**FRONT-END TUTORIALS WEEK 01**

**The Web & The Internet**

**What Is Website Development?**

Website development refers to the work that goes into building a website. This could apply to anything from creating a single plain-text webpage to developing a complex web application or social network.

While web development typically refers to web markup and coding, it includes all related development tasks, such as: Client-Side Scripting, Server-Side Scripting, Server and Network Security Configuration, E-commerce development, and Content Management System (CMS) development.

**Why is web development important**?

You might be a business owner hiring a freelance developer to build your website, a marketer pitching a vision to your development team, or a student learning about development as a career. Regardless of who you are or why you’re reading this guide, understanding the basics of website development can be helpful in this technology-driven world.

The internet isn’t going away anytime soon. In fact, it’s become a portal and primary method of research, connection, education, and entertainment in the world. As of 2022, there were 5 billion global internet users. That’s more than half the world’s population, and these folks are using the internet for a vast variety of reasons.

What’s the one thing those reasons have in common? They require a website, and each website requires a skilled web developer. Web development is also a rapidly expanding industry. Between now and 2028, the employment of web developers is expected to grow by 13%. That’s much faster than most other technology careers.

**Web Development Basics**

● What a Website Is

● What an IP Address Is

● What HTTP Means

● What Coding Is

● What Front-end Means

● What Back-end Means

● What a CMS Is

● What Cybersecurity Is

1. **What a Website Is**

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).

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.

1. **What an IP Address Is**

Internet Protocol is a set of standards that govern interaction on the internet. To access a website, you need to know its IP address. An IP address is a unique string of numbers. Each device has an IP address to distinguish itself from the billions of websites and devices connected via the internet. To find your device’s IP address, you can also type “what’s my IP address” into your search browser.

While you can access a website using its IP address, most internet users prefer to use domain names or by going through search engines.

1. **What HTTP Means**

HyperText Transfer Protocol (HTTP) connects you and your website request to the remote server that houses all website data. It’s a set of rules (a protocol) that defines how messages should be sent over the internet. It allows you to jump between site pages and websites.

When you type a website into your web browser or search for something through a search engine, HTTP provides a framework so that the client (computer) and server can speak the same language when they make requests and responses to each other over the internet.

It’s essentially the translator between you and the internet — it reads your website request, reads the code sent back from the server, and translates it for you in the form of a website.

1. **What Coding Is**

Coding refers to writing code for servers and applications using programming languages. They’re called “languages” because they comprise vocabulary and grammatical rules for communicating with computers.

They also include special commands, abbreviations, and punctuation that can only be read by devices and programs. All software is written by at least one coding language, but languages vary based on platform, operating system, and style.

There are many different types of coding languages…all of which fall into two categories, front-end and back-end.

1. **What Front-end Means**

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.

1. **What Back-end Means**

Back-end (or server-side) is the side that you don’t see when you use the internet. It’s the digital infrastructure, and to non-developers, it looks like a bunch of numbers, letters, and symbols. There are more back-end coding languages than front-end languages.

That’s because browsers — at the front-end — only understand JavaScript, but a server — at the back-end — can be configured to understand pretty much any language.

1. **What a CMS Is**

CMS is a software application that provides a user-friendly interface that enables users (content authors, editors, and publishers) to easily create, modify, and organize content, and publish it on a website or other digital platform.

Some popular examples of CMS platforms include WordPress, Drupal, and Joomla. While not required to build a website, using a CMS makes things easier.

It provides the building blocks (like plugins and add-ons) and lets you create the structure with your code. CMSs are typically used for e-commerce and blogging, but they’re useful for all types of websites**.**

1. **What Cybersecurity Is**

As long as the internet exists, there will be users looking to find vulnerabilities in websites to expose private information, steal data, and crash servers. Cybersecurity is the practice of securing data, networks, and computers from these threats. The methods used by hackers are constantly evolving, as are the security measures taken to defend against them. Failing to understand how your site could be targeted could result in disaster.

**Building Blocks Of A Website**

1. **Write your website code**

Developers will use different coding languages for the front-end and back-end of websites, as well as for different functionalities of the site (such as design, interactivity, etc.) These different languages work together to build and run your site. Let’s start with the most commonly-used languages. Almost every website uses these three together, and yours probably will, too.

**● HTML HyperText Markup Language (HTML)**

has been used since the 1990s. It’s the foundation of all websites and represents the bare minimum of what’s needed to create a website. (Yes, you can create a website with only HTML. It wouldn’t look too pretty, though.) Languages like CSS and JavaScript enhance and modify the basic site structure built by HTML codes. HTML5 is the most recent version and supports cross-platform browser functionality, making it popular in mobile application development.

**● CSS 6 Cascading Style Sheets (CSS)**

was developed in the late 1990s. It adds design elements like typography, colors, and layouts to websites — it’s the cosmetic code. CSS allows developers to transform your website to match the aesthetic you envisioned for your site, and like HTML5, CSS is compatible with all browsers.

**● JavaScript**

JavaScript is the cherry-on-top of coding languages. Created in the mid-90s, JavaScript is used to add functionality to websites. Developers use it to add animations, automate tasks within certain pages, and add interactive features that enhance user experience. JavaScript is rapidly evolving.

Once considered a “toy” language, JavaScript is now the most widely used coding language in the world. With the help of Node.Js,

it’s now a back-end coding language. It’s the first language to be understood by browser, and some have even discussed applying machine learning to it, too. HTML, CSS, and JavaScript are the “big three” of web development.

Almost every website uses them in some capacity. There are plenty of others, such as server-side languages like Java, C++, Python, and SQL, but understanding these three is foundational to your website development knowledge.

**The front-end of your website**

If you’ve ever dabbled in web design or toyed with a website in WordPress, Squarespace, or Google Sites, you’ve touched front-end web development.

The front-end stuff is important — it’s what your visitors, customers, and users see and how they’ll use your website. Front-end (or client-side) development includes a combination of JavaScript, HTML, and CSS.

It also controls components such as typography and fonts, navigation, positioning, and browser compatibility and responsiveness. This part will reflect more of your initial site vision and what you included in your wireframe. As technology and consumer preferences change, client-side coding tends to become outdated … a lot faster than back-end development does. This is where coding resources (like the ones we’ve included below) come in handy.

**The Back-end of your website**

Writing code might be one of the more complicated parts of web development, but it’s hardly the only component. You also have to build your back-end and front-end site structures and design. Let’s start with the back-end. The back-end handles the data that enables the functionality on the front-end.

For example, Facebook's back-end stores my photos, so that the front-end can then allow others to look at them. It’s made up of two key components:

Databases, which is responsible for storing, organizing, and processing data so that it’s retrievable by server requests Servers, which are the hardware and software that make up your computer. Servers are responsible for sending, processing, and receiving data requests.

They’re the intermediary between the database and the client/browser. The browser will, in effect, tell the server "I need this information", and the server will know how to get that information from the database and send it to the client.

These components work together to build the foundation for each website. As for building your website, backend developers will establish three things.

● Your logic code, which is essentially a set of rules for how your website will respond to certain requests and how objects of your website will interact.

● Your database management, which is how your website will organize, manage, and retrieve its data.

● Your infrastructure, which is how your site will be hosted. Hosting your own site will give you greater control, but it’s much more expensive and requires you to maintain your own server health and security.

With these components and decisions in place, your website will be ready for front-end development. Note: The back-end is slightly tangential to web development because you don't always need a back-end if you're not storing any data. “Data” in this context means any user-entered information that you need to save and persist. Think about logging in to a website.

If they don't have a back-end, how could they remember your login information? Or what your profile settings are? To get this information, you need a back-end. Facebook, as an example, needs to know what people are in your Friends list, what events you have joined, what posts you have created, and more.

This is all "data" that lives in a database. If they didn't have a back-end with a database, none of that data would be accessible to them. On the other hand, a website that’s purely informational and doesn’t require the users to enter any data wouldn’t need a back-end. So, if you have no data, you don’t necessarily require back-end development. But that’s not saying you shouldn’t learn the basics. You never know when you might need it.

1. **Acquire a domain name**

At this point, your website will have an IP address. It also needs a domain name, a memorable website name that your visitors can use to find your site.

Perhaps you’ve heard of sites like GoDaddy and Hover. These services help you purchase a domain name and register with ICANN (Internet Corporation for Assigned Names and Numbers). Most domain registrations are good for a year before you’re required to renew. Website builders and hosting services, like WordPress and Squarespace, also allow you to purchase a domain name.

1. **Launch your site**

Once you’ve set up a domain name and linked it to your host, you’re almost ready to unveil your work to the web. But, not so fast — there are still a handful of things you’ll need to check before an official launch.

These include planning out responsibilities on your team, testing your site thoroughly for any glitches, optimizing for SEO, and a final check before “flipping the switch” and making your site live.

**Basic Command-Line Interface (CLI) commands**

**you should know**

Command Line Interface (CLI) or Terminal, is considered by many to be the Holy Grail of computer management.

At one time, the CLI was the only way to accomplish anything on a computer; then, the CLI gave way to the graphical user interface (GUI) as the popularity of PCs increased. The purists among us often prefer to use the CLI as a means of manipulating the computer and getting it to perform tasks instead of using a mouse to get things done.

The Command Prompt (cmd.exe) and PowerShell are both command-line interfaces (CLIs) in Windows. PowerShell provides a more advanced and flexible command-line experience for Windows users, but the Command Prompt may still be useful for basic tasks and in situations where compatibility with older systems is a concern.

1. **Change Directory Command: cd**

This command will change the directory that you’re currently working with in the Terminal in order to execute other commands on a different directory, view the contents of a different directory or open a file in a different directory.

This is a very common command that will be used when working with the CLI. If you ever lose your place and which directory you’re in, type pwd (print working directory) and press Return to echo the current path. If you want to move back up to the previous directory, you can use two dots: cd ..

1. **Listing Directory Command: dir**

In the terminal, the "dir" command is used to list the contents of a directory in a file system. It is commonly used in Microsoft Windows systems and is equivalent to the "ls" command 11 in Unix-based systems. The "dir" command can display various information about the files and directories, such as the names, sizes, creation dates, and permissions. Windows systems have a PowerShell environment that provides many Unix-like commands, including the "ls" command.

The "ls" command in PowerShell works similarly to the "ls" command in Unix-based systems, allowing Windows users to list the contents of a directory in a similar way. By using "ls" in PowerShell, Windows users can access the benefits and familiar syntax of the Unix-style command, even on a Windows system.

1. **Remove a file Command: rm**

This command removes the specified file. For example, rm mdn-example.txt deletes a single file called mdn-example.txt. Note that this delete is permanent and can't be undone via the recycle bin that you might have on your desktop user interface.

1. **Move a file Command: mv**

When you don’t want to copy a file, but instead move it, use the same format of the cp command, but instead replace ‘cp’ with ‘mv’. This will perform a file move from one location to another, removing the file from the original location and putting it into the new location. Example: mv “filename” “path/to/new/file/location”

1. **Create a directory Command: mkdir**

This command will allow you to create a directory (folder) right from the CLI. When you need a place to store new files, just use this command to add a new directory in the current 12 working directory, or specify a full path to the location where you want the new directory to be placed. Example: mkdir “path/to/new/directory”

1. **Remove an empty directory Command: rmdir**

When you’ve created a folder that has the wrong name, you can easily use the mv command to rename it, or if you want to remove a directory altogether, use the rmdir command followed by the path to the directory. Example: rmdir “path/to/directory” (Figure H)

1. **Clear the Terminal screen**

of all previous commands Command: clear It removes all previously typed commands from the Terminal view and gives you back a clean slate to work from. Instead of typing clear, you can also alternatively press Command+K to perform the same action.

**8. The “exit” command Command: exit**

This command will close out the current session in the Terminal. Note: A very useful terminal shortcut is using the tab key to autocomplete names that you know are present, rather than having to type out the whole thing. For example, after typing the above two commands, try typing cd D and pressing tab — it should autocomplete the directory name Desktop for you, provided it is present in the current directory. Bear this in mind as you move forward.