTML tags are inserted before and afterward or phrases to locate their format and location on the page.

#### **2.7.3 Internet Navigation**

This is one of the most important uses of HTML which is revolutionary. This navigation is possible by utilizing the concept of Hypertext.

#### **2.7.4 Responsive images on web pages**

At the elementary level in applications of HTML, queries can be set to utilize the images, which are responsive in nature. With the srcset attribute of img element in HTML, and combining it with picture elements, a developer can fully control how the user will render an image.

**2.7.4 Client-side storage**

Earlier, a user could not save the user’s browser data that would persist across sessions. To meet this requirement, server-side infrastructure has to be built or user’s cookies can be used. But with HTML5, client-side storage is feasible using local Storage and Index DB.

**CHAPTER 3**

**CASCADING STYLE SHEETS**

**3.1 Introduction to CSS**

CSS stands for Cascading Style Sheets. CSS describes how HTML elements are to be displayed on screen, paper, or in other media. CSS saves a lot of work. It can control the layout of multiple web pages all at once. External stylesheets are stored in CSS files.CSS is the language we use to style a Web page.CSS handles the look and feel part of a web page.

Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.

A **document** is usually a text file structured using a markup language — [HTML](https://developer.mozilla.org/en-US/docs/Glossary/HTML) is the most common markup language, but you may also come across other markup languages such as [SVG](https://developer.mozilla.org/en-US/docs/Glossary/SVG) or [XML](https://developer.mozilla.org/en-US/docs/Glossary/XML).

**3.2 CSS Syntax**

CSS is a rule-based language you define rules specifying groups of styles that should be applied to particular elements or groups of elements on your web page.

A style rule is made of three parts: -

**3.2.1 Selector**

A selector is an HTML tag at which a style will be applied. This could be any tag like <h1> or <table> etc.

**3.2.2 Property**

A property is a type of attribute of HTML tag. Put simply, all the HTML attributes are converted into CSS properties. They could be color, border etc.

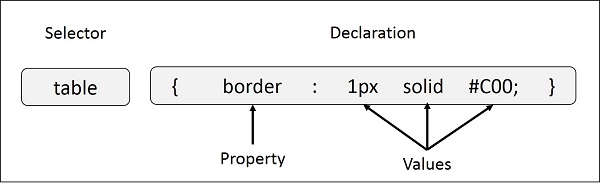
**3.2.3 Value**

Values are assigned to properties. For example, color property can have value either red or #F1F1F1 etc.

CSS value definition syntax, a formal grammar, is used for defining the set of valid values for a CSS property or function. In addition to this syntax, the set of valid values can be further restricted by semantic constraints (for example, for a number to be strictly positive).

You can put CSS Style Rule Syntax as follows: -

selector {property: value}



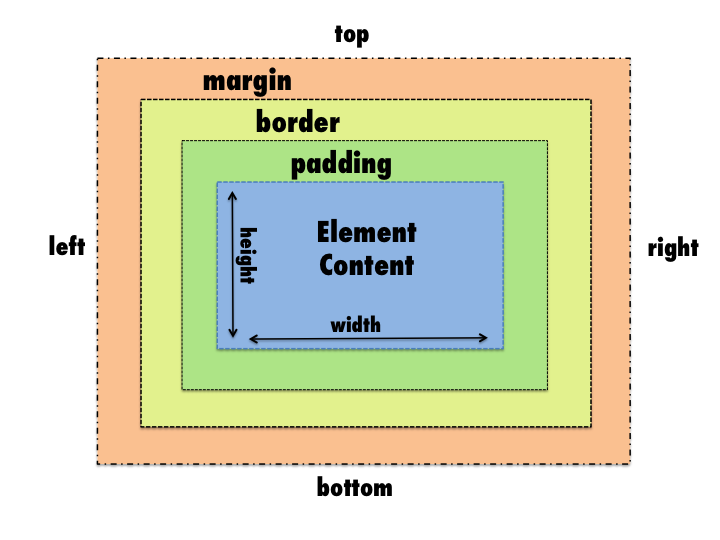
**Figure 3.1: CSS Syntax**

## **3.3 CSS Margin**

The CSS margin properties are used to create space around elements, outside of any defined borders.

**3.4 CSS Padding**

The CSS padding properties are used to generate space around an element's content, inside of any defined borders.



**Figure 3.2: CSS Margin and Padding**

**3.5 Types of CSS**

**3.5.1 External Style Sheet: -**

An ESS is ideal when the style is applied to many pages.

<head>

<link rel=“stylesheet” type=“text/css” href=“style.css”>

</head>

**3.5.2 Internal Style Sheet: -**

An ISS would be used when a single document has a unique style.

<head>

<style>

p{margin-left:20px;

}

body{background-image:url(“images/logo.png”);

}

</style>

</head>

## **Advantages of CSS**

**3.6.1 CSS saves time** − You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.

**3.6.2 Pages load faster** − If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So, less code means faster download times.

**3.6.3 Easy maintenance** − To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.

**3.6.4 Superior styles to HTML** − CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.

**3.6.5 Multiple Device Compatibility** − Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.

**3.6.6 Global web standards** − Now HTML attributes are being deprecated and it is being recommended to use CSS. So, its a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.

**CHAPTER 4**

**JAVASCRIPT**

**4.1 Introduction to JavaScript**

JavaScript is a lightweight, interpreted programming language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java. JavaScript is very easy to implement because it is integrated with HTML. It is open and cross-platform.

• JavaScript gives HTML designers a programming tool

• JavaScript can put dynamic text into an HTML page

• JavaScript can react to events

• JavaScript can read and write HTML elements

• JavaScript can be used to validate input data

• JavaScript can be used to detect the visitor's browser

**4.2 Uses of JavaScript**

Adding interactive behavior to web pages-JavaScript allows users to interact with web pages. There are almost no limits to the things you can do with JavaScript on a web page.

Creating web and mobile apps-Developers can use various JavaScript frameworks for developing and building web and mobile apps.

Building web servers and developing server applications-Beyond websites and apps, developers can also use JavaScript to build simple web servers and develop the back-end infrastructure using Node.js.

Game development-Of course, you can also use JavaScript to create browser games.

* 1. **JavaScript Syntax**

JavaScript can be implemented using JavaScript statements that are placed within the <script>... </script> HTML tags in a web page.

The <script> tag alerts the browser program to start interpreting all the text between these tags as a script. A simple syntax of your JavaScript will appear as follows.

You can place the <script> tags, containing your JavaScript, anywhere within your web page, but it is normally recommended that you should keep it within the <head> tags.

This attribute specifies what scripting language you are using. Typically, its value will be JavaScript. Although recent versions of HTML (and XHTML, its successor) have phased out the use of this attribute. This attribute is what is now recommended to indicate the scripting language in use and its value should be set to "text/JavaScript".

<script ...>

JavaScript code

</script>

## **Figure 4.1: Syntax of JavaScript**

## **Advantages of JavaScript**

* **Speed: -** Client-side JavaScript is very fast because it can be run immediately within the client-side browser. Unless outside resources are required, JavaScript is unhindered by network calls to a backend server.
* **Simplicity: -** JavaScript is relatively simple to learn and implement.
* **Popularity: -** JavaScript is used everywhere on the web.
* **Interoperability**: - JavaScript plays nicely with other languages and can be used in a huge variety of applications.
* **Server Load: -** Being client-side reduces the demand on the website server.
* Gives the ability to create rich interfaces.

**4.5 Features**

The following features are common to all conforming ECMAScript implementations, unless explicitly specified otherwise.

### 4.5.1 Imperative and structured

JavaScript supports much of the [structured programming](https://en.wikipedia.org/wiki/Structured_programming) syntax from [C](https://en.wikipedia.org/wiki/C_(computer_language))  One partial exception is [scoping](https://en.wikipedia.org/wiki/Scope_(computer_science)): JavaScript originally had only [function scoping](https://en.wikipedia.org/wiki/Function_scoping) with . ECMAScript 2015 added keywords and block scoping, meaning JavaScript now has both function and block scoping. Like C, JavaScript makes a distinction between [expressions](https://en.wikipedia.org/wiki/Expression_(computer_science)) and statements. One syntactic difference from C is [automatic semicolon insertion](https://en.wikipedia.org/wiki/Defensive_semicolon), which allows the semicolons that would normally terminate statements to be omitted.

JavaScript ignores spaces, tabs, and newlines that appear in JavaScript programs. You can use spaces, tabs, and newlines freely in your program and you are free to format and indent your programs in a neat and consistent way that makes the code easy to read and understand.

**4.5.2 Weakly typed**

JavaScript is [weakly typed](https://en.wikipedia.org/wiki/Strong_and_weak_typing), which means certain types are implicitly cast depending on the operation used.

* The binary operator casts both operands to a string unless both operands are numbers. This is because the addition operator doubles as a concatenation operator
* The binary operator always casts both operands to a number
* Both unary operators always cast the operand to a number

Values are cast to strings like the following:

* Strings are left as-is
* Numbers are converted to their string representation
* Arrays have their elements cast to strings after which they are joined by commas

Values are cast to numbers by casting to strings and then casting the strings to numbers.

JavaScript has received criticism for the way it implements these conversions as the complexity of the rules can be mistaken for inconsistency. For example, when adding a number to a string, the number will be cast to a string before performing concatenation, but when subtracting a number from a string, the string is cast to a number before performing subtraction.

misleading: the is interpreted as an empty code block instead of an empty object, and the empty array is cast to a number by the remaining unary operator. If you wrap the expression in parentheses the curly brackets are interpreted as an empty object and the result of the expression is as expected.

### 4.5.3 Dynamic Typing

associated with a value rather than an expression. For example, a variable initially bound to a number may be reassigned to a string. JavaScript supports various ways to test the type of objects, including duck typing.

**4.5.4 Run-time evaluation**

JavaScript includes a function that can execute statements provided as strings at run-time.

**4.5.5 Object-orientation (prototype-based)**

Prototypal inheritance in JavaScript is described by as:

You make prototype objects, and then … make new instances. Objects are mutable in JavaScript, so we can augment the new instances, giving them new fields and methods. These can then act as prototypes for even newer objects. We don't need classes to make lots of similar objects… Objects inherit from objects. What could be more object oriented than that?

In JavaScript, an [object](https://en.wikipedia.org/wiki/Object_(computer_science)) is an [associative array](https://en.wikipedia.org/wiki/Associative_array), augmented with a prototype (see below); each key provides the name for an object [property](https://en.wikipedia.org/wiki/Property_(programming)), and there are two syntactical ways to specify such a name: dot notation and bracket notation A property may be added, rebound, or deleted at run-time. Most properties of an object (and any property that belongs to an object's prototype inheritance chain) can be enumerated using a loop.

**4.5.6 Prototypes**

JavaScript uses [prototypes](https://en.wikipedia.org/wiki/Prototype-based_programming) where many other object-oriented languages use [classes](https://en.wikipedia.org/wiki/Class_(computer_science)) for [inheritance](https://en.wikipedia.org/wiki/Inheritance_(computer_science)). It is possible to simulate many class-based features with prototypes in JavaScript.

**4.5.7 Functions as object constructors**

Functions double as object constructors, along with their typical role. Prefixing a function call with new will create an instance of a prototype, inheriting properties and methods from the constructor (including properties from the prototype). ECMA Script 5 offers the method, allowing explicit creation of an instance without automatically inheriting from the prototype (older environments can assign the prototype to the constructor’s property determines the object used for the new object's internal prototype. New methods can be added by modifying the prototype of the function used as a constructor. JavaScript's built-in constructors, such as, also have prototypes that can be modified. While it is possible to modify the prototype, it is generally considered bad practice because most objects in JavaScript will inherit methods and properties from the prototype, and they may not expect the prototype to be modified.

ECMA Script 5 offers the method, allowing explicit creation of an instance without automatically inheriting from the prototype (older environments can assign the prototype to the constructor’s property determines the object used for the new object's internal prototype. New methods can be added by modifying the prototype of the function used as a constructor.

**4.5.8 Functions as methods**

Unlike many object-oriented languages, there is no distinction between a function definition and a [method](https://en.wikipedia.org/wiki/Method_(computer_science)) definition. Rather, the distinction occurs during function calling; when a function is called as a method of an object, the function's local this keyword is bound to that object for that invocation.

**4.5.9 Functional**

A [function](https://en.wikipedia.org/wiki/Subroutine) is [first-class](https://en.wikipedia.org/wiki/First-class_function); a function is considered to be an object. As such, a function may have properties and methods, such as A function is a function defined within another function. It is created each time the outer function is invoked. In addition, each nested function forms a [lexical closure](https://en.wikipedia.org/wiki/Closure_(computer_programming)).

The call stack shrinks and grows based on the function's needs. When the call stack is empty upon function completion, JavaScript proceeds to the next message in the queue. This is called the [event loop](https://en.wikipedia.org/wiki/Event_loop), described as "run to completion" because each message is fully processed before the next message is considered. However, the language's [concurrency model](https://en.wikipedia.org/wiki/Concurrency_(computer_science)) describes the event loop

### 4.5.9 Delegative

### JavaScript supports implicit and explicit [delegation](https://en.wikipedia.org/wiki/Delegation_(object-oriented_programming)).

**4.5.10 Functions as roles (Traits and Mixins)**

JavaScript natively supports various function-based implementations of [Role](https://en.wikipedia.org/wiki/Role-oriented_programming) patterns like [Traits](https://en.wikipedia.org/wiki/Traits_(computer_science)) and [Mixins](https://en.wikipedia.org/wiki/Mixin). Such a function defines additional behavior by at least one method bound to the keyword within its body. A Role then has to be delegated explicitly via or to objects that need to feature additional behavior that is not shared via the prototype chain.

**4.5.11 Object composition and inheritance**

Whereas explicit function-based delegation does cover [composition](https://en.wikipedia.org/wiki/Object_composition) in JavaScript, implicit delegation already happens every time the prototype chain is walked in order to, e.g., find a method that might be related to but is not directly owned by an object. Once the method is found it gets called within this object's context. Thus [inheritance](https://en.wikipedia.org/wiki/Inheritance_(computer_science)) in JavaScript is covered by a delegation automatism that is bound to the prototype property of constructor functions.

**4.5.12 Run-time environment**

JavaScript typically relies on a run-time environment (e.g., a [web browser](https://en.wikipedia.org/wiki/Web_browser)) to provide objects and methods by which scripts can interact with the environment (e.g., a web page [DOM](https://en.wikipedia.org/wiki/Document_Object_Model)). These environments are single-[threaded](https://en.wikipedia.org/wiki/Thread_(computing)). JavaScript also relies on the run-time environment to

provide the ability to include/import scripts (e.g., [HTML](https://en.wikipedia.org/wiki/HTML) elements). This is not a language feature per se, but it is common in most JavaScript implementations. JavaScript processes [messages](https://en.wikipedia.org/wiki/Message_(computer_science)) from a [queue](https://en.wikipedia.org/wiki/Queue_(abstract_data_type)) one at a time. JavaScript calls a [function](https://en.wikipedia.org/wiki/Subroutine) associated with each new message, creating a [call stack](https://en.wikipedia.org/wiki/Call_stack) frame with the function's [arguments](https://en.wikipedia.org/wiki/Parameter_(computer_programming)) and [local variables](https://en.wikipedia.org/wiki/Local_variable). The call stack shrinks and grows based on the function's needs. When the call stack is empty upon function completion, JavaScript proceeds to the next message in the queue. This is called the [event loop](https://en.wikipedia.org/wiki/Event_loop), described as "run to completion" because each message is fully processed before the next message is considered. However, the language's [concurrency model](https://en.wikipedia.org/wiki/Concurrency_(computer_science)) describes the event loop as [non-blocking](https://en.wikipedia.org/wiki/Asynchronous_I/O): program [input/output](https://en.wikipedia.org/wiki/Input/output) is performed using [events](https://en.wikipedia.org/wiki/Event_(computing)) and [callback functions](https://en.wikipedia.org/wiki/Callback_(computer_programming)). This means, for instance, that JavaScript can process a mouse click while waiting for a database query to return information.

**Var**

An indefinite number of parameters can be passed to a function. The function can access them through [formal parameters](https://en.wikipedia.org/wiki/Formal_parameter) and also through the local object. [Variadic functions](https://en.wikipedia.org/wiki/Variadic_functions) can also be created by using the method.

This lets asynchronous methods return values like synchronous methods: instead of immediately returning the final value, the asynchronous method returns a promise to supply the value at some point in the future. Recently, combinator methods were introduced in the JavaScript specification, which allows developers to combine multiple JavaScript promises and do operations on the basis of different scenarios. The methods introduced are: Promise. race, Promise.

**Arr**

Like many scripting languages, arrays and objects ([associative arrays](https://en.wikipedia.org/wiki/Associative_arrays) in other languages) can each be created with a succinct shortcut syntax. In fact, these [literals](https://en.wikipedia.org/wiki/Object_literal) form the basis of the [JSON](https://en.wikipedia.org/wiki/JSON) data format.

Reg

JavaScript also supports [regular expressions](https://en.wikipedia.org/wiki/Regular_expression) in a manner similar to [Perl](https://en.wikipedia.org/wiki/Perl), which provide a concise and powerful syntax for text manipulation that is more sophisticated than the built-in string functions.

JavaScript processes [messages](https://en.wikipedia.org/wiki/Message_(computer_science)) from a [queue](https://en.wikipedia.org/wiki/Queue_(abstract_data_type)) one at a time. JavaScript calls a [function](https://en.wikipedia.org/wiki/Subroutine) associated with each new message, creating a [call stack](https://en.wikipedia.org/wiki/Call_stack) frame with the function's [arguments](https://en.wikipedia.org/wiki/Parameter_(computer_programming)) and [local variables](https://en.wikipedia.org/wiki/Local_variable). The call stack shrinks and grows based on the function's needs. When the call stack is empty upon function completion, JavaScript proceeds to the next message in the queue. This is called the [event loop](https://en.wikipedia.org/wiki/Event_loop), described as "run to completion" because each message is fully processed before the next message is considered. However, the language's [concurrency model](https://en.wikipedia.org/wiki/Concurrency_(computer_science)) describes the event loop

There is a huge demand for Front-End Web Developers or Web Designers in IT and Front-End Developer Jobs are also one of the high paid jobs. These all are the reason people love to choose this field. Frontend development is all about UI/UX where the main concern is related to layout, styling or designing of the website. Every web designer or frontend developer journey start from HTML and CSS and after a better understanding of both, they need to jump into JavaScript to add interactive features in your HTML and CSS design. When it comes to learning method for frontend development most of the people love to go with online videos or materials however sometimes for reference or to understand the concept in-depth from beginning till the end books are the best options.

This lets asynchronous methods return values like synchronous methods: instead of immediately returning the final value, the asynchronous method returns a promise to supply the value at some point in the future. Recently, combinator methods were introduced in the JavaScript specification, which allows developers to combine multiple JavaScript promises and do operations on the basis of different scenarios. The methods introduced are: Promise. race, Promise.

Below is the list of some best books to learn frontend development or web designing. These books are good for both entry-level and experienced developers. We have categorized these books into two sections- for beginners and for experienced. One thing you need to keep in mind that in programming you need to keep updated yourself but basics concepts always remain the same and they should have strong command on basics concepts first before moving on to the advanced level.

If you are a beginner in designing and recently started learning HTML and CSS then this book is best for you to learn everything from scratch to an expert level. The quality of content in this book is very high, also the presentation of everything is well organized. You will find the definition of every topic along with the code and its output. The author of this book has made code so visual so that everybody can understand it. Graphics are wonderful in this book and all the properties, examples, code is explained using a color-coding system to differentiate everything. It has magazine-style layouts and designs with high-quality pages.

Pro

JavaScript also supports [promises](https://en.wikipedia.org/wiki/Futures_and_promises), which are a way of handling asynchronous operations. There is a built-in Promise object that gives access to a lot of functionalities for handling promises, and defines how they should be handled. It allows one to associate handlers with an asynchronous action's eventual success value or failure reason. This lets asynchronous methods return values like synchronous methods: instead of immediately returning the final value, the asynchronous method returns a promise to supply the value at some point in the future. Recently, combinator methods were introduced in the JavaScript specification, which allows developers to combine multiple JavaScript promises and do operations on the basis of different scenarios. The methods introduced are: Promise. race, Promise. all, Promise. All Settled and any.

This lets asynchronous methods return values like synchronous methods: instead of immediately returning the final value, the asynchronous method returns a promise to supply the value at some point in the future. Recently, combinator methods were introduced in the JavaScript specification, which allows developers to combine multiple JavaScript promises and do operations on the basis of different scenarios. The methods introduced are: Promise. race, Promise.

### You will be learning about HTML, CSS, JavaScript as well as web graphics and responsive web designing in this book. Title of this book mentions that the book is for beginners but most of the beginner find that it has advanced topics so if you already have basic knowledge of web designs then this book is best for you to have in-depth knowledge in web designing. If you are a beginner, we recommend you to learn basics first then pick up this book.

### Vendor-specific extensions

Hist [JavaScript engines](https://en.wikipedia.org/wiki/JavaScript_engine) supported these non-standard features:

* conditional clauses (like Java)
* [array comprehensions](https://en.wikipedia.org/wiki/List_comprehension) and generator expressions (like Python)
* concise function expressions (this experimental syntax predated arrow functions)
* [ECMAScript for XML](https://en.wikipedia.org/wiki/ECMAScript_for_XML) (E4X), an extension that adds native XML support to ECMAScript (unsupported in Firefox since version 21)

**CONCLUSION**

In a nutshell, this internship has been an excellent and rewarding experience. I can conclude that there have been a lot I’ve learnt from my work at the training. Needless to say, the technical aspects of the work I’ve done are not flawless and could be improved provided enough time. I believe my time spent in training and discovering new languages was well worth it and contributed to finding an acceptable solution to an important aspect of web design and development. This “Training” was a way to practice and get a little command over our interests. I would like to thank our professors for giving us an opportunity to build and polish some technical skills.

The emergence of responsive web design, more than six years ago, showed us the vision for our future sites: a world where users can have great experiences no matter what devices or what screen sizes, they have those experiences on. However, it was also clear to us that this change was not going to happen without effort—and a tremendous one, at that. The beautiful thing about responsive retrofitting projects —and what can also make them look so scary—is the number of improvements that can be validated by the need to make your site’s experience better on any device. As professional web designers and developers, we often know perfectly well what our sites need in order to be greatly improved. But sometimes the task ahead seems too complex, too riddled with obstacles,

Through this hands-on project I was able to see how well you understood the material presented, and your critical thinking as you developed the restaurant site. You were also able to show me your creative side, through the layout and design of your project. As a team you both were able to use your problem-solving skills to troubleshoot errors when something did not work the way you intended. Knowing basic HTML is a skill that will help you with blogging, working with Wiki's, Facebook, and much more

In the scope of this study, we intend to explore the main advantages and limitations associated with responsive Web design. We adopted a quantitative approach based on a questionnaire filled by 181 professionals in the industry that allowed us to identify the reasons that lead software developers to the adoption of the responsive design and also address the limitations felt by them. The results obtained indicate that offering a good user experience and increasing accessibility stands out as being the most important advantages. On the other hand, the main limitations include the compatibility with older Web browsers, the higher loading time and the difficulties in optimizing user experience. Finally, it was found that the perception of the advantages and limitations of responsive design is distinct for professionals with more professional experience in the field and for freelancer developers.

**FUTURE SCOPE**

As we head towards an even more tech-driven future, the development skills are progressively in demand. Artificial Intelligence, Machine Learning, Internet of Things (IoT), quantum computing, etc. are changing the tech world. And this even comes down to their relationship with the websites. Some of the major future aspects of using this technology are:

Artificial intelligence does not need to be introduced. Every article on technology, which we read on the web, reflects the need for AI and how it would be the next thing on the web. Several companies and institutions are already implementing AI to meet their needs. Some have reaped the benefits of AI for their businesses.

The Internet of Things, or IoT, refers to the billions of physical devices, objects, animals, or individuals with unique identifiers that are distributed across the network are now connected to the Internet, all of which collect and exchange data, without requiring human-to-human or human-to-computer interaction.

Virtual reality is the long-awaited dynamic that many thoughts would take the world by storm, affecting everything in its path. Virtual reality is designed to affect many different elements of web design for a variety of different reasons. VR is sculpting a web path and is set to affect web design in many interesting new ways.

The 360-degree video offers a significantly enhanced user interface that leads to the creation and growth of websites. The 360 degrees are guaranteed to draw more attention, thus enhancing interaction for more views. It enriches the user’s experience as if he or she is genuinely present. The video clips cover a view of the spherical scenario in 360 degrees, while the camera captures a view of all possible angles to a seamless future experience. The ultimate objective is to provide end-users with high-tech features.

An example of artificial intelligence implementation. This wasn’t so amazing before, because this technology just worked poorly. High-quality voice input items were extremely expensive. But it’s one of the trends now. In several smartphone and web applications voice commands are actively implemented. You can call Uber, too, by voice. You don’t have to spend time using your application, so life is much easier.

Motion design helps to make user interfaces attractive and intuitive. When correctly used, motion can guide your digital product user experience by conveying a message about your brand. To add value and enhance many aspects of their interaction, you can align animation to the preferences of your users.

It becomes easier to create a 360-degree video experience if you have sufficient and enough experience in VR technology. If you have experience and expertise as a web developer with the new and best modern technologies, it helps to increase the visibility of a company’s brand.

Some MNCs and giants, such as Google and Microsoft, have already implemented virtual reality or web VRs. Virtual reality can simply be defined as the process of implementing computer technology to create a stimulating environment. VR can put users into an experience that cannot be observed in any of the existing traditional interfaces. Instead of accessing a screen placed in front of them, users can engage and interact with the 3D world in real-time.

**REFERENCES**

**1. Other Sources**

[1] Solution of Problem - www.codewithharry.com

**2. Websites**

[1] [www.greatlearning.com](http://www.greatlearning.com)

[2] [www.w3school.com](http://www.w3school.com)

[3] [www.geeksandgeeks.com](http://www.geeksandgeeks.com)

[4] [www.stackoverflow.com](http://www.stackoverflow.com)

[5] www.codeorg.com

[6] www.tutorialpoint.com

**3. Books on Web Development**

[1] Introduction to HTML and CSS

[2] Design and Build by Jon Duckett

[3] Eloquent Java Script