LEARN TO CODE IN 2021, GET HIRED, AND HAVE FUN ALONG THE WAY

Andrei Neagoie

V1.01

HEEELLLOOOOO!

I'm Andrei Neagoie, Founder and Lead Instructor of the Zero To Mastery Academy.

After working as a Senior Software Developer over the years, I now dedicate 100% of my time teaching others valuable software development skills, help them break into the tech industry, and advance their careers. In only two years, **over 200,000 students** around the world have taken my courses and many of them are now working at top tier companies like **Apple, Google, Amazon, Tesla, IBM, UNIQLO**, just to name a few.

This guide provides step by step instructions on how to become a web developer from having zero knowledge… **for free**. By putting in the work, you'll have the opportunity to take control of your life, work in an exciting industry with infinite possibilities and live the life you want.

Happy Coding!
Andrei

Founder & Lead Instructor, Zero To Mastery
Andrei Neagoie

P.S. I also recently wrote a book called Principles For Programmers. You can download the first five chapters for free here.

Preface

For the past several years, I have been writing a guide (this is last year's edition) that goes viral every year which gives you step by step instructions on how to become a Web Developer from scratch, **for free.** Thousands of students have followed these steps since then and have gotten hired. However, a lot has changed since last year's edition, so I wanted to share with you the updated guide and changes for 2021 (with new resources)! The focus is on efficiency: Learn the right topics that are in demand right now so you can get hired as soon as possible.

These are the steps that you should be taking if you want to learn to code in 2021, change your career, and become a Web Developer (or get into the tech industry).

This is **part 1** of a 2 part series. You can read the second part here.

If you are a complete beginner, junior developer, or are curious about this industry, this post is for you. However, if you are an established developer, you may find some useful links in here as I list the best free resources to supercharge your skills, but I also wrote a post on how to become a senior software developer that may be more useful to you.

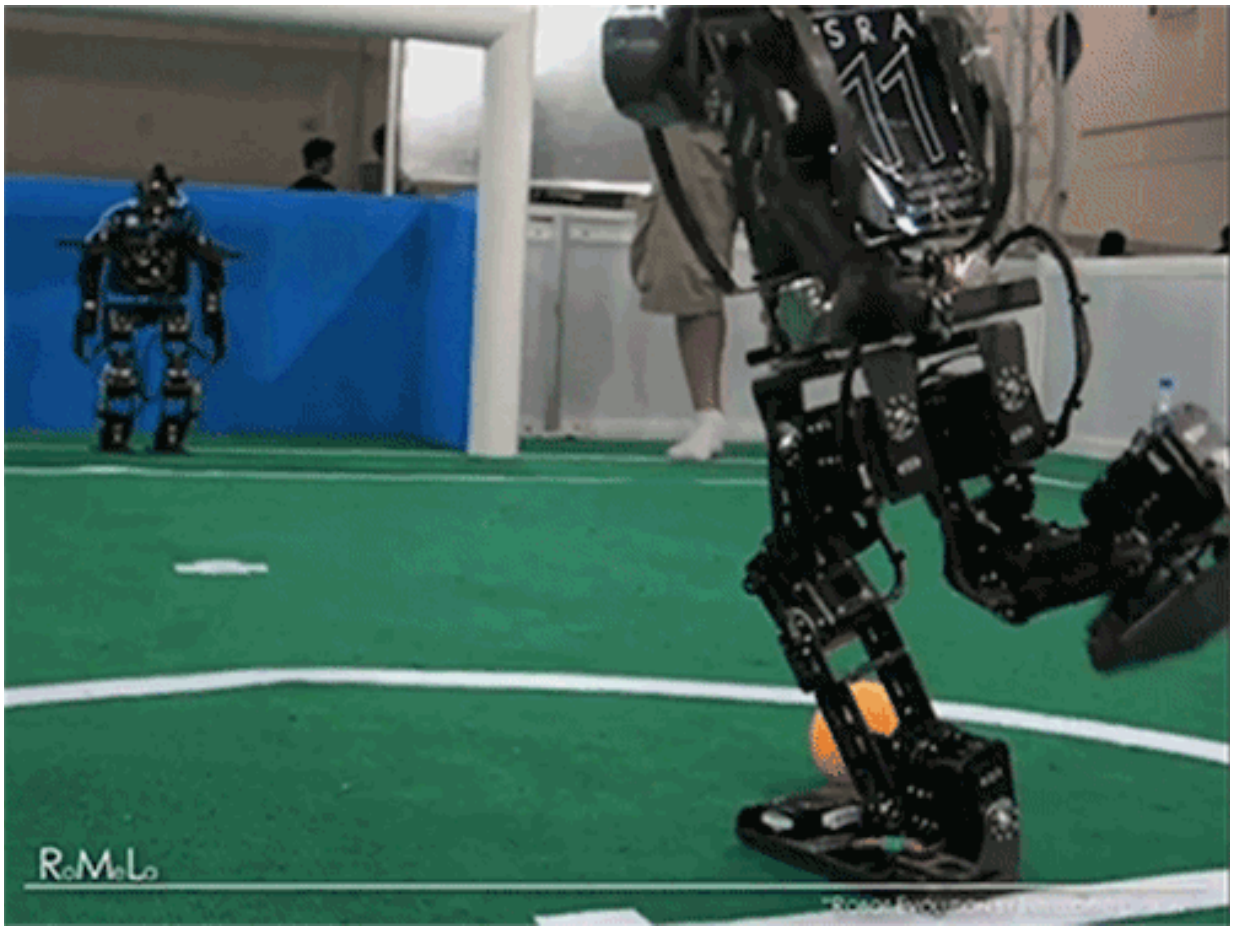
If you find this post too long, you can skip over and start from the **5 Months, Step By Step Section**. But you'll hurt my feelings…so you know, you can live with that guilt.

> ***Ok you're still here. Great! I like you already. Let's keep going…***

Using only free online courses, tutorials and free tools, you can gain a valuable skill that will allow you to be employed in a great industry that is rewarding, challenging, and with a lot of options to move around the world (more on this later). Best part? You don't need a college degree or an expensive bootcamp. Nor do you need to give away part of your income once you get hired which some new schools are doing (which sounds great until you have to start giving away some of your paychecks).

Important note: The post may seem like a step by step guide of what to do to become a developer, but if you look closely, it is a strategy that you can apply to any sort of learning.

Why coding?



One day you can build the best soccer goalie in the world…

Before we get into the steps you can take to become a developer, we must first dive into why you would want to go down this path. Every decision that will require significant time of your life should be justified. Time, after all, is the most important resource we have:

A. You want to be working in an industry where there is a high demand for the skill and many possibilities to be in important roles at the top of the food chain.

B. You love being location independent. You want a skill that allows you to go anywhere in the world and still be able to find a job easily. If you decide to move to Iceland tomorrow, you want to make sure that you won't have issues finding a job.

C. You've noticed the difference between 2010 and 2021 and how much of a technological progress we have made in those short 10 years. You want to be at the forefront of an industry that is impacting the world.

D. The biggest industry growth in the last couple of years has been in the artificial intelligence (Machine Learning), bio tech, autonomous cars, blockchain (Bitcoin) space. We interact with technology on the daily, and you don't want to be left behind in the dust as these take over our future. You want to understand and be able to pick up the skills underlying all of these: Programming. Web Development is a great foot in the door to these industries.

E. You think change is good, and learning should never stop. So why not do something new?

> ***But I don't have a computer science degree and I don't even know how the internet works! Don't worry, we will use that to your advantage. Keep reading…***

When choosing a new career path here are some good must/nice to-haves:

1. It must be relevant for the next 10+ years. This skill should be valued many years in the future guaranteeing you job security.

2. Demand for people with this skill must be higher than the supply. The less available pool of skilled workers in the industry, the more control you can have over your job and companies you work for.

3. Ability to have a high salary regardless of years in the industry. You don't want to spend many years climbing the corporate ladder until you make a decent living.

4. An industry that doesn't require a specialized degree from a university. You don't want to spend the next 4 years getting into debt and going to a graduate program before you start making money. And yes, I think there are better alternatives than going to an expensive coding bootcamp.

5. Ability to catch up to the top performers in the industry in the shortest amount of time. Can little experience still get you employed? And can you close the gap as fast as possible to be considered a senior or an expert in the field?

6. It must allow you to build foundational skills that will give you multiple career options no matter what the future holds. For example, by learning to code, you're able to better understand new and up-and-coming technologies like distributed applications, data science, machine learning (AI), and cloud computing, and choose which field you want to jump into next.

7. Have fun. The most important one. Can you see yourself doing this 40 hours a week for a long time?

Coding hits every one of the points above in my experience. Your mileage may vary.

One of my favourite books is titled So Good They Can't Ignore You. In it, the author argues that passion is a myth. You shouldn't go into the travel industry because you are "passionate" about travel. Most people find passion by struggling and working hard to master a skill. Once people start acknowledging your valuable skills, and you are able to feel respected for these skills, that's when you develop passion for what you do.

> ***Still with me? I haven't scared you off? Ok, we shall keep going then…***

IMPORTANT POINT, READ IT: Keep in mind that the first 2 months will feel like you are climbing an insurmountable mountain. Every tutorial, course or lesson you do will make you feel like you are the only person in the world that doesn't know this stuff. Stay strong. You will get there and you will have more and more 'AHA!' moments as time progresses. We call this the Impostor Syndrome: You feel like you are the only one who doesn't know this information and you are surrounded

by self-doubt. Rest assured we all feel this way when we learn something new. This is good. This is how we know we are stretching our boundaries.

What you will learn at the end of it all is that being a good developer isn't necessarily memorizing a whole bunch of documentation. It's about learning how to solve problems using all of the tools that are available to you. It's about being a problem solver and getting from a state of not knowing to knowing. This guide will help you get those skills.

Who are you and why should I listen to you?

Always wave back…

Wow, you're direct, but I guess that's a fair question. First off, I'm a Senior Software Developer that has worked in various locations including Silicon Valley and Toronto at some of the top tech firms. I've been very fortunate in my career and for the past 3 years I have taught 400,000+ people around the world how to become developers from scratch. Some of those people now work at companies like Google and Amazon. But I wasn't born a computer wiz. I didn't graduate with a computer science degree. I am completely self-taught.

> *P.S. This part is all about me, so if you don't care (totally fair point), just skip this section. I'll get over it eventually.*

It all started many years ago… I wanted a career change and decided to teach myself computer programming.

I spent the first month avoiding any tutorials or books. Instead, I spent this month looking at the best way for me to learn and get hired. I wanted to be efficient, not waste my time and learn outdated technologies, or learn things that I would forget after a month. I studied other people's experiences, looked at job postings, spoke to established developers, reviewed online courses, looked at bootcamps, and even read articles by futurists on where we will be with technology in 20 years. Based on those, I created a curriculum for myself focused on efficiency: **The critical amount of learning in order to be employable in the shortest amount of time.**

If you love the works of **Tim Ferriss** as much as I do, you're going to love this. The curriculum isn't focused on doing the least amount of work. Instead, it is focused on working really hard at the things that matter most in order to be employed in the optimum way. This doesn't mean doing the bare minimum and being hired as a junior developer. If you can work hard and skip the line by jumping straight into an intermediate developer role, that is a better outcome. Luckily for you, I have already sifted through everything for you.

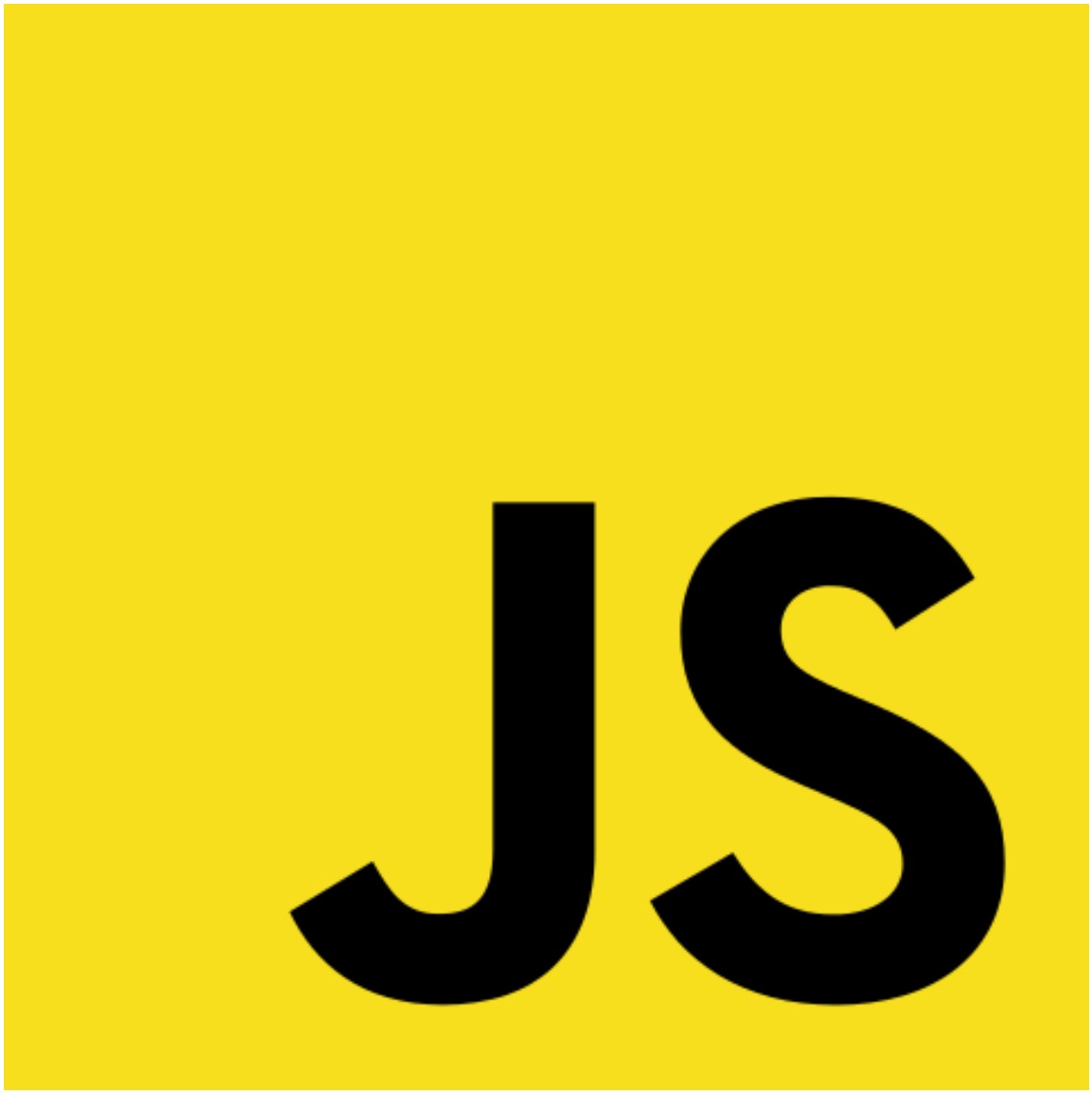
Although I spent one month planning my studying instead of actually studying, it was a benefit in the long run because I wasn't running blind. I knew where I was going, and I had a map to the finish line. You will too.

So yes, I have been where you are and I know what it takes. When I was getting started, I wish there was something like this that outlined things for me step by step. I also found many tutorials were taught by people with a lot of technical knowledge but without being able to properly teach a beginner. Alternatively, some courses were taught by people who took advantage of beginners not knowing much about the industry and selling them a course that sounds great but doesn't actually teach you how to succeed (we call these superficial skills). I've read and studied every single video, tutorial and course that time permitted, and I still continue to do so to try and find the most efficient path to succeed. I'm obsessed with the art of learning and even developed a system around efficient learning.

Since then, I have consulted for Fortune 500 tech companies, ran coding workshops, consulted on published tech books, given technical talks, and I have helped those with zero experience in programming get jobs in just a few months. Mainly because I think bootcamps and colleges overcharge you money. Don't worry, you can do it for free as you will see below. I am now in a position where I don't have to work for anybody. I love this career and I think many people would enjoy it and benefit from it as well. So I'm on a mission to help others who want to make this jump no matter what their economic situation.

Ok that last sentence was a wee bit dramatic… 🤔

What language are we going to learn?



Yep, this one up here.

You are going to become a Javascript ninja/ninjess/ninjother for the following reasons:

→ Javascript is everywhere. Every company that has a website or an app needs someone with Javascript knowledge. This language is a requirement for a TON of job postings (If you don't trust me, search for Javascript in your area on LinkedIn).

→ With the introduction of Node.js, you can use Javascript to create a full-stack app (English = you can use javascript to build your entire project). Using tools like Electron, React Native and many others, Javascript allows you to build a desktop app, a mobile app, a web app, and even VR apps. You can control robots by using something like Jonny-Five. You can build your own blockchain. Want to be in Data Science or Data Analysis? Maybe AI and Machine Learning? Great, you can use Javascript to do all of that too.

→ If you didn't read the first point and you were thinking about something else, let me reiterate: Industry demand for Javascript experts is HUGE. It is the most in demand language by employers and most popular. It is not uncommon for developers to get contacted by recruiters and head hunters multiple times a week for job offers.

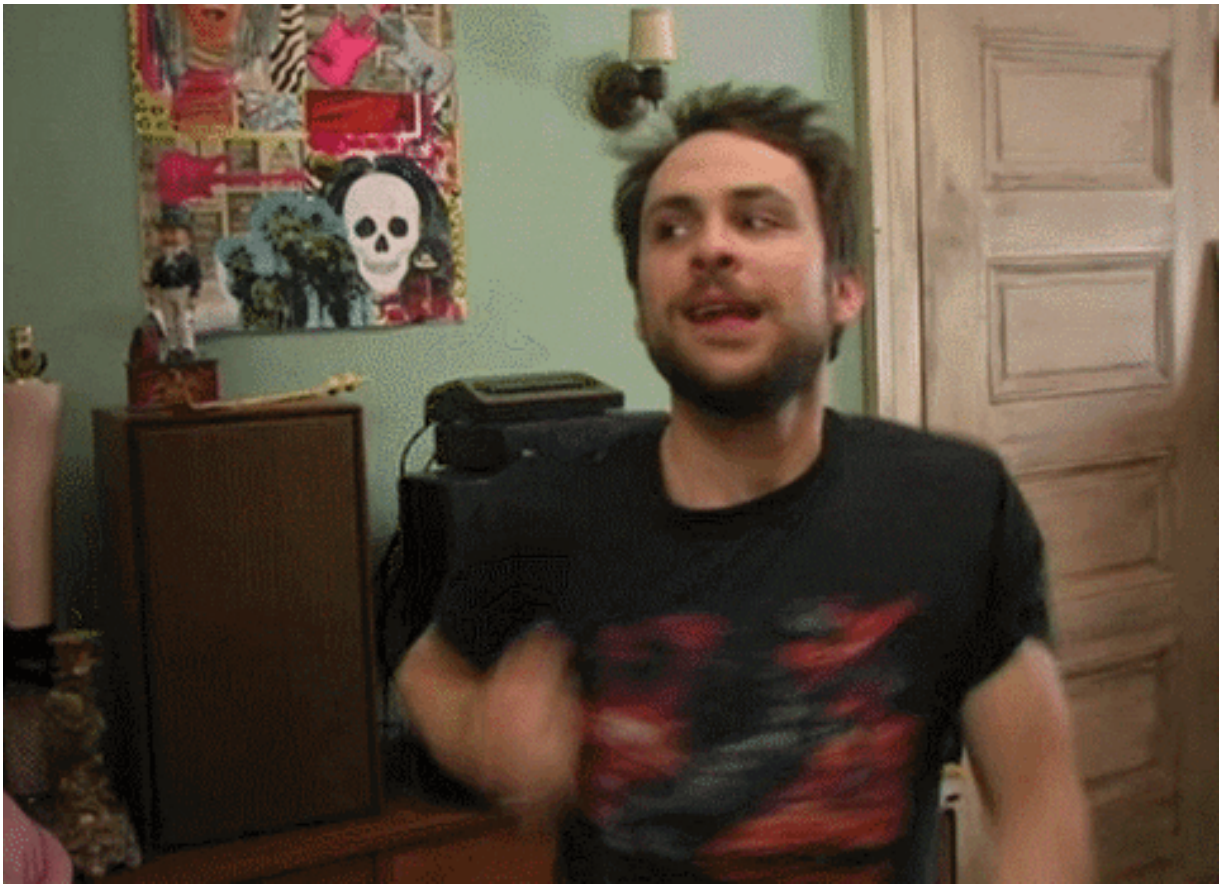
→ Javascript community is growing at a crazy fast pace. There is a lot of new developments in the community. Some people complain about Javascript fatigue, since more and more new tools are being developed every day. However, we can use this to our advantage. You would be learning these new technologies at the same time as people who have been in the industry for years.

→ But what about Python? I heard that is all the rage? Python may be great for things like Data Science and Machine Learning, but you don't get that immediate satisfaction that JavaScript gives you because it isn't really used to build websites (which is the easiest way to get excited about coding and learn). Both languages are in demand, but the path of JavaScript developer is more defined and focused. You can learn Python later in your career.

Trust me, it is a great community with a lot of demand.

Enough jabber, let's get started. Below you will find what I believe are the best resources for you to get the most out of your time. By the end of 5 months, you should be able to land your first real non-entry level programming job. No bootcamps. Just you and your determination.

The 5 months — step by step



Get excited!

We will be focusing on the most employable and in demand skills in 2021. No time for outdated technologies like PHP or jQuery. There is nothing wrong with them, and I have total respect, but based on some of the emails I have received over the years from you, a lot of people are in financial need and have families that they have to support. Time is important to you and you want to be employable as soon as possible and learn the modern skills.

1st Month: The Big Picture

Big question to answer: *How do computers, the internet, and websites work? How can I build a website?*

- [ ] Understand the Feynman Technique and the Trunk Method for learning so you really learn over the course of the next 5 months instead of just using your short term memory.
- [ ] Throughout the months you will be building lots of projects. In order to help you, I have compiled a list of assets like free images, icons and logos you can use to make your projects look nice. Bookmark this
- [ ] How does the internet work: This from LearnCode.academy and this from thenewboston.
- [ ] The best overview of Computer Science: Crash Course Computer Science.
- [ ] Watch this to understand the full web developer industry so that you understand how all the skills fit together. When you first watch this, it will seem very confusing. At the end of the 5 months, watch this again and you will see how everything makes sense now.
- [ ] Follow this Harvard course on YouTube. This is just pure gold from probably the best computer science instructor. No need to do the exercises. They do a new one every year but this one from 2017 is the best version of the lectures. If you have time you can watch the lectures from 2021 as well.
- [ ] How to use the command Line: This by Zed Shaw.
- [ ] How to build a website/get a domain/and have it up and running: LearnCode.academy video.
- [ ] How does HTML, CSS fit together: Watch this shorter playlist or this longer playlist. Or both if you have the time.
- [ ] Learn to build websites with Bootstrap. Start with this then go to the Bootstrap 4 documentation and add components you see there to a

sample website. Understand the benefits that it provides vs writing CSS yourself. BUT don't get stuck on this, because instead we want to focus more on the next two that are more employable skills:

- Learn how to use Flexbox. Watch this playlist and then go and do this 5 minute exercise. Now that you know Flex-box, go and learn CSS Grid for more website layouts. Then do this and this exercises. Build your own website layout from scratch.
- Understand how to use templates to build websites using free themes and templates.

- If you have time, you can do a few of the courses on the HTML and CSS Responsive Web Design sections at freeCodeCamp. It's 300 hours long so I wouldn't say this is the best use of your time but a nice thing to skim through.

THIS IS IMPORTANT: Don't try and memorize all HTML and CSS properties and tags. This is a mistake I made as well. You want to start learning Javascript as soon as possible, which is the main part of being a web developer. No matter how "unready" you feel or incomplete your knowledge about CSS, move to the next part as you will be using HTML and CSS throughout the rest of the months. Trust me on this.

2nd Month: Javascript

Big question to answer: *How does Javascript make machines do what you want?*

- This is where most of your focus will be in the later months as well. What problem does Javascript solve? Start writing little programs in Javascript to make your website behave in a certain way. This language makes your websites do things other than just look pretty. Start with this in depth free course.
- Learn about DOM manipulation. Learn to inject `<script>` tags in your html to run javascript files. And then do this exercise.

- [ ] Read this great article about programming.
- [ ] This is a long series that you won't finish but use it as a reference anytime you encounter something you don't understand in javascript.
- [ ] Learn the new ES6, ES7, ES8, ES9 and ES10 features with this tutorial. If you don't get everything in here, don't worry, we will go over another resource next month on the topic of "Asynchronous". Understand the difference between JavaScript and ECMAScript by reading this and understand how JavaScript gets updated every year.
- [ ] Learn Git and Github with this 40 minute tutorial (yes, that's me). Create a Github profile and start making commits every day. Start developing a sample website. Use Github pages to put your websites online for free. Also use this Git Explorer to practice, and then learn more about Git Branching here.
- [ ] Terminology/Jargon: use this.

3rd Month: Javascript + NPM + Building Your Website

Big question to answer: *Can I build a professional looking website and understand the entire process?*

- [ ] Google Developer Tools → learn how to debug your programs and websites using Google Chrome. Finish this short little course. Dive deeper into this skill because it is very important: do this course (sign up for the free trial).
- [ ] Start attending local meet-ups on coding and Javascript.
- [ ] Learn the difference between synchronous and asynchronous javascript.
- [ ] What is the event loop? → Once you have a good grasp of Javascript this talk will be a game changer. Hands down the best talk on

Javascript ever given. Watch this video every month for the next 3 months. Then watch this free video that I made.

- [ ] Learn about Promises, and Async Await in ES7 here.
- [ ] Learn about the history of modules in Javascript here.
- [ ] Download node.js and npm. Download lodash from npm and use browserify to use CommonJS imports. Learn about it here. Understand why npm is such an amazing tool for developers. Now learn about why we no longer use Browserify, and learn about native imports and exports.
- [ ] Now using the above knowledge, learn a skill often overlooked by beginners: how to read documentation. Learn to use Parcel by reading their documentation and see how it bundles your code.

4th Month: React.js

Big question to answer: *What problem does React solve?*

I'm heavily biased. I love React.js. As a matter of fact, I teach it to others and run workshops on it. So just trust me on this one. React dominates the industry when it comes to job demand. In 2021 this trend is even stronger. There is also Svelte, Angular and Vue.js as an alternative, but you want to stick with React for the best outcome. For example, check out the average salary of a developer that knows React.

- [ ] React
    - [ ] Do these in order: one, two, three.
    - [ ] Then head on over to the official documentation and read through everything.
    - [ ] Then learn about React Hooks.
- [ ] If you have the time and you want even more in depth tutorial on React here it is.

- [ ] [Optional] Learn Redux → Watch this course. Don't let your head explode. Then read the documentation for it as well. Learn why managing state is a big problem that all large applications need to solve.
- [ ] Build a sample React application using create-react-app. Create-react-app will blow you away. It will open up a new world for you. Command Line Interfaces (CLIs for short) are now becoming common practice with all front end frameworks and lets us set up a project quickly.
- [ ] Deploy your React app on GitHub pages. In the future you should deploy all your projects on Github pages to show off in your portfolio.
- [ ] Deploy your React app on Heroku.
- [ ] Deploy your React app on Netlify.
- [ ] Read all the articles here in the must-read category.
- [ ] Sign up to these email lists to keep in touch with what is happening in the industry: Javascript and React and Web Developer Monthly.
- [ ] Start building your online resume. There are people that give better advice than me on this. Check this, this and this out. Or you can go all out and check out this... but this post is already getting too long and you're starting to give me evil eyes 👀.

Last Month: Servers, Databases and Connecting the Dots

Big question to answer: *Where do servers, databases, and raspberryPis fit into all of this?*

- [ ] HTTP, JSON and AJAX. Learn how these allow you to communicate with servers.
- [ ] Go a step further and master Node.js and Express.js here. Learn how to build an API server.

- [ ] Once you are done with this, use a fun API like this one and build a simple app.
- [ ] Subscribe to the computerphile YouTube channel and watch their videos as they come. Even though topics may be difficult, it will introduce you to some amazing things.
- [ ] [Optional] What is a Computer/Server/OS: buy a raspberryPi. Look up different projects on youtube you can do with your raspberryPi. Finally, build a simple script that makes lights attached to your raspberryPi blink. Follow this course. Host your website on the raspberryPi. Be amazed at how cool you are. I know I said this is supposed to be all free, but when I was starting out, this was one of the best "Aha" moments I had.
- [ ] Build a small project using a database you create here. Go a step further and create an app using firebase as the database and use firebase to set up user login/logout.
- [ ] Learn basic Web Architecture concepts by reading this.
- [ ] If you have the time, spend a day building this chat application using react hooks and sockets. Add this to your portfolio.
- [ ] Start practicing for interviews by trying to answer all of these questions. If you get something wrong, learn why you made that mistake and learn from it.
- [ ] Spend one day each on the below subjects. You don't need to have a good grasp on them. Just learn why they are there and what problems they are solving: Testing (TDD), Machine Learning Basics, Time Complexity (Big O), SQL, TypeScript, UX/UI, Continuous Delivery, Basic Data Structures (You should be able to explain what a data structure is. Hint: Arrays and Objects are two popular Javascript data structures).

I can already hear people screaming at me with the above suggestion. "Are you out of your mind?! You don't think < Enter topic in the last part here > is important? Only 1 day to learn each of those?" But hear me out. I do agree that these are important topics to cover in order to be a good developer, and

everybody should learn the skills. However, we are trying to build a trunk of foundation here. It is easy to start diving deep into a topic, but without the foundation you won't actually know why it's important, or how it relates to what you are doing. Additionally, in most job postings I found, there was very little mention of the above skills. Just save learning these until you are on the job.

REMEMBER: Your goal is to get employed in the most efficient manner.

Let's Recap

By the end of the 5 months you should have the below requirements completed:

1. Learn HTML and CSS. Then, buy a domain, buy hosting from a place like BlueHost or HostGator, get the cheapest option, make a website, and put it online. You can skip this option if you would like to use Github Pages which is free or using something even more bare bones like Digital Ocean. My personal favourite is Netlify. But if you can afford it, actually buy one of the above hosting platforms so you understand how they work. This is going to be your portfolio

from now on. Learn how to update it and make edits. As you learn new things, continue to make it nicer and nicer. Don't spend too much time on this. Just enough to show that you're able to put something online and make it look nice. Focus on having 1~2 really good and big projects in your portfolio instead of 30 small ones that anyone can build in a day (since employers won't find this impressive).

2. Start learning Javascript. Now how can you make your website interactive? Go through the above resources and see what Javascript does.

3. Start pushing your little projects to GitHub. Employers will look at your GitHub profile and how active you are on there. Try to make commits 5 times a week on your personal projects. Also, try reading through this and contributing to some open source projects like freeCodeCamp or Zero To Mastery Open Source (we set up the projects here so that you can participate no matter what your level, or when you join. You can read the get started guide here).

4. Learn to google and use StackOverflow when you have problems. 99% of problems you will encounter when you start out can be found online. Or join a local Discord of Slack server for developers and ask questions when you are stuck. If not, pick one from here and talk to other developers. The key is to figure out how to solve your own problems and not always follow a tutorial and watch somebody else answer your questions.

5. Become comfortable using a command line to do things. Always have it open when practicing and try using it instead of the GUI (graphical user interface).

6. Learn the newest language features and trends in Javascript, and learn to solve problems with them (i.e. Promises, ES6, ES7, ES8, ES9, ES10, ES2020 functional programming techniques). Also keep an eye out on the state of javascript survey every year to see what is trending in the industry.

7. Attend local meet-ups and start talking to people. You will be really overwhelmed and confused by all of the things you don't know. Don't worry as this is natural. Just start meeting other coders so you can be surrounded by the lingo and jargon.

8. Start listening to the podcast: Javascript Jabber or SyntaxFM. This will get yourself familiar with the jargon so when interview time comes, it doesn't overwhelm you. The first few times you listen, you will have no idea what they are talking about. Don't lose hope. Eventually it will all make sense. For a more advanced podcast, but probably the best on software, check out Software Engineering Daily. This is a podcast you will appreciate a lot more later on in your career. I'm not going to mention Youtube here because we all know it. Search YouTube anytime you want to learn quickly about a certain topic. A lot more options out there now compared to when I first started years ago.

9. Start applying to recruitment agencies early. We are going to use them as practice. Most of these have practice interviews with professional coders so they can rank your skill, but you can use these to practice programming questions, and ask these experts any questions you want!

10. Start applying for jobs for which you are way under-qualified for. You will get some interviews. You should never settle for a job. If you never ask, the answer is always no. See part 2 for more detail on this.

11. Make your LinkedIn profile look nice. Join our group to help endorse your skills. Don't spend too much time on your resume. Make it one page, make it concise, and write down all the skills you've learned in the previous months. Use a prebuilt template like this. Being self taught shows a lot of courage. Remember that your resume is just to get you an interview, after which, they are as good as paper towels… ok bad analogy because paper towels are very useful. I spent less than 2 hours on my resume. What makes you different than other developers is the fact that you come from a different field and background. How is this going to differentiate you?

12. Interview and be amazed at how employable you are. Not all of them will go well, but then again, not many developers learned everything in the last 5 months. It shows ambition. ONLY apply to jobs on LinkedIn, and the rest should just be you emailing directly, referrals, or calling the company you want to work for. Don't waste your time on sites like Craigslist, Kijiji, or Monster.com, or other

job board ads. Finally, you can check out this handbook for some technical interview advice.

13. * Watch this again to understand the full web developer industry so that you understand how all the skills fit together. Now that you have learned a lot, this should make sense and give you confidence in your roadmap moving forward.

What is the 20% that will get me 80% of results?

Most people have an idea that you need to get something 100% before they can move on to the next step. However, for most skills, including programming, the closer you get to 100%, the longer it takes to get there. You only have 5 months. The last 20% will be better served actually working in teams, on real projects (and getting paid). So we are only focusing on getting 80% of the knowledge to use our time efficiently.

Biggest takeaway from all of this

Technology is always changing. This is especially true with web development. Things are moving so fast right now that it is impossible to know every single library, syntax, or framework. What you do need to know is how everything fits together and what each technology is trying to solve. Most importantly, you just need to know it exists so you can look into it and figure it out when the time comes on the job. Programmers are problem solvers. Learn to solve problems with the tools available to you. Most of us spend a lot of time on pages like StackOverflow or researching Google because there are so many resources out there. Once you build the foundation of your knowledge, you can go anywhere. You just need to know how to look for answers and ask questions.

Conclusion

Focus on efficiency. The reason most of us give up on a goal is because we don't see results. By focusing on the things that matter, it makes learning fun. But it doesn't end here. Learning never stops, and your goal was to get employed as soon as possible so that from that point on, everyday you are receiving a salary to learn.

Coding gets more and more fun with each passing day and it's even better when you are getting paid every day to solve problems and develop your skills. The real growth happens when you start working on real projects with real teams. That's why I strongly believe that you want your 'study' period to be as short as possible, in order to avoid debt, and increase your time in the best environment for learning: working in teams. I wouldn't even recommend freelancing to start off. You want to surround yourself in an environment where everybody is smarter than you and you are working everyday with them. From there, be a sponge and absorb all of the information.

We're building that trunk. When that trunk gets big and strong, and the roots are all put into place, your rate of learning new things will be exponential. You'll form leaves of knowledge faster and faster with each passing day.

> ***Make 2021 the year that you took a risk, you learned a highly in demand skill, you were terrified, you had new experiences, and you received new opportunities. Good luck!***

One last thing...

I created an online course: **The Complete Web Developer in 2021** where I walk you through the entire steps I mentioned above if you want everything in one place, extra help with your questions, or you want to support my work.

It's over **200 HD videos and 35+ hours of content**. It took an insane number of hours to make. But I'm really proud of how everything turned out. I strongly believe it is better than any bootcamp material out there or any other course online!

Enjoy 1 hour of free lessons (no signup necessary) by clicking the PREVIEW button beside the lessons here: The Complete Web Developer in 2021 | Free Lessons

We also have a **private community of thousands of developers** going through the course and helping each other out every day.

Due to popular demand, I also wrote a series that will help you go from Junior to Senior Developer:

Part 1: Don't Be A Junior Developer

Part 2: Don't Be A Junior Developer: The Roadmap