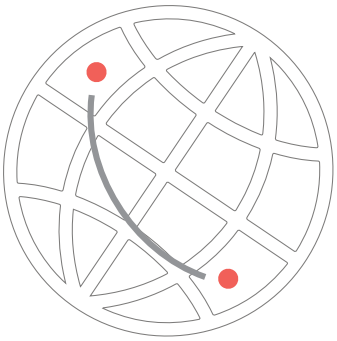
The Ultimate Guide to

CODING FOR BEGINNERS

And How You Can Use Your New Skills to Get a Job!

skillcrush

YOU HAVE THE OPPORTUNITY—SAY YES



You are in possession of the most powerful machine humans have ever made. Your computer can connect you to anyone, anywhere on the planet. In fact, it can connect you to *everyone*, everywhere on the planet, in a matter of milliseconds.

And that's barely the beginning of what it can do.

But, if you're like most people, chances are you're taking advantage of less than one percent of what this machine can do.

Look, there's nothing wrong with using your computer to check email and see what your friends are up to on Facebook—I do those things all the time!—but if that's all you're doing, then you have your hands on the most profound and life-changing opportunity that humanity has ever been given, and you're blowing it.

. . . . . .

The barrier to owning a computer is lower than it's ever been, so with a few hundred dollars plus Wi-Fi, you have everything you need to begin your path toward world domination.

So why do so many people own computers but so few take advantage of the opportunities that computers afford them? It's simple: People don't know just how much computers can do, or think they won't know how to harness computers' power.

After having answered thousands of emails from students worried they aren't young enough, smart enough, or math-y enough to learn to code, I have come to the conclusion that the problem is 100 percent BS: We overestimate the complexity of computers and underestimate our own intellectual abilities!

When I think back on the funny mix of circumstances that led me to where I am today—CEO of an online tech education company—the answer isn't family money or a background in business or even a computer science degree.

Instead, because of a funny set of coincidences and one well-timed layoff in 2009, I ended up in a situation where I had only an inkling of how powerful this machine could be—but the time and motivation to learn how I could use it to my advantage. When I started to learn to code, I didn't know where it would lead me or how I would get there, but I was lucky that I believed I could figure it out.

If nothing else, I want you to walk away from reading this guide knowing just one thing:

You *can* learn to harness the power of these machines.

Yes, *you*!

No matter how old you are, or whether you're good at math, or whether you've ever worked at a technical job before. None of that matters.

If you have an interest and are inspired to learn, you can do it. And the rest of this guide will show you how.

Adda Birnir
SKILLCRUSH
FOUNDER & CEO

BEFORE YOU BEGIN

TALK THE TALK

One of the first things you need to learn when you're thinking about starting a career in tech is the language. You've probably heard the basics before: terms like HTML, CSS, WordPress, etc. But do you know what those things actually are?

You don't have to memorize every single tech term out there, but getting familiar with the major ideas is critical. Knowing the tech terms you're most likely to encounter makes it easier to ask the right questions from the right people.

Here are the most common terms you're likely to hear. For even more tech terms, check out Kelli Smith's 99 Terms You Need to Know When You're New to Tech on the Skillcrush Blog.

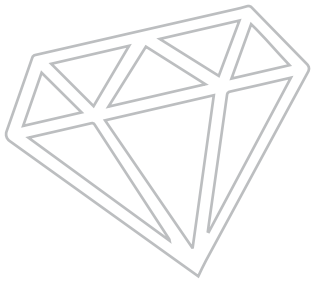
AGILE, OR AGILE SOFTWARE DEVELOPMENT
A set of principles for coding software that prioritizes continuous improvement by launching as soon as possible and releasing frequent updates to a piece of software instead of waiting until it's perfect.

BACK END
Part of a website or web service that makes it work and includes applications, web servers, and databases.

BUG
Mistake or unwanted piece of code that keeps a website or program from working like it should. More specifically, you call something a bug when it's not working as expected.

CLOUD COMPUTING
Storing and accessing information and services via the internet instead of on your computer.

CODE
A simplified form of language with very strict rules and syntax used by humans to tell computers what to do.



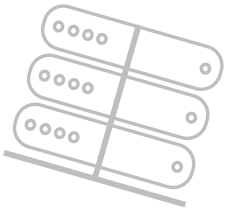
CODING LANGUAGE
A *specific* set of rules and syntax for writing the code that tells computers what to do. This includes programming, assembly, and markup languages such as Ruby, PHP, and HTML.

COLOR THEORY
Characteristics of colors and the relationships between them.

COMPUTER PROGRAMMING
The process of writing and implementing various instructions for a computer to do a particular task (or set of tasks), using code.

CSS (CASCADING STYLE SHEETS)
Code that tells browsers how to format and style HTML for a web page and controls things such as font type and colors.

CSS3
The most current version of CSS.



DATABASE
Collection of electronic information (data) stored on a web server.

FRONT END
The part of a website that can be seen by users and is made up of HTML, CSS, and JavaScript code.

GRID SYSTEM
Set of columns and rows that can be used as guidelines to arrange content on a web page.

HTML (HYPERTEXT MARKUP LANGUAGE)
A coding language used to put content on a web page and give it structure. Since HTML doesn't tell computers to do anything, it's not considered a programming language (this is a distinction that only matters in job interviews when an interviewer asks if you can "program").

HTML ELEMENT
HTML code made up of an opening tag, a closing tag, and information between them.
Example: This is my paragraph element!

HTML5
The most current version of HTML.

HTML5 APP
A web application designed specifically for use on mobile phones using the latest HTML5 and JavaScript technologies.

LEAN OR LEAN STARTUP

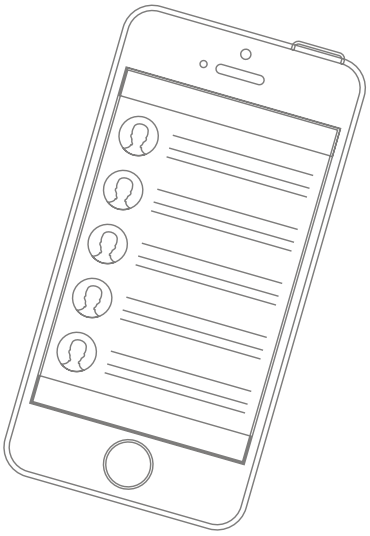
A popular process for launching products and quickly iterating on them to better meet customer needs, based on continuous customer feedback. Think of it like agile but for companies. This term was popularized by the book *The Lean Startup*.

MOOD BOARD

An inspirational collection of content showing the visual style for a website including color palette, images, icons, fonts, etc.

MINIMUM VIABLE PRODUCT (MVP)

A product with the minimally adequate features to meet the needs of early adopters, often used to test a concept or idea without a huge outlay of resources. Popular among lean startups.



NATIVE APP

A mobile app built using the software development kit (SDK) native to a specific mobile device.
Example: any app coded for the iOS (Apple) operating system

OBJECT-ORIENTED PROGRAMMING (OOP)

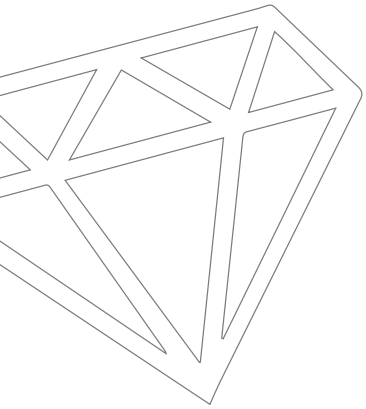
A popular way to design software programs (commonly known as a design pattern) where code is organized into objects that have specific and unique attributes and abilities.
Example: A blog might include a blog post object that has a title, date, and content attribute
Examples of OOP language: Ruby, PHP, Python

PROGRAMMING LANGUAGE
Technically a subset of coding languages that specifically tell computers what to do vs. how to display something. For example, HTML and CSS are *not* considered programming languages but instead are markup languages.

RESPONSIVE DESIGN & DEVELOPMENT
A way to design and code websites such that they can adapt to different-sized devices like phones, tablets, wearable devices, etc.

SDK (SOFTWARE DEVELOPMENT KIT)
Set of tools for creating a specific kind of software.

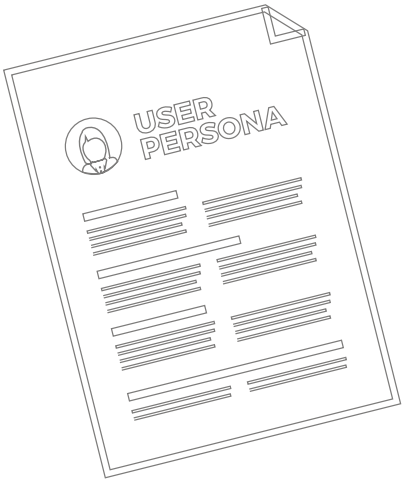
SEMANTIC ELEMENT
HTML element that gives the browser more information about the content in it.
Examples: aside (for sidebars), header, footer.



SITEMAP
An outline or map of the pages needed for a website. Usually drawn using lines and boxes to visualize the hierarchy of pages.

SOFTWARE DEVELOPMENT
The process of programming, documenting, testing, and bug fixing involved in creating and maintaining all manner of software applications and frameworks.

TEXT EDITOR
Software used to write plain text (text with no formatting) that's used for coding and programming.
Examples: SublimeText, TextEdit, TextWrangler, Notepad++



UI (USER INTERFACE)
How a website is laid out and how users interact with it.

USER FLOW
Map of the path users take from getting to a website to taking an action on the site.

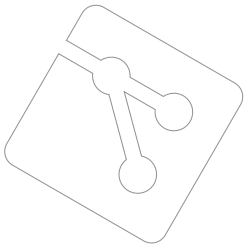
USER PERSONA
Profile of an imaginary person who would use a website; used to define who a site is for and what their needs are.

USER RESEARCH
Investigating how users act and what they need and want in order to better design a website for them.

UX (USER EXPERIENCE)
What a user experiences when they browse a website. This can range from straightforward usability (can they accomplish a given task?) to the less tangible (what do they feel when they're on the website?).

VERSION CONTROL
Software used to keep track of changes to code files, similar to the Track Changes feature of Word. Used by software teams so that they can work on the same code files at the same time without overwriting one another's work.
Example: Git, Subversion

VIRTUAL REALITY OR VR
A computer-generated simulation of a three-dimensional environment that users can interact with in a somewhat realistic way, often using equipment like a helmet with a screen or interactive gloves.



WEB APP OR WEB APPLICATION

A website with complex functionality and heavy interactivity.
Example: Twitter, Facebook, Bank of America

WEB APPLICATION FRAMEWORK

A series of pre-written code that is used by developers as a starting point to building their web applications.
Examples: Ruby on Rails, Bootstrap, AngularJS

WEB DESIGNER

A designer who specializes in designing websites and web applications.

WEB DEVELOPER

A software developer who specializes in coding websites and web applications.

WIREFRAME

A simple sketch of the key information that goes on each web page, usually done in black and white with boxes, line, and placeholder text.

LEARN MORE

For an even more in-depth look at the basics of tech, sign up for Skillcrush's free 10-day Coding Bootcamp. You'll learn all about common tech terms, plus get a taste of what taking a coding class is really like.

SIGN UP

NOW WHAT?

You've familiarized yourself with key tech terms that you'll hear in the industry. And you've taken the Skillcrush Bootcamp to get a more comprehensive view of the basics of tech. Now it's time to make moves towards a gaining the tech skills you'll need to make a career change.

In order to avoid getting overwhelmed, we've broken down learning about tech into phases. Not only will this make the learning process more manageable, but you will find that after each phase, you'll have more career options.

PHASE ONE: Build Your Foundation

PHASE TWO: Hone in on Your Interests and Specialize

PHASE THREE: Get Started on Your Own Business

We'll walk you through each phase, step by step.

phase one

BUILD YOUR FOUNDATION

step 1

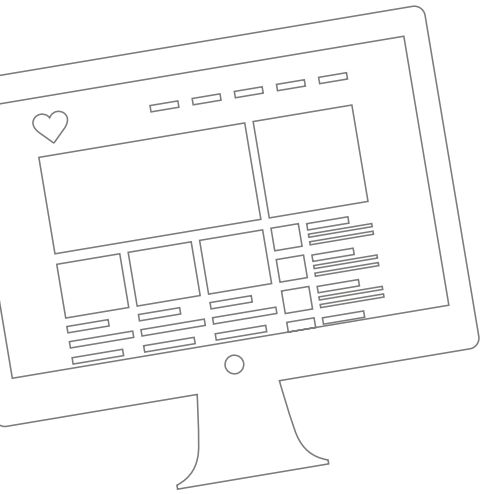
LEARN ABOUT WEB DESIGN

You might have a rough idea of what constitutes web design, but there's more to it than just creating something that looks nice.

Websites are built with a purpose: It could be to provide information, to sell something, to entertain the visitor, or something else entirely. And most importantly, they're interactive.

The way a website looks has to serve the function of the site. When the form and the function work together, the site is considered to have good UX, or User Experience.

The fundamentals of design are the same regardless of the medium. Considerations like balance, harmony, and color theory apply whether you're designing a building, a T-shirt, or a website. But as mentioned above, websites have to function in ways that other types of design don't.

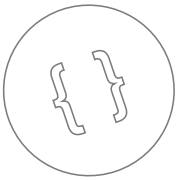


Learning about the basics of web design is an essential first step in any tech career. Even if you later decide you want to work on the coding end of things, understanding how web design works and what makes a good design is invaluable in any area of tech.

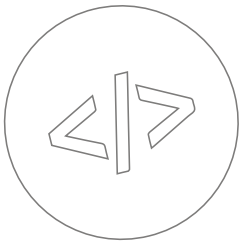
step 2

HTML & CSS, AND JAVASCRIPT

HTML and CSS are the building blocks of the web. You can build entire websites with just HTML and CSS. In fact, with the newest versions of both, you can also build games, animations, and more.



Think of HTML like the framing for the walls and roof of a house. They form the structure of the house and the basics of how it's laid out. You can tell it's a house by looking at it, but it's not necessarily very functional or beautiful like that. It's just a shell.



CSS adds things like the finishes on the walls and floors, the windows and doors, and all the other things that make the house comfortable and livable. You can even use CSS to move the parts around and configure them differently (just like swapping the furniture in a house can transform the purpose of different rooms and alter the layout).

By separating the content (HTML) from the presentation (CSS), you can change the way your page looks without having to rebuild everything from scratch, and you can easily add new content without having to design the whole page every time you want to add something.

JavaScript doesn't fit neatly into the house metaphor—it's what makes slideshows on sites, autocomplete—or any change on a page that happens without you clicking.

LEARN MORE.

Skillcrush's free 10-day Coding Bootcamp is the perfect way to get the basics of HTML and CSS down. And HTML and CSS are two of the first skills you'll learn more in-depth when you enroll at Skillcrush.

UNDERSTANDING HTML, CSS & JAVASCRIPT

Think of HTML, CSS & JavaScript as parts of a building.

1

HTML creates the structure of the building. It's the foundation, the walls, and the roof. With just these parts, you can recognize it as a building (even if it's not a particularly inviting one!).

Here's what some of the HTML that makes up the Skillcrush website looks like (the HTML is the tags that have a letter and sometimes a number inside two bracket like this <h2>):

```
<h2>After completing our Break Into Tech Blueprint you
will be able to:</h2>
<ul>
    <li>Make more money</li>
    <li>Feel confident in your job security</li>
    <li>Work the hours you want</li>
    <li>Build the career of your dreams</li>
</ul>
```

As you can see, HTML wraps the content of a website and gives it structure. Here you see a second level headline and an unordered list with four list items. Nice, right?

2

CSS makes the building more attractive and inviting. Think of CSS as like the paint color, the flooring, the trim details, and the interior design. It can turn that barebones building into something people actually want to live and work in.

Here's what some of the CSS from the Skillcrush site looks like:

```
.blog-landing p {
   font-size: 16px;
   line-height: 25px;
}

.blog-landing .entry-excerpt + p {
   max-height: 100px;
   overflow: hidden;
}
```

As you can see, the CSS dictates what the HTML should look like, what its font-size is, its line height, and its width.

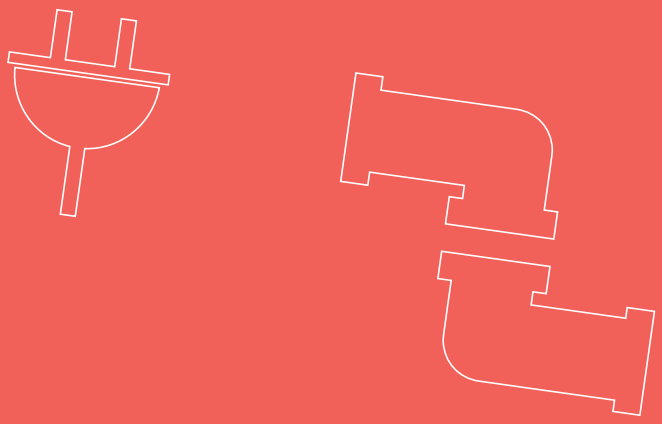
3

JavaScript is another important part of many modern websites. Like CSS, it's not required, but it sure can be nice. In a building, think of JavaScript like the electrical and plumbing systems, the parts of a house that are active and DO things for you. You don't need electricity or plumbing in your house, but you're gonna want 'em!

Here's an example of JavaScript from the Skillcrush site:

```
$('.show-transcript').on('click', function(){
$('.transcript-wrap').removeClass('full').slideToggle()
.toggleClass('show');
   if($('.transcript-wrap').hasClass('show')){
         $(this).text('Hide Transcript');
   } else {
         $(this).text('View transcript');});
```

What this JavaScript code does is show (or hide) the transcripts below a video.

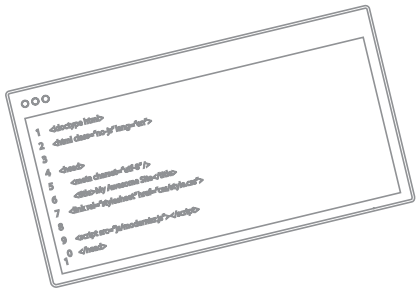


step 3

THE TOOLS OF THE TRADE

As in any industry, there are certain tools you'll need to get comfortable with for a successful tech career. Here are the most common tools you'll want to learn, though different companies might use additional tools or alternatives to the ones mentioned below.

Git: Git is a version control software that developers use for keeping track of code history and changes. Version control software makes it easier to see how code has changed and evolved over time, and also makes it possible to rollback to earlier versions in the event that a new version of the code creates problems or bugs. Git also makes it possible for multiple developers to work on the same code without having to worry about overwriting each other's work.



Text Editor: Professional web developers generally use text editors specifically designed for coding to write and edit the code they create. The difference between a developer-friendly text editor and a plain text app is that the former uses syntax highlighting (usually in the form of different colors for things like HTML tags, CSS, elements, PHP, comments, and the like) to help keep your code organized. Syntax highlighting also makes code about a billion times easier to read and write properly (since it will usually highlight when your code is incomplete or, in some instances, incorrect).

UNDERSTANDING GIT

1. Git is a version control system that keeps track of all the changes you make to your code files. It works a lot like Track Changes in Word.

   Git can be used via fancy Git software, but most often, you will use Git via your computer's terminal by typing commands like this one:

```
git commit -m "This is a git commit message. It's where
I write a note to myself about the work I just did."
```

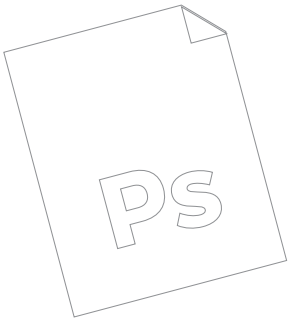
2. When you work as a professional developer you'll want to use Git from the very beginning of every project. What you'll do is:

   - Start tracking your code files with Git
   - Make updates to your code files
   - Save those updates & log those changes in Git with a short note to yourself about the code edits you made
   - Rinse & repeat until your project is done

   This will help you keep track of your code as you work on it, organize your changes, and make sure you have a copy of your work in case anything goes wrong.

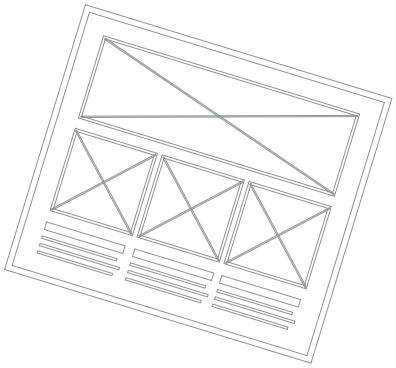
3. Where Git will be especially useful, however, is when you work with other developers on the same codebase. By using Git you can all work on the same code files at the same time without worrying about overwriting or accidentally losing each other's work.

   You can learn more about Git at: git-scm.com.



Graphics Software: If you're interested in design, you'll need to learn to rock a graphics program to create all of those visuals that you see online. The two most popular ones in the world of web design are Sketch and Photoshop (there are others, but these are the two you're most likely to encounter). Even if you're more interested in web development, becoming familiar with these programs is key so that you can more easily work with the designers on your team.

Prototyping and Mockup Tools: Creating wireframes, prototypes, and mockups is a key part of designing and developing any website or app. Wireframes are essentially a sketch of the layout a site will have (with little or no indication of how the interactive parts work).



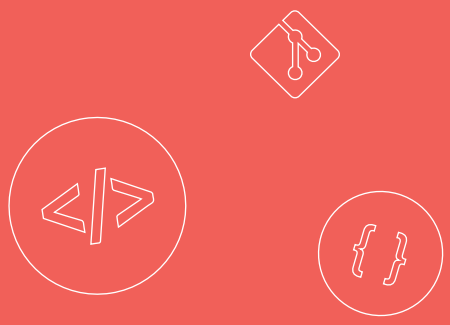
Mockups are generally a bit more polished and give a clearer idea of what the final site will look like. Prototypes are usually interactive, and show how an app or website will actually function, often with fake user data and a full picture of the user experience.

Some designers still start out with pen and paper for wireframes, but eventually those need to be translated into some kind of digital format. Some examples include Wireframe.cc, Moqups, UXPin, Fluid UI, or Axure.

YOUR PHASE ONE CHEATSHEET

1 SKILLS TO LEARN:

- [ ] User Experience Design
- [ ] Web Design
- [ ] HTML
- [ ] CSS
- [ ] JavaScript
- [ ] Git

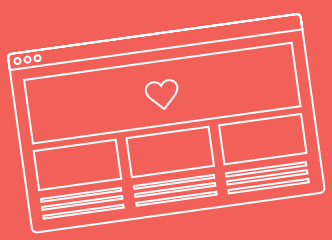


2 SOFTWARE TO TRY:

- [ ] Text Editor
- [ ] Adobe Photoshop
- [ ] A wireframing tool such as Balsamiq or Axure

3 YOUR PHASE 1 GOAL:

- [ ] To build, design, and launch your own portfolio website



WHAT YOU SHOULD KNOW AT THE END OF PHASE ONE

Believe it or not, if you've completed all of the steps laid out in Phase One, you're ready for a number of entry-level tech jobs, including junior designer and even some junior developer jobs, plus plenty of freelance projects!

You're also more qualified for other tech jobs including digital marketing, customer service, and QA (quality assurance) testing.

Ready to learn more? Dive into Phase Two!

Q&A BREAK WITH ADDA

Skillcrush CEO

Q: HOW DO I FIGURE OUT WHAT TO CHARGE?

A: This is one of the most common questions we get from beginning coders and I love getting it because it's one question I have a simple answer to!

$100! You should charge $100 for your first project.

I encourage everyone to start by charging $100 for their first project for two reasons:

1. $100 is enough money that you're going to be excited to get it, but it's not SO much money that you'll be afraid to charge it AND

2. Very quickly, you'll realize that $100 is NOWHERE near enough money to build a website, which will light a fire under your butt to charge much more!

Because the ACTUAL answer is that you should be charging anywhere from $500-$2,000 or MORE to build websites for people, but very few students feel comfortable doing that right outta the gate.

So start with $100. Then get mad at yourself for so woefully undercharging and use that rage to start charging what your brand new skills are actually worth ;)

SKILLCRUSH STORIES

JESSICA

Job: Coder/Author

WHAT KIND OF WORK WERE YOU DOING BEFORE LEARNING TECH SKILLS?

I moved from New York City to Los Angeles in early 2015. When I did that, I also left a very good social work job. I always thought of it as perhaps the best social work job in the City. It was well-funded, quiet, mostly predictable. I wasn't bogged down with paperwork or bureaucracy, and my safety was never in jeopardy.

When I started working in the social work field in Los Angeles it was nightmare experience after nightmare experience. I will spare you the details. I spent six figures on a fancy social work degree and dedicated almost 15 years of my life to the field and came to a point where I said, "I cannot and will not do this. I refuse to work like this." I am passionate about social justice and social change but promised to find other ways to help people.

HOW DID YOU LEARN THE TECH SKILLS YOU HAVE?

One day, I saw a Skillcrush ad on Facebook and decided to look into it further. I researched different programs: free ones, very expensive ones, and Skillcrush, which was not free but not exorbitantly expensive. I decided Skillcrush was right for me because I am the type of learner who needs guidance and support—a real person to talk to and who allowed me to ask questions. And so one night I said, "OK, I'm just going to take the plunge and see what happens." I wasn't sure what I would like—design, coding, etc. But I knew I'd figure it out.

WHAT KIND OF JOB DID YOU LAND AS A RESULT OF LEARNING THESE SKILLS?

About two months after I started the course (or maybe less), I applied for a job in tech doing software support that includes some CSS and HTML for client website redesign. Because I could demonstrate interest and skill in coding, I was able to apply for the job, which required that as a baseline. Before I took the Skillcrush course, I would not have been able to demonstrate this other than daydreams I had about programming robots! So there is a straight line between taking the Skillcrush course and applying and then getting this tech job. It's amazing to me that I was actually able to change careers pretty quickly, successfully, and relatively inexpensively.

The job is remote and the culture of the organization is different than anything I experienced working in the non-profit/social services world. They are supportive and truly team oriented. No one micromanages anyone. They promote learning and professional development, patience, and empathy. They care about our stress levels. And what's particularly important to me is that they make an active effort to be supportive and inclusive—of gender, neurodiversity, and personality, for example.

phase two

SPECIALIZE

ZOOM IN

Basic tech skills open up plenty of of career options, but they also limit how far you can go in a lot of career paths. In order to really get ahead and have a rewarding career, you'll want to specialize.

There are two main options for specializing in tech that we'll talk about here: design or development. Within each of those are additional specialties that can mean more fulfilling (and higher paid) career paths.

Phase two of the roadmap to learning tech and landing your dream career is all about figuring out which path you want to embark on and then what you need to learn for each.

If you can't make up your mind, it's totally fine to learn both! In general, though, you'll want to pick one area to focus on first. Trying to learn both at the same time might be a bit overwhelming.

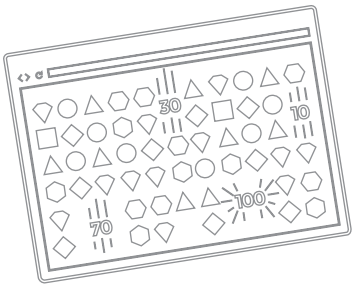
step 4

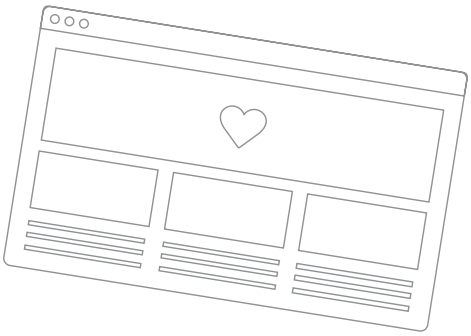
DESIGN VS. DEVELOPMENT

Okay, so you know you need to decide between web design and web development, at least to start. But how do you actually do that? And then from there, how do you figure out what kind of designer or developer you want to be?

One of the best things you can do to get an idea of which career path you're more suited for is to take Skillcrush's free career quiz.

Beyond the career quiz, take a little time to analyze yourself and decide whether design or development is right for you. You'll see what both are all about in the next section.





step 4a

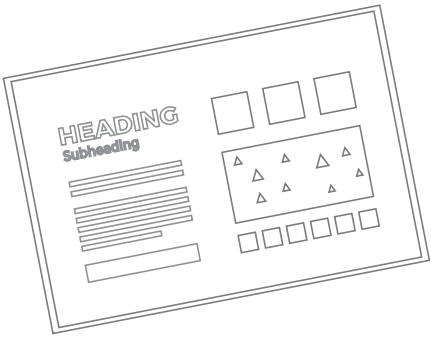
DESIGN CAREER

Crazy for typography, color theory, and striking visuals? Web design is all about the visual experience, using a mix of the client's brand heaps of user research, and a bit of personal intuition.

Job Titles and Salaries for Designers

There are a ton of different job titles out there that fall under the umbrella of "web designer." Here are a few examples, complete with the average salaries:

Web Designer: Average salary $61,000
UX Designer: Average salary $96,450
Visual Designer: Average salary $86,860
Mobile Designer: Average salary $106,470
UI Designer: Average salary $88,800



SKILLS WEB DESIGNERS NEED

1

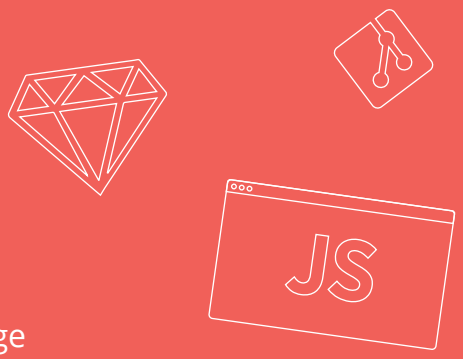
MUST-HAVE SKILLS:

- [ ] Typography
- [ ] Color Theory
- [ ] Branding
- [ ] Responsive Design
- [ ] UX Design (including how to create wireframes, mockups, mood boards, etc.)
- [ ] HTML
- [ ] CSS

2

OPTIONAL SKILLS:

- [ ] JavaScript
- [ ] jQuery
- [ ] Git
- [ ] Sass or LESS
- [ ] Web programming language such as PHP, Ruby, or Python



3

SOFTWARE:

- [ ] Adobe Photoshop and/or Sketch
- [ ] Adobe Illustrator
- [ ] Text Editor
- [ ] Wireframing software such as Balsamiq or Axure
- [ ] Command Line

HOW TO LEARN WEB DESIGN

There are lots of free resources online that can help you learn web design. If you're good at learning independently, then blogs, YouTube videos, and books are a great place to start. If you'd rather have some hands-on help and a community to support you while you learn, then make sure to check out our signature Break Into Tech Blueprint.

INVISION DESIGN SNACKS FREE

This collection of video tutorials from InVision will give you Photoshop and Sketch tips, as well as the basics of web design.

DON'T FEAR THE INTERNET FREE

These design video lessons cover the critical skills you'll need for a career in web design like HTML, CSS, typography, and layout.

1ST WEB DESIGNER: PSD TO HTML TUTORIAL FREE

This tutorial will teach you each step for turning a Photoshop PSD file into a full web page, a responsive website, a Bootstrap website, and more.

DON'T MAKE ME THINK, BY STEVE KRUG

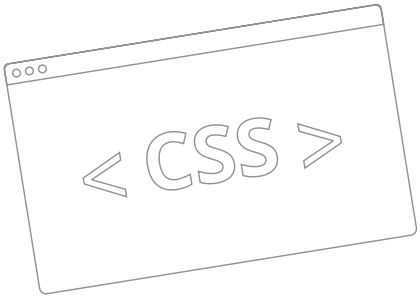
This is the quintessential book on creating amazing experiences for your users.

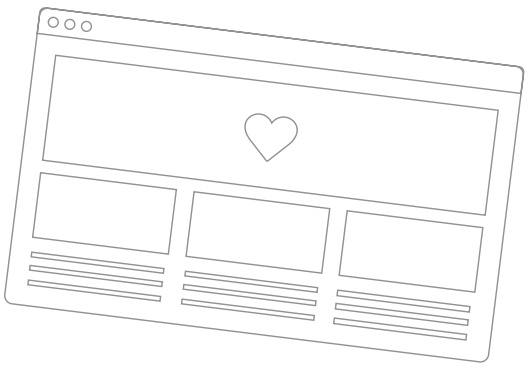
LEARN CSS LAYOUT FREE

This is a slightly more advanced CSS tutorial, though it's still great for those who are just starting out. It assumes you know the very basic parts of CSS, and builds from there.

METHOD OF ACTION FREE

A collection of games, tools, and articles all about learning how to create amazing website designs. It's both a fun way to learn and a great way to procrastinate.





SKILLCRUSH STORIES

MIRANDA

Job: Marketing Specialist

WHAT KIND OF WORK WERE YOU DOING BEFORE LEARNING TECH SKILLS?

Non-profit work and some marketing.

HOW DID YOU LEARN THE TECH SKILLS YOU HAVE?

I read *HTML & CSS: Design and Build Web Sites* by Jon Duckett and got super excited about coding. I learned a lot from the book, but it was information overload! With Skillcrush, I've learned the industry standards for website design preparation (user personas, wireframes, user flow, etc.) which is helpful for organizing my work, and I'm also in the middle of learning CSS. The course breaks everything down really well for me.

WHAT KIND OF WORK ARE YOU DOING NOW AS A RESULT OF LEARNING THESE SKILLS?

I do more marketing using my own email templates, and I got to help redesign our organization's website!

And I got a new job! My interviewer was super impressed by the skills I took the initiative to learn on my own, and I was able to talk the talk. I feel confident going into my new role in just two weeks!

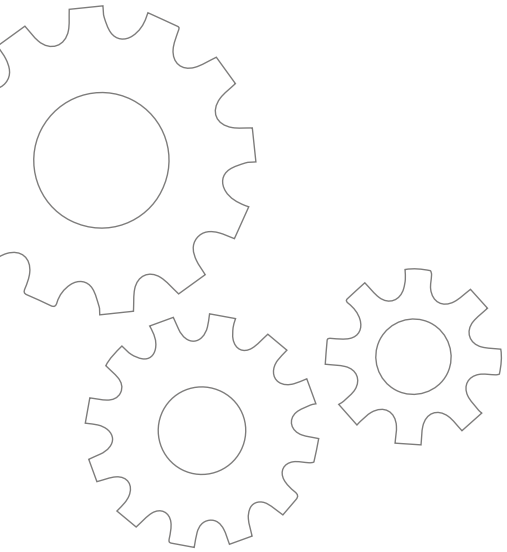
step 4b

DEVELOPMENT CAREER

Web developers take the aesthetic and functional designs from web designers and use code to turn them into live sites. This might be the right track for you if you especially enjoy problem-solving.

Job Titles and Salaries for Developers

Just like with design careers, there are numerous job titles out there that fall under the umbrella of "web developer." Here are some examples, with average salaries:



Web Developer: Average salary $76,000
Front End Developer: Average salary $102,555
Ruby Developer: Average salary $118,670
WordPress Developer: Average salary $71,426
JavaScript Developer: Average salary $110,162
Mobile Developer: Average salary $106,780

SKILLS DEVELOPERS NEED

1

MUST-HAVE SKILLS:

- ☐ HTML
- ☐ CSS
- ☐ JavaScript & jQuery
- ☐ Git + Github
- ☐ A back-end programming language such as PHP, Ruby, or Python
- ☐ MySQL and/or another database querying language



2

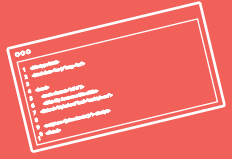
OPTIONAL SKILLS:

- ☐ Sass or LESS
- ☐ Responsive Design
- ☐ Photoshop and/or Sketch
- ☐ UX Design
- ☐ Ruby/Ruby on Rails
- ☐ PHP/WordPress
- ☐ Agile best practices
- ☐ Object Oriented Programming


3

SOFTWARE:

- ☐ Text editor
- ☐ Command Line
- ☐ Adobe Photoshop and/or Sketch




HOW TO LEARN WEB DEVELOPMENT

Just like with web design, there are fantastic resources online for learning development. In addition to books, blogs, and tutorials, there are also interactive apps like Codecademy that can help you learn.

If you're ready to commit to learning web development, the Skillcrush Break Into Tech Blueprint will give you the added benefits of a community of alumni and instructors to help you along the way.

JAVASCRIPT & JQUERY, INTERACTIVE FRONT-END WEB DEVELOPMENT BY JON DUCKETT

This book shows you how to read and write JavaScript, along with the basics of computer programming, all in a simple, visual, and beginner friendly way.

CODECADEMY JAVASCRIPT FREE

This free course teaches fundamental programming concepts, like data types, functions, loops, control flow, and objects.

LEARN PYTHON THE HARD WAY FREE

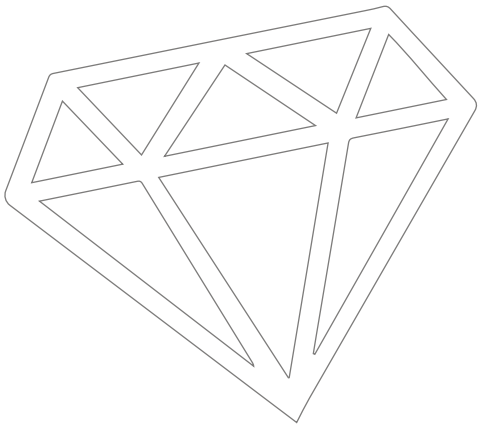
This book is aimed at those who are completely new to coding, and will teach you the foundations you need to dive into more complex Python books and resources.

TRY RUBY FREE

This super basic 15-minute interactive tutorial will give you a basic idea of how Ruby works and how to use it. It's a perfect starting place if you're not sure if Ruby is the right language for you to learn.

RUBY ON RAILS TUTORIAL

This book and screencast series teaches you to build a real, production-ready app with Ruby on Rails.



SKILLCRUSH STORIES

SAMI

Job: Associate Salesforce Administrator

WHAT KIND OF WORK WERE YOU DOING BEFORE LEARNING TECH SKILLS?

When I enrolled in my Skillcrush Blueprint, I was doing well in my role as a Project Coordinator, but the next rung on the ladder was Project Management and I knew that wasn't my passion.

HOW DID YOU LEARN THE TECH SKILLS YOU HAVE?

I dabbled in Khan Academy's Javascript lessons, but I didn't stick with it. Most of what I had learned in middle and high school was also wayyyy out of date.

At Skillcrush, I learned HTML, CSS, JavaScript. I learned about Git and Github, and a little Ruby. I learned programming concepts that turned out to be critical building blocks—I learned to think like a programmer.

And, the community sent me towards tons of other FREE resources to expand my knowledge. I learned to embrace and grow from mistakes and to power through perceived roadblocks.

WHAT KIND OF JOB DID YOU LAND AS A RESULT OF LEARNING THESE SKILLS?

Well, it took some time and patience. I really wanted to stay with my current company, so I had to wait for an internal opportunity to arise. I was originally trying to get into our support department, because they work with databases and a little code, but there were a few aspects I wasn't thrilled with. I'm so grateful that the stars didn't align on that one, because a position for Salesforce Admin opened up in our IT department, and it's a much better fit. I love my team and my work, and I'm really proud to be in this role. I've been able to blow expectations out of the water, and I think that's in large part thanks to what I learned in my Blueprint. It took about a year to get here, but it is so, so worth the wait.

Q&A BREAK WITH ADDA

Skillcrush CEO

Q: WHAT ARE COMMON CHARACTERISTICS OF YOUR MOST SUCCESSFUL STUDENTS?

A: It's funny, Skillcrush is an online education company, but the truth is that we EASILY learn as much from our students as you all learn from us!

And our favorite things to learn are tips and techniques from our most successful students.

After helping thousands of students learn to code, start making money, and land new jobs with their coding skills, we've definitely seen some patterns among our most successful students. And the good news is that you can totally learn from them too!

Here's what we've noticed that our most successful students have in common:

- They make sure to finish & launch their web projects, especially their portfolio website.
- They attend hackathons and meetups.
- They aren't afraid to put themselves out there, meaning they start telling everyone about their new career right away.
- They know that being out of their element and falling on their face a few times is all part of the awesome learning journey they are on.

phase three

GO PRO

LAND YOUR FIRST TECH JOB!

With the skills you've learned in the first two phases, you are perfectly poised to land your first job in tech. In fact, most successful designers and developers don't know anything beyond the required skills outlined in the second phase when they get started.

But knowing you have the skills to land a job in tech and *actually* landing one are two different things. You'll need a few key things to get your first tech job:

- A fantastic portfolio
- An standout resume
- A cover letter tailored to each job opening
- The patience to keep at it

Beyond that, you'll need to figure out where to find the best tech jobs, whether you're looking for something local or a remote job you can do from anywhere. We've got you covered.

step 5

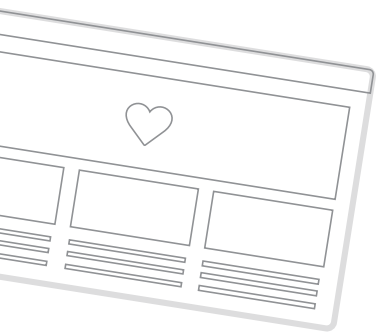
CREATE AN AMAZING PORTFOLIO

Regardless of which career path you choose, there's one thing that's an absolute must-have: a stellar portfolio. You'lll need to show off your design or dev skills as well as what sets you apart from all the other designers and developers out there.

An employer—whether they're looking to hire you permanently or as a freelancer for just one project—is going to insist on seeing examples of the work you've done. And while you can send them a list of links to your work, you're way better off putting everything into a portfolio that tells your story the way you want it told.

Skillcrush has plenty of resources for putting together a killer portfolio that will help you land your dream job. Here are some of the best:

- Everything You Need in Your Portfolio to Get Hired
- 10 Tips for a Stand-Out Web Developer Portfolio
- Create the Perfect Portfolio with These 17 Tips
- 24 Essential Portfolio Tips For New Techies
- How to Build an Impressive Portfolio When You're New to Tech



step 6

CRAFT A STELLAR RESUME

Hiring managers often they don't even read your resume, let alone spend time with a printed copy. They're too busy sorting through hundreds of email applications, PDF attachments, LinkedIn profiles, and long, wordy cover letters.

That means you need to get smarter about writing resumes that end up in the right hands. Beyond that, you have to write a resume that stands out, floats to the top of the pile, and ends up on the desk of the person who will hire you.

If you're a developer, you can probably get by with a standard resume and a killer portfolio (though a well-designed resume certainly won't hurt your chances).

But if you're a designer, consider your resume *part* of your portfolio. It's the first piece of design work a potential employer will see from you, and something with a design that stands out might make the difference in whether they take the time to look at your portfolio.

We've put together tons of information about how to create a fantastic tech resume, including the free Ultimate Guide to the Perfect Resume ebook (complete with examples!).

Once you've downloaded that, be sure to check out our other resume resources:

- 12 Skills Your Resume Should Already Have
- 22 Things to Remove from Your Resume Immediately
- 9 Rookie Mistakes That Will Ruin Your Resume
- 10 Things Your Resume Needs When Applying at Startups
- 12 Skills to Highlight on Your Remote Resume

Q&A BREAK WITH ADDA

Skillcrush CEO

Q: HOW WILL I KNOW WHEN I'M READY TO START APPLYING FOR JOBS?

A: This is a great question! But unfortunately, it's not the easiest one to answer because the answer is... it depends!

Helpful, no?

That said, a few things do come to mind:

1. There is a minimum set of skills that I encourage all students to get comfortable with before I would consider them "ready," and those are:

   - Basics of web & UX design
   - HTML & CSS
   - Git
   - 1 more specialized skill (AKA more design skills if you want to be a designer, more development skills if you want to be a developer)

2. So if you do not at least have these skills under your belt, I'd say you're not ready YET.

   This question, to me, really demonstrates why it's so critically important that you don't try to go it alone! The truth is that you need other people—friends, mentors, fellow students, instructors—to help you determine when you're ready. So no matter how you do it, take the time to build up a community to support you in your learning to code journey.

3

Finally...the answer is: WAAAAY sooner than you'd think. If there's one mistake I see beginners make over and over is that they wait too long to get started freelancing and applying for jobs.

The thing is, learning how to apply for jobs and what employers are looking for and how to pitch and land a client, aren't skills that you learn once you're ready, those are skills you need to learn TO BE READY. So if you wait until you feel "ready" to learn those things, you are going to wait way too long and rob yourself of valuable skills you could be learning and MONEY you could be earning.

step 7

FIND THE BEST JOBS IN TECH

Search for "web designer" on LinkedIn's Jobs board and you'll get nearly 4,000 openings in the US alone (and that's just *right now*—there are many times that number of openings each year). Search for "web developer" and you'll get over 5,500 openings in the US.

That's a lot of openings. And that's just on one job board.

If you're looking locally for a job, then that will narrow down the number of openings considerably. If you're looking for a remote job, then check out Skillcrush's roundup of 37 different job boards that cater to remote workers. And if you're looking for freelance work, check out the 25 Top Sites for Finding the Freelance Jobs You Want.

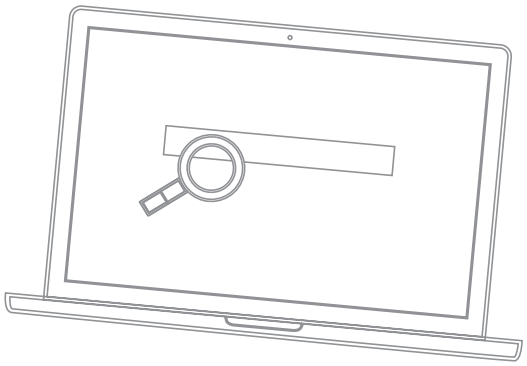
WHERE TO LOOK FOR ENTRY LEVEL JOBS

THE MUSE
The Muse offers up job listings as well as tons of career resources to help you land the job. They also offer great career content.

AFTER COLLEGE
After College can help you find entry-level jobs and internships when you're just starting out.

IDEALIST.ORG
If you want to work for a non-profit, then Idealist is the place to search. They have listings for both regular jobs and internships.

AUTHENTIC JOBS
Authentic Jobs lets you filter search results based on experience level. They include numerous remote job listings, too.



step 8

CREATE TAILORED COVER LETTERS

It happens all too often: you write a "standard" cover letter and then send it out to every job you apply for.

Guess what? That's a *horrible* way to land a job.

Every employer out there is bombarded with those kinds of cover letters. And every employer is looking for slightly different things from the people they hire.

Do you see the conflict here?

One more thing: no one does cover *letters* anymore, since 99 percent of jobs are applied for electronically or via email now. So your cover email needs to be short, sweet, and to the point.

Not sure how to do that? Check out the Ultimate Guide to the Perfect Email Cover Letter for complete instructions on crafting an eye-catching (interview-getting) cover letter. Then head to these articles for even more:

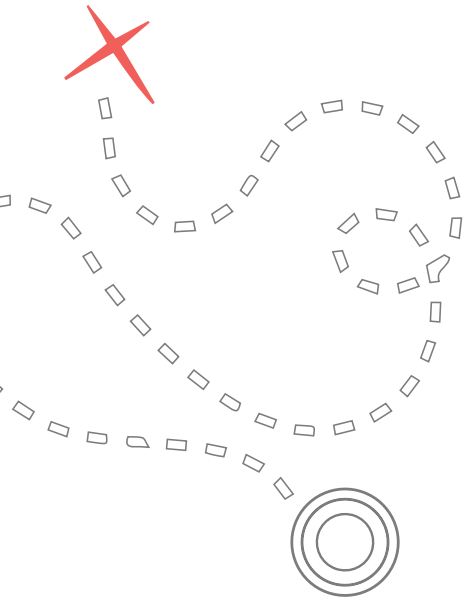
- How to Write a Cover Letter That Will Get You Hired
- 16 Secrets for Writing Cover Letters
- 10 Things You Should NEVER Do In Your Cover Letter

step 9

IT'S A NUMBERS GAME, SO STICK WITH IT

Landing that first job can feel like an insurmountable task some days, but there are a few things to keep in mind that will make the job search a little easier.

First, you DON'T have to meet 100 percent of the qualifications for a job in order to apply. Think of the job's "requirements" as an employer wish list. They know they're probably not going to find someone who meets every single requirement (and if they do that person will be in super high demand and will likely be fielding offers from a bunch of companies), but if they list what their "ideal" candidate would have, then they're likely to get applicants who are at least close to that person.



Second, set a goal for yourself for the number of jobs you'll apply for each day or week. It might be one job every day, or 10 jobs every week, or 100 jobs total. Don't set the goal too high, though, because you want to have time to tailor your resume and cover email to each job opening.

Third, don't let yourself get discouraged. Think of every job you don't hear back from as just one step on the path to your dream job. And take every job application as an opportunity to learn more about the company, the industry, and the types of jobs that are available.

And finally, be sure to keep your portfolio, resume, and cover letters up to date with your latest accomplishments and projects. Eventually, something will catch the eye of the perfect employer.

SKILLCRUSH STORIES

ANTONIA

Job: Junior Web Developer & Freelance WordPress Developer

HOW DID YOU GO ABOUT LEARNING TECH SKILLS?

I got my tech skills through a mixture of learning (an e-commerce diploma, which I rarely used) and on-the-job learning when I worked in a travel agency and helped build a website. I left work to have my child, feeling like I was floundering as a developer in a world of younger, smarter people. I knew I could do the job, but was lacking in confidence and feeling like an imposter.

Then, six years after having my first child, aged 43, I got back into web development. I signed up for the Skillcrush free Bootcamp and the most inspirational newsletters I've ever received. They really helped me to believe I could become a developer again, having written myself off.

After this, I embarked on a web development/Rails course with Career Foundry (the Skillcrush course, while seriously tempting, was aimed at beginners and I had some coding skills). Finally, I happily signed up for a Blueprint with Skillcrush. The WordPress class cemented my abilities as a freelance web developer, flooded my system with invaluable career advice, and, perhaps most importantly of all, was hugely responsible for helping me to build up the confidence to believe in myself as a developer again and to network successfully.

WHAT KIND OF JOB DID YOU LAND AS A RESULT OF LEARNING THESE SKILLS?

Some years ago, I was traveling down the escalator in the London Underground and I saw an advert: “Learn computer skills.” I had a vision of sitting at a desk in my home, working flexible hours, children running around me, earning a decent wage and dictating my own hours.

This may not be everyone's idea of their future career—but my dream literally has come true and is really working. Now that I have tech skills I do work flexible hours from home as a freelancer, plus I also recently landed my first "proper" job in seven years as a junior web developer, which also allows me to work from home for most of the week.

bonus step

KEEP LEARNING MORE ADVANCED SKILLS

If you want to keep advancing your career, then the key is to keep improving and learning new things.

Let's say you start out as a front end developer. Learning additional back-end skills is a great way to open up more career options. Or you could spend time learning more libraries and frameworks for front end development.

Tech is a constantly evolving industry. And because of that, it's key that you stay on top of how your particular specialties keep growing and evolving. Standards improve each year, new technologies and techniques are released, and old techniques disappear or fall out of favor.

One great way to keep learning is obviously to keep taking classes to improve and expand your skills. Other ways include reading awesome design and development books, listening to podcasts from industry leaders, following the best designers on Twitter, subscribing to email newsletters, and reading tech blogs.

Joining local meetup groups, networking with others in tech, attending conferences, and finding a mentor also help to keep the learning going!

5 GREAT WAYS TO KEEP UP WITH THE TECH INDUSTRY

Tech news aggregators and communities like:

- HackerNews
- Webdesigner News
- Stack Overflow
- Designer News

Newsletters such as:

- FrontEnd Focus
- Ruby Weekly
- JavaScript Weekly

Blogs like:

- Smashing Magazine
- A List Apart
- Webdesigner Depot
- CSS Tricks

Podcasts like:

- Let's Make Mistakes
- The Web Ahead
- This Developer's Life
- Happy Monday

The Skillcrush Blog & Newsletter

LEARN TO CODE WITH US. MAKE MONEY WHILE YOU DO IT.

Are you pumped about working in tech but not sure where you'll fit in? Worried you might not have what it takes? Do you know you want to take advantage of the flexible work schedules and high salaries that are common in the tech industry but have NO IDEA where to start?

Start here: The Skillcrush Break Into Tech Blueprint.

You'll gain the confidence and marketable skills you need to start earning money within 3 months, find fulfilling and flexible work within 9–12 months, and level up your career for *good*. And with 1-on-1 support from your first line of code to the last line of your cover letter, we'll be with you every step of the way.

LEARN TO CODE

Get hired. Make more money. It's that simple!

ENROLL NOW!

JUST SOME OF THE CAREERS YOU CAN PURSUE WITH OUR BREAK INTO TECH BLUEPRINT:

WEB DESIGNER
AVERAGE SALARY: $61,000

WEB DEVELOPER
AVERAGE SALARY: $76,000

FREELANCE WORDPRESS DEVELOPER
AVERAGE SALARY: $85,000

LEARN MORE ABOUT THE BREAK INTO TECH BLUEPRINT HERE

HAVE YOU JOINED *the* CLUB?

If you found this resource (and the ones listed here) helpful, there's even more to discover on the Skillcrush blog.

We share top career advice, inside tech tips, real-life stories of getting into tech, and tons of FREE guides, worksheets, and resources.

Sign up for our newsletter now so you won't miss a post. Each week, we send you two researched, detailed, easy-to-read articles to help you harness the power of tech and get the career you deserve.

Join thousands of happy readers! Here's what a few of them have to say:

"I read a blog post every day and learn something new. The resources you give us are priceless!"

SARA EVANS

"I have been on your mailing list now for 8 months and in that time alone I saw the platform grow so much. Seriously, in the online education space you guys have some of the best content marketing. (Some online learning platforms neglect it entirely, while others only post articles about latest updates or course offerings, which is a bore.) Skillcrush actually generates valuable content for readers."

LAURENCE BRADFORD

"The 99 Tech Terms You Need to Know When You're New to Tech has been a big help! I love materials that cover the basics."

APRIL HARRIS

"Skillcrush, Thank you for offering such a great newsletter to your subscribers. The content that you send out is always helpful and something that I can refer back to as I improve my coding skills. I especially love your PDF quicksource guides."

JUBILEE GRACE

"Your email newsletters, videos and articles are so inspiring, I really look forward to them and I love your attitude, all of you, and the great images. You make me feel included in a world where it seems everyone is an expert, and I wanted to say thanks."

ATTY CRONIN

"I love your blog posts! They always end up being my lunchtime reads at work!"

CAROLINE KIM

WHAT *are* YOU WAITING FOR?

Sign up to our newsletter now so you won't miss a post.

SIGN UP NOW

THANKS FOR JOINING US.

Feel free to email us with any questions at hello@skillcrush.com