Published in codeburst

You have 2 free member-only stories left this month. Upgrade for unlimited access.

Emma Bostian
May 22, 2018 · 8 min read ★ · Listen

Web Development For Beginners

You can now subscribe to get stories delivered directly to your inbox.
Got it

We all use the web every single day, but how many of us actually know how it works or the skills necessary to become a web developer?

There are many wonderful jobs you can have as a front-end developer (FED). And it's becoming more commonplace for companies to hire FEDs with either no college degree or an unrelated (to computer science) college degree.

What's so wonderful about this career is that **anyone can learn to become a FED** without shelling out major money for it.

Today we'll dive into the basic skills required to become a web developer.

This article is meant to be a high-level guide for beginners to web development. Feel free to skip sections you're already familiar with.

The Web

https://images.techhive.com/images/idge/imported/article/itw/2014/03/14/20140313_www_birthday-small-100480259-orig.jpg

Our story begins in 1989 with a man called Tim Berners Lee. A British Computer Scientist, Berners Lee set out to solve one major problem: how can we share and manage information throughout the world?

At the very core of it, the Internet is just a series of computers connected by millions of cables. In contrast, the web consists of billions of digital pages that are viewable through web browsers on your computers.

Clients & Servers

We have these things called clients and servers that make the web possible.

Clients are essentially your devices. Anything you (the user) uses to connect to the internet is a client.

Servers are big computers that store web pages, data, etc. So, when you visit a website, your device talks to the server to get the web page you requested.

DNS stands for Domain Name Servers, and their only job is to take in a web address and retrieve the real website address. It's essentially a big address book.

HTTP, or Hypertext Transfer Protocol, defines the language for clients and servers to communicate with.

TCP/IP, or Transmission Control Protocol/Internet Protocol, define how data should travel through the web.

Let's walk through the scenario of what happens when you type a website into your browser and hit Enter.

1. Your browser heads over to the **DNS** server to find the real address of the server your requested website lives on. You can think of the address you type in to the search bar as the street address, and the address returned from the DNS server as the geographic coordinates of the same building. URLs (Universal Resource Locators) are useful for humans, because they're easily memorizable. However, computers connect to other computers using IP addresses that look like this: 63.245.215.20.
2. Next, your browser sends an **HTTP** request to the server asking if it can send a copy of the website back to your client device. The messages sent between the client and server are communicated through **TCP/IP**.
3. Once the server approves the request, it sends a "200 OK" message, indicating that you're allowed to view the website. If something goes awry, you'll receive a different response code back. One of the more common responses you might get is a "404 Not Found" error.
4. Lastly, the browser assembles the bits (chunks) of website into a full web page and displays it to you. When data is sent across the web, it's sent in packets; small chunks of r[illegible]t many users can access the same content at the same [illegible]

773 | 5 | ...

Get unlimited access

Search

Emma Bostian
5K Followers

Software Engineer @ LogMeIn by day, cat Mom by night. Also a full-time Bibliophile. I enjoy all things Front-end & Design

Follow

More from Medium

Sumsha
Top front-end technologies for web development

Matt Studdert in Frontend Mentor
Frontend Mentor January 2022 review

Ben Merchant
Day04 - #100DaysOfCode - React State

Duy K. Nguyen
[Tip For Developer] Surfing the website that needs custom-host without manually customizing...

Help Status Writers Blog Careers Privacy Terms About Knowable

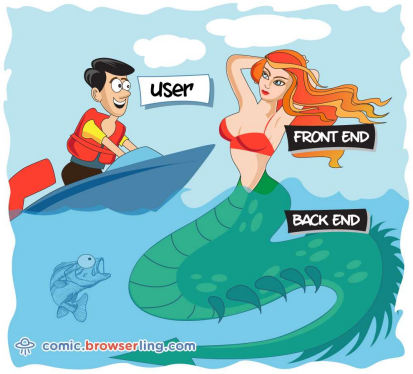
If you want a more in-depth explanation of the web, check out this awesome post by Mozilla.

Now that we have a *general* understanding of the web, let's dive into front-end development.

What Is Front-end Development?

Front-end Development is responsible for developing the face of the web page that users interact with. Front-end development also functions as a bridge between design (research, user experience, visual) and back-end development.

Back-end Development is responsible for creating the computational logic behind a website or application.



Front End vs Back End

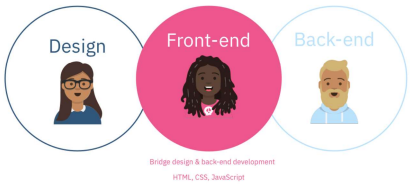
Each area of development (front-end and back-end) requires specialized skills. For example, a back-end developer might use Java, or C++, whereas a front-end developer uses HTML, CSS, and JavaScript.

For developers looking to dabble in both front-end and back-end, we call them "Full-stack" developers. Full stack developers must have knowledge of all areas of the stack. This position is much more difficult to master (as it requires expansive, and diverse, knowledge), but the upside is that it can pay a lot more in salary.

The Importance Of Design

We often forget how vital design is throughout the application development process. What's great about front-end development is that it allows you to bridge the gap between design and back-end development.

Thus, it's important for front-end developers (FEDs) to care about best UX and visual design practices.



At a high-level there are a few areas of design to be aware of:

Visual Design is responsible for producing the branding for a website. They have deep knowledge of typography, iconography, color, space, textures, and more.

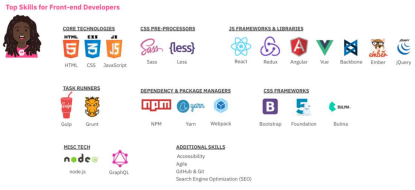
UX Design is responsible for enhancing users' experiences within a product by accounting for usability, accessibility and interactions.

UX Research is responsible for investigating users and their requirements to design the most optimal solution.

Now that we have a *general* understanding of design, let's move into FED.

FED Skills

Front-end development can be overwhelming with the amount of technologies, frameworks, and libraries to learn. These are some of the top front-end skills you should have as a FED, broken down into categories.



Top FED Skills

What Can I Do With Front-end Development?

There are many roles you can take on as a front-end developer. Here are a few of the most popular titles you might see. Each role's domain will shift slightly, but generally the same thought process is required for each.

Job areas within FED

HTML, CSS, and JavaScript

Let's think about a website as a human. We can think about HTML, CSS, and JavaScript respectively as the following:

HTML: The skeleton. HTML is responsible for the structure of content in a web page.

CSS: Hair color, eye color, height, weight, etc. CSS is responsible for the visual presentation of a website.

JS: Movements, personality. JavaScript is responsible for interactions with page elements, making database requests, etc.

These are the core languages that power the face of your websites, so it's important to fully understand each of them.

I'll be writing a more detailed blog post on each of the three languages above, so stay tuned for that. In the meantime, check out the links below for helpful tutorials.

A Note on JavaScript Frameworks

Many front-end job postings will add the requirement of knowing a popular JS framework, such as Angular (slightly outdated), React, or Vue. I'm not arguing that you shouldn't learn a framework, or library, however I caution you to learn the basics of JavaScript prior to diving into these.

I got my current job by completing a take-home assessment only using "vanilla" (no framework) JavaScript. So while the ability to work with these popular libraries is an important skill to have, deep JavaScript knowledge is much more valuable in the long-run.

Where Do I Start?

Front-end development can be overwhelming when you're starting out. So I've outlined a path to get started that I hope can help remove some of that angst.

1. Learn about the web
2. Learn HTML
3. Understand the DOM
4. Learn CSS
5. Learn JavaScript
6. Learn JSON, APIs, & REST/GraphQL
7. Learn the CLI
8. Learn workflow tools
9. Learn Node.js
10. Learn About UX & Design

FAQ

Below are some questions I had when I started learning front-end development. If you have more questions, feel free to leave them below!

Do I need to know back-end development to make a website?

No! Unless you want to make a fully-functioning, scalable (client-facing) web application, you do not need to have a database. When I need data, I either grab a free API, or use a JSON file for my data.

Won't this take a lot of time to learn?

Yes, but it's an ongoing process and you should always keep learning.

What is JavaScript vs. ES6 vs. ES2015?

They're all the same thing. ES stands for ECMAScript, which is the scripting language specification. You can use these interchangeably (although the version number will change as new JS versions are released).

How can I pass a technical interview?

There are a ton of resources to help you with technical interviews. I'll list a few of my favorites below.

How can I find a front-end job?

There are a ton of ways to find front-end jobs. I'll link a few of my favorites below.

Do I need to learn a framework/library?

See my note above about frameworks and libraries.

Which framework/library should I learn?

Check out this post about popular JS frameworks. I'd recommend React (with Redux) for now!

Which IDE is right for me?

There are many popular front-end development environments. Some of my favorites are Atom, Sublime, and of course VisualStudio Code. But paid IDEs like Webstorm are also powerful.

Do I need to know how to write an API?

You should learn this, but not in the beginning of your FED journey.

How do I get my website online?

I'll be writing another post about getting your website online, but for now, check out this post. You'll need a domain name, a host, and an FTP client.

Do I need to make a mobile app for my website?

Yes, since the majority of websites are accessed on mobile devices, however you can make your site responsive with media queries, or use progressive web apps to do this.

How Can I Learn FED?

Online Tutorials

Front-end Masters ($)

Lynda ($)

Coursera ($$)

Level Up Tutorials

Traversty Media

CodeAcademy

Blogs

CSS Tricks

JavaScript Weekly

CSS Weekly

Books

You Don't Know JS

Front-end Developer Handbook

Eloquent JavaScript

JavaScript: The Good Parts ($)

JavaScript Allongee

Technical Interviews

Cracking The Coding Interview 5th Edition (this is outdated, but a free PDF)

Cracking The Coding Interview 6th Edition ($)

Coderbyte

CodeFights

HackerRank

Front-end Interview Handbook

Front-end Interview Questions

Job Search

LinkedIn

Monster

Indeed

Hired*Also check out the company's website for open positions.*

Tools

CodePen

Code Sandbox

If you'd like to know more, follow me on Twitter, LinkedIn, or Instagram!

codeburst.io

Subscribe to *CodeBurst's* once-weekly **Email Blast**, Follow *CodeBurst* on **Twitter**, view **The 2018 Web Developer Roadmap**, and **Learn Full Stack Web Development**.