

How to Become a Full Stack Developer (Step-by-Step Guide)

bootcamp.cvn.columbia.edu/blog/how-to-become-a-full-stack-developer

November 11, 2020

No one could fault you for feeling a bit overwhelmed.

After all, a quick Google search for “full stack development” renders an impossibly long list of acronyms: HTML, CSS, JS, MySQL, and PHP, just to name a few. At heart, full stack developers are highly versatile jacks-of-all-trades in an industry that demands comprehensive programming knowledge.

Take a breath: learning how to become a full stack web developer isn't as difficult or time-consuming as you initially might think. In this article, we'll walk you through the basics of full stack development, what you need to learn, and how you can prepare yourself for a full-blown career in development.

Wondering what's next? Here's a sneak peek at our six steps to becoming a full stack developer:

1. Identify The Skills You Need to Learn
2. Determine Your Ideal Timeline
3. Start Learning in a Format That Suits You
4. Begin Building Your Portfolio
5. Start the Job Search
6. Ace the Interview

How long does it take to become a full stack developer? At Columbia Engineering Coding Boot Camp, you can learn the most in-demand web development skills in 12 or 24 weeks.

What is a Full Stack Developer?

To become a full stack developer, you'll need to familiarize yourself with two basic web development concepts: the front end and the back end.

The front end, also known as client-side or customer-facing side, refers to all the elements of a computer application that users interact with directly. By contrast, the back end encompasses all the behind-the-scenes tech that computes business logic, fulfills user requests, and permanently stores sensitive data.

You can think of the front end as the tip of an iceberg, while the bulk of the unseen back end extends far below the water's surface.

Front end developers create and optimize visible parts of a website, making components responsive to different viewing environments, such as smartphones, tablets, and desktops. They use client-facing programming languages, like HTML, CSS, and JavaScript, to render an engaging site.

On the other hand, back end development requires programmers to build and refine the internal software that operates a company's databases, servers, and proprietary software. Back end developers are responsible for creating the pathways to deliver information to and from users who operate front end interfaces.

This is where full stack development comes in. “Full stack” refers to the entirety of a site or application's front and back end architecture. As you might expect, full stack professionals are capable of addressing both sides of the proverbial equation.

Rise, Fall, and Resurgence

At the onset of the Internet age, most web developers were dubbed full stack developers. In the 1970s and 80s, it was common for a single person to craft a software program from start to finish because there weren't many layers of complexity between the computer's hardware and the programmer.

However, as applications became increasingly elaborate, full stack development began to waver in its ubiquity. The advent of client-server computing in the nineties, coupled with the rising popularity of the net, necessitated the emergence of trained specialists. Programmers were subdivided based on the company's stack: Networks, the front end, databases, and servers were all managed by distinct departments.

The diversified trend only started to reverse in the late 2000s. New technologies became streamlined, allowing aspiring developers to construct a fully fledged application in a shorter period. Gone were the days of the Java stack and Oracle databases — instead, high-powered databases like MySQL, PHP, and Ruby on Rails claimed the spotlight.

In recent years, the rarefied full stack developer has become significantly more critical. Employers understand their centrality; after all, every team benefits from at least one individual with a holistic knowledge of a project's parts. Full stack developers have a special knack for drawing together various tech tiers into a neatly packaged application.

In short, full stack developers are responsible for the following activities:

- Working with a website from start to final launch
- Prototyping, creating, and building websites along with user interface teams
- Designing user interactions through robust front end architecture
- Managing database and server administration duties
- Communicating with end-users to determine product effectiveness

According to Stack Overflow's 2020 Developer Study, at least fifty percent of developers identify as full stack professionals, while 55.2 percent identify as back end, and 37.1 percent as front end.

Of course, becoming a full stack developer isn't as simple as titling yourself one. There are a few mission-critical skills you'll need to master before sending out those job applications.

So, what does it take to become a full stack developer? Below, we'll jump into the core competencies you'll need to succeed.

Identify the Skills You Need to Learn

Every learner's path will be slightly different depending on their preexisting experience and skill set. If you're a true beginner, you'll require specialized training and project experience to become a developer. The same applies to hobbyists, who will need to upskill much more than professional developers who hope to upgrade their careers.

Take stock of your skills! Here are a few basic capabilities that you'll need to master if you want to thrive as a full stack developer:

While a full stack developer's education is never really complete, the above skills will provide you with a sturdy foundation!

Learn the skills you need to become a full stack developer at [Columbia Engineering Coding Boot Camp](#).

Get Boot Camp Info

0%

Determine Your Ideal Timeline

How soon do you want to go from hitting the books to landing your first entry-level developer position? As with your earlier skill assessment, establishing a concrete timeline can help you choose your educational journey.

Consider the following questions:

- *Are you willing to take part-time courses while maintaining your current job?*
- *Would you prefer an intensive, full-time education that would allow you to land a job as quickly as possible?*
- *Do you prefer quicker, skills-based courses or formal, theory-heavy education?*
- *What learning opportunity will allow you to pursue your education without infringing on your personal obligations?*

If you've already locked yourself into a full-time career or family commitments, it can be challenging to start a new training regimen. But it is by no means impossible. Think about the changes you're willing to make to pursue your dreams — and build your educational schedule accordingly.

Start Learning in a Format That Suits You

In the era of open-source everything, it's easier than ever to learn how to become a full stack developer — but it's also easier to get lost in the muck. There's a near-daunting number of distinct educational routes available to newcomers.

Below, we touch on the primary methods that aspiring developers use to become marketable programmers, and provide resources to help you on your journey.

There isn't one best way to learn full stack development, so while you're perusing the options, make sure to compare your timeline and learning requirements to the listed opportunities.

Boot Camps

Coding boot camps are short-term, highly intensive training courses that equip you with all the industry-ready skills you need to become a fully fledged developer. By the time you complete a boot camp, you'll have gained the necessary skills, made vital industry connections, developed an impressive coding portfolio, and prepared yourself for the job hunt.

Unlike conventional four-year degree tracks, boot camps provide a quick way for hobbyists and total newcomers to learn the necessary skills. Most of these intensive courses complete within three to six months and are cheaper than a university education. Boot camps also offer more flexibility than pursuing a formal degree; many offer part-time, full-time, virtual, and in-person training options.

If you're curious, make sure to check out [Columbia Engineering Coding Boot Camp](#), which dives into all the most in-demand front and back end technologies, including HTML, CSS, JavaScript, MySQL, MongoDB, React, and Java.

Self-Guided Courses

Are highly structured training courses not your cup of tea? Thankfully, there's an enormous selection of instructor-guided, self-led virtual courses readily available for new learners.

Compared to boot camps, self-guided courses offer much greater flexibility and a lower cost — some are even free! Depending on the platform you opt for, you may even have access to additional resources, instructor support, and an active learner community.

That said, self-guided courses differ significantly in terms of quality. While some may provide extra resources, many don't. The amount of support you receive is, in most cases, tied to the money you shell out — although that logic shouldn't preclude you from thoroughly vetting platforms before you enroll! Check out participant reviews and only take courses that have been published through reputable platforms.

Additional Resources

Below, we've listed a few of the most-loved coding course platforms. Each of the linked sites provides easy-to-follow and relatively inexpensive front end, back end, and full stack development tutorials.

- **Coursera:** Coursera offers an enormous menu of full stack web development courses ranked by learner star ratings. Many courses focus on specialties such as multi-platform app development, responsive website design, Android/iOS development, and object-oriented programming.
- **Udemy:** Nearly 300,000 registered learners are currently learning full stack development on Udemy. Considered one of the most trusted self-led platforms, Udemy offers both technology-specific courses and theoretical materials for those that want a deeper dive.
- **FreeCodeCamp:** Want to avoid paying tuition for the fundamentals? FreeCodeCamp provides an easy way to build projects, earn certifications, and learn to code from home. The site covers topics ranging from data visualization to web design, data structures, and coding interview prep.

Degree Programs

Four-year degree programs remain one of the most common educational paths for aspiring developers. According to Stack Overflow's 2020 Developers Study, roughly three-quarters of professional developers worldwide hold *at least* a bachelor's degree, while 49.3 percent only have a bachelor's degree.

The benefits are evident: You receive a comprehensive, broad education that demonstrates competency to employers. College allows you to build a knock-out portfolio, land internships, and make valuable connections.

However, it requires time and financial commitment to enroll in and complete a degree program. College may be best for those that have yet to pursue their undergraduate education or those who want to gain a comprehensive understanding of both development theory and practice. Learners should opt for majors that prepare them for full stack development, such as computer science or web development.

Stack Overflow's 2020 Developers Study also found that the most popular majors for developers were:

- Computer science, computer engineering, or software engineering (62.6%)
- Other engineering discipline, such as civil, electrical, or mechanical (9.3%)
- Information systems, information technology, or system administration (7.9%)
- Natural sciences, such as biology, chemistry, or physics (4.4%)
- Web development or web design (3.3%)

Books and Tutorials

If you're not a fan of formal courses, you're not out of luck. It's possible to learn full stack skills independently by referencing books, online tutorials, and other media.

Stack Overflow's 2020 survey found that 85 percent of 46,383 developers felt that formal education is at least "somewhat important." However, that doesn't mean it's impossible to succeed through self-directed learning; nearly 16 percent responded that formal education is "not at all important or necessary."

Keep in mind that self-learning requires a special blend of discipline and passion. Even with all the book knowledge in the world, many employers are reluctant to hire someone who hasn't received any formal certification.

However, book learning may function better as an entryway into more formal instruction down the road. By establishing coding as a hobby, you're much more likely to learn enthusiastically in a structured course.

Moreover, all experienced developers are, in one way or another, self-directed learners. Technologies change so rapidly that developers need to learn quickly or risk being left behind.

Interested? Check out a few of our favorite books covering full stack development.

- **The Full Stack Developer** by *Chris Northwood*. In this pivotal book, Northwood explores the essential skills every full stack developer needs. The book is primarily geared towards developers new to AGILE development, project management, QA, and back end patterns.
- **Eloquent JavaScript** by *Marjin Haverbeke*. Haverbeke balances the nitty-gritty technical aspects of the language JavaScript with a principled approach. The book is tutorial-style, allowing developers to follow along with their own projects.
- **ASP.NET Core 3 and Angular 9** by *Valerio De Sanctis*. De Sanctis explores the latest developments in designing, building, and deploying a Single Page Application with ASP.NET and Angular. Full stack at heart, the book touches on data management, web APIs, app design, and deployment.
- **Full-Stack React Projects** by *Shama Hoque*. Curious about the MERN stack? Hoque uncovers how to create dynamic web applications and leverage the power of React.js to build complex interfaces.

Once you're done hitting the books, it's time to start building an "employer-worthy" portfolio.

Begin Building Your Portfolio

Contrary to what a frustrated job-searcher might think, employers don't organize interviews to turn people down. A sane hiring manager wants to believe that you're the perfect person for the position. It's your job to provide them with as much evidence of your capabilities as possible.

That's why portfolios are so crucial: they're the first impression recruiters have of you. If you have any projects you're particularly proud of, use your portfolio to showcase them.

Display your work by describing the technologies used, your role in the project, and showing off screenshots. As a full stack design professional, your work should demonstrate an eye for design and excellent technical know-how.

Make sure to publish your code to a public repository on the code versioning site GitHub. Don't be afraid to include projects that are currently in development, either. Building a portfolio is perhaps the best way to learn full stack development.

Start the Job Search

Once you've built a portfolio with two or more polished projects, you're ready to start job hunting. Although it's a little daunting, the right resume and skill set can capture a hiring manager's attention.

Make sure to define your career goals ahead of time, get cover letter feedback, and tailor your resume for the specific positions you're applying for. Don't limit yourself to online searches; reach out to companies directly, attend career fairs, and sign up for weekly job alerts.

Most importantly, don't get discouraged if your inbox fills with rejections (or nothing at all). According to Indeed, it takes the average job-seeker roughly nine weeks to apply, interview, and land a new role.

Do Well on the Interview

Got the interview? Take a moment to pat yourself on the back — and then explore a few of the most important tips for getting that second interview and, hopefully, landing the role.

Refresh Your Coding Skills

Virtually all full stack coding interviews will include a technical portion, which allows the hiring manager to gauge your current skill level. Continually practice on online platforms like HackerRank, LeetCode, or reference a copy of Gayle Laakmann McDowell's *Cracking the Coding Interview*.

Keep in mind that there are many different varieties of technical challenges, such as live coding exercises, take-home assessments, design challenges, and soft analysis of programmatic thinking.

Prepare for each type by completing code challenges within a 30- to 40-minute timeframe. Look at examples of broken code to spot bugs, fix them, and then explain how you did it.

Prep for Behavioral Questions

Hiring managers don't just want to hire a programming wizard; they want someone who will mesh with the company culture and work well with others.

Prepare answers beforehand that show how you deal with stress, work disagreements, or challenges in daily life. Read up on the most common behavioral questions so that you're not caught off guard.

Prepare for Verbal Explanations

In the tech world, live-coding and whiteboarding are nearly universal. During a whiteboard exercise, you should explain your thinking process and how you arrive at a solution.

When approaching a problem, always ask the hiring manager for more specification: What type of input does the function consume? Should the function return a specific data type?

Be prepared to analyze the efficacy of solutions based on their time and space complexity. How efficient is your code? Can your application be refactored to take up less memory or run faster.

Even if you don't get it right on the first try, managers are more concerned that you're able to think your way through a complex problem. Making code work isn't everything; you've got to understand why it does what it does. Employers want candidates to understand all the necessary parts of the tech stack. You should make it known that you're prepared to be a lifelong learner and feel comfortable exploring new technologies.

Additional Resources:

- **Cracking the Full Stack Developer Interview** (*Hirako San*) Representing a broad curation of the most common interview questions, San teaches you everything you need to know about getting the best developer jobs.
- **10 Essential Full Stack Interview Questions** (*Toptal*) This notecard-formatted article gets right to the point, providing practice for the most important full stack interview questions.
- **The Best Resources to Ace Your Full Stack JavaScript Interview** (*freecodecamp*) This article explores the most vital JavaScript, Node.js, React, and Redux questions to master before stepping foot in the interview room.

Final Thoughts

Full stack development allows you to be on the emerging frontier of technology, exploring novel solutions with cutting-edge companies. Moreover, because the industry is rapidly growing, it offers tremendous career potential and security to tech-savvy coders who want to make their mark in the programming field.

If you're thinking about becoming a web developer, why haven't you started pursuing your dream? Start considering your educational opportunities now; there's no time like the present to start furthering your goals and landing a fulfilling position. Make sure to check out Columbia Engineering Coding Boot Camp for a great place to embark on your new journey.