The Ultimate Guide:

How to Bootstrap Your Career in Web Development

First Edition

CODING DOJO®

Section 1

What is Web Development?

Web development is one of the fastest-growing occupations in the early 21st century.

The term web developer is used ubiquitously throughout the tech industry, yet unsurprisingly – to those not already a part of the development community – it isn't always clear what web development is or what a web developer does.

To answer these questions effectively we must first delve a tiny bit into the realm of websites themselves: How a website recognizes when a user visits the site and performs the necessary function to display the appropriate page to the user.

Digital Ping Pong: The Inner Workings of a Modern Website

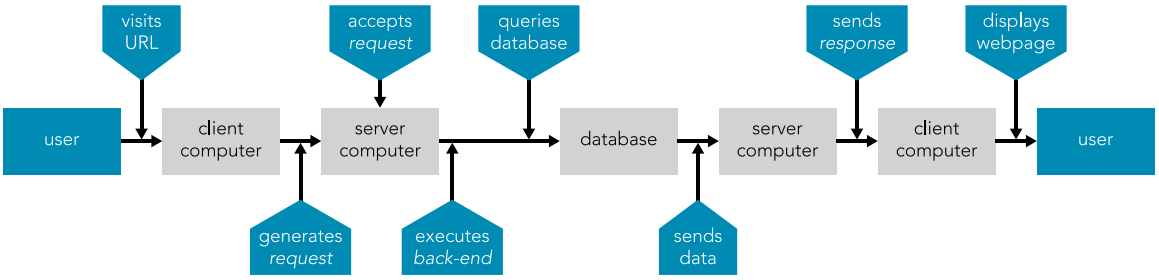
While every website is built slightly differently, there are a few fundamental components that handle every interaction between a user and the site:

Client: The local computer (desktop/laptop) or device (phone/tablet) the user is interacting with to access the website.

Server: The remote computer that "physically houses" all the files (and thus code) that make up the website.

Database: A sub-component of the remote **server**, the **database** is a large series of data tables used to store all the dynamic information generated or used within the website. For example, the account information of a logged in user would be stored in the database.

With our three fundamental components identified, we can briefly examine how a website recognizes a visiting user and ultimately displays the appropriate page for viewing. The following diagram provides an approximate illustration of the process.



As an example, let's imagine Jenny wishes to visit google.com.

1. Jenny first enters the url of the website (google.com) in the browser on her local computer (the **client**).

2. Jenny's computer generates a *request* that is sent out to the server computer, which then accepts the *request*.

3. The **server** runs (or executes) the *back-end* code, usually grabbing data from (or querying) the **database**.

4. The **database** sends the requested data back to the **server**.

5. The server takes the data and executes the *front-end* code to produce a *response*.

6. This *response* is sent back out to the client where it is then displayed (or rendered) on the **client** computer as a standard web page.

The end result is that Jenny is now looking at the Google homepage as expected, all within a matter of milliseconds in most cases.

What's My Line?: Where the Web Developer Fits In

Now that we've explored the fundamental process of how a web page is displayed to a user, we can dive into the deep end and discover where web development comes in and how it is applied to allow that magical ping-pong process to occur.

As a broad definition, a web developer's primary purpose is *to create a functional website that performs a set of particular, defined functions*. Accomplishing this goal breaks down into three core phases.

Phase 1: Planning

During this preliminary phase, a web developer will work closely with the client and other developers to plan the structure and core concepts of the site. This first phase is an ideal time to decide how the various pages and components of the site link to one another (also known as a *sitemap*). While the *sitemap* can take on many forms, it should effectively outline how a user will navigate around the site.

During the planning phase, it is also vital to determine how the *client* will interact with the site as well. If the client will be posting blogs or adding products to the online store component, it is the planning stage that should specify exactly how these tasks will be performed.

Phase 2: Design

The design phase is when the visual look and feel of the site is determined. This entails everything from color palette and fonts to page width and static image placement. If the planning phase determines *what* the user will do with the site, the design phase determines the *where* and the *how*.

Typically a *mockup* for each page or component of the site is created in Photoshop by a designer or multi-disciplined developer. This *mockup* should typically include every visual element that is expected in the final page and is thus representative of what the client wants to see when visiting the website.

Throughout the design process, it is critical to consider the target audience and demographic of the website. *The design should correspond to both the appropriate user base the site is marketed toward as well as the intended use of the site.*

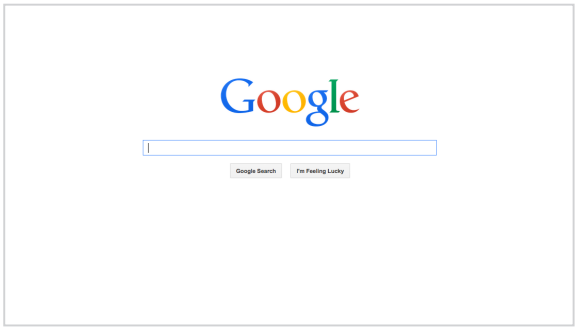


Image: Google.com

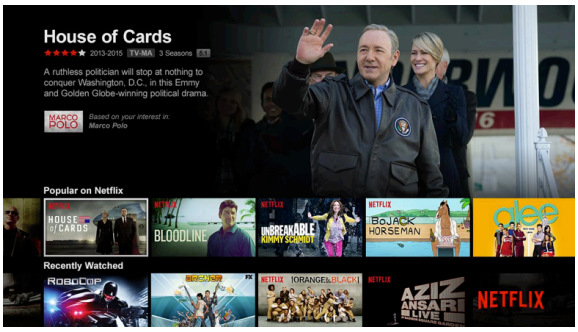


Image: Netflix.com

For example, Google is intended for all audiences and emphasizes speed and efficiency of search results, which fits the minimalist design Google uses, including a visual look comprised of almost exclusively text.

Netflix, on the other hand, is all about audio and visual content and thus focused on a very colorful, full-screen design to highlight the multitude of shows available on the platform.

Phase 3: Development

The development phase is of course the most crucial for web developers involved in the project, and where the majority of time and energy will be spent producing the final product.

For most modern websites, the development process is broken down into three architectural components that the web developer will intermix throughout the procedure.

> **Application Logic:** Often referred to as the *model* component, this represents the majority of the back-end code a developer will write to make the site function as expected. This logic is also where a developer must understand and utilize the connection between the site and the database that powers it.
>
> **Presentation:** Commonly known as the *view* component, this is where the *mockup* that was created during the design phase is used by a developer to **recreate** the look and feel of the *mockup* image utilizing the basic building blocks of HTML and CSS, such that the end result is a webpage that looks identical to the *mockup*.
>
> **Connection:** Also referred to as the *controller* component, this code defines the connections between the back-end business logic that handles the grunt work of the site and the front-end pages that users will access: It **connects** the back- and front-end code together.

Choices, Choices, Everywhere: The Varied Disciplines of Web Development

The exciting thing about web development as a field is the multitude of differing disciplines that a newcomer can focus on, depending on his or her particular skillset and desires. While the **core** of web development is generally considered to be from a coding perspective and thus an education in coding is expected, there are numerous disciplines within the web development field with slightly varied focus.

Graphic/Visual Designer: The visual designer is often well-trained in the arts, utilizing Photoshop and others tools to create mockups for pages or entire websites that will please the client and appeal to the audience. In some development shops these positions are "codeless," while often in others, visual designers would be expected to convert visual mockups into workable front-end code.

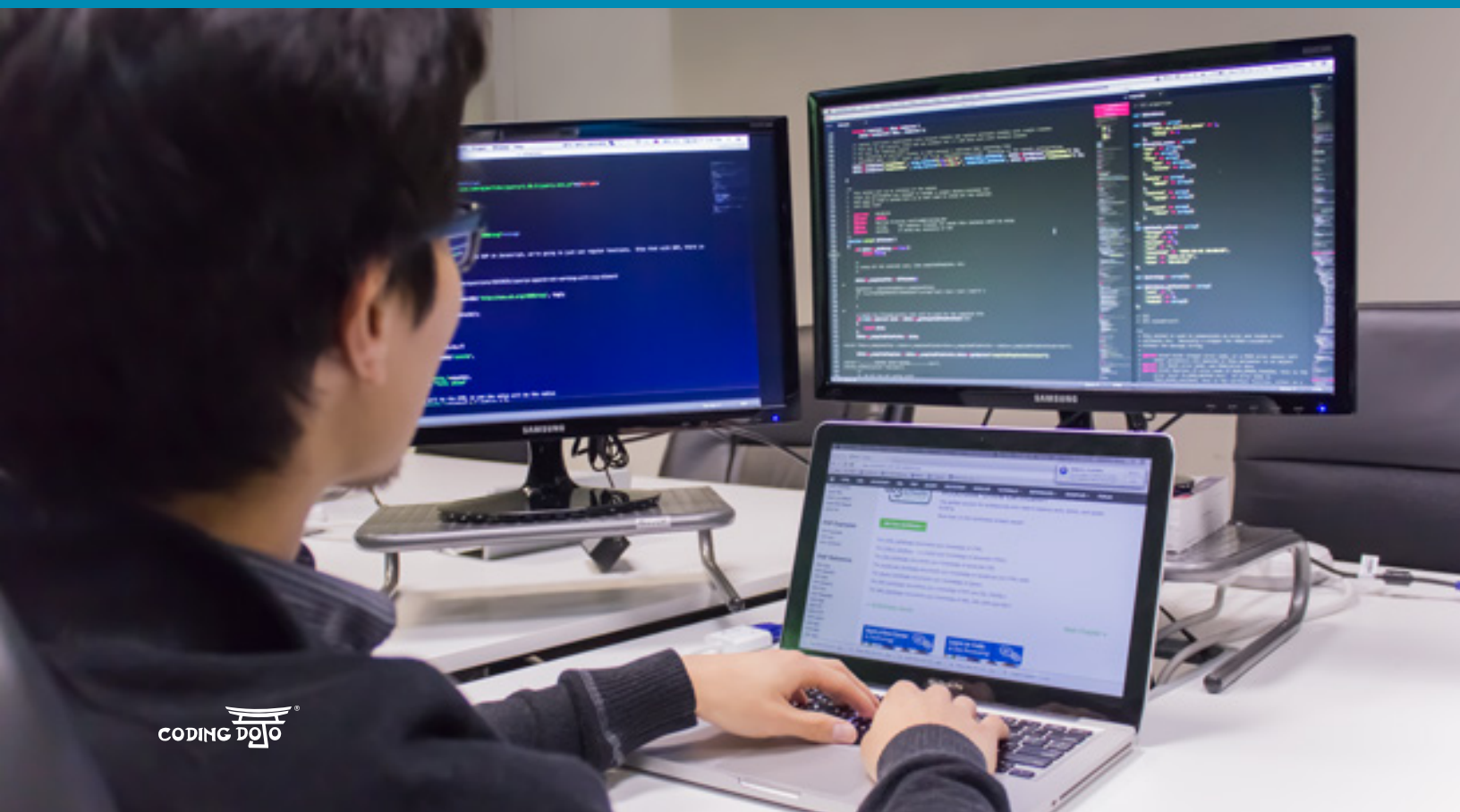
Front-End Developer: A developer focused on the look and feel of the site (the *presentation* layer discussed above) and almost exclusively utilizing the languages of HTML, CSS, and JavaScript.

Back-End Developer: A back-end developer writes all the code necessary for the core logic of the website: Grabbing data from the database and molding how that data is appropriately used and displayed to the user through the front end. Languages commonly used for back-end development are varied, but a handful among the most popular are Ruby on Rails, Python, PHP, and Node.js.

Full Stack Developer: A much-lauded position, and rightfully so, the full stack developer is one who is adept at **all** aspects of the development process and is capable of contributing code and functional solutions every step of the way, from planning and design to both front- and back-end coding.

Section 2

Why the Time is Now to Start a Career in Web Development

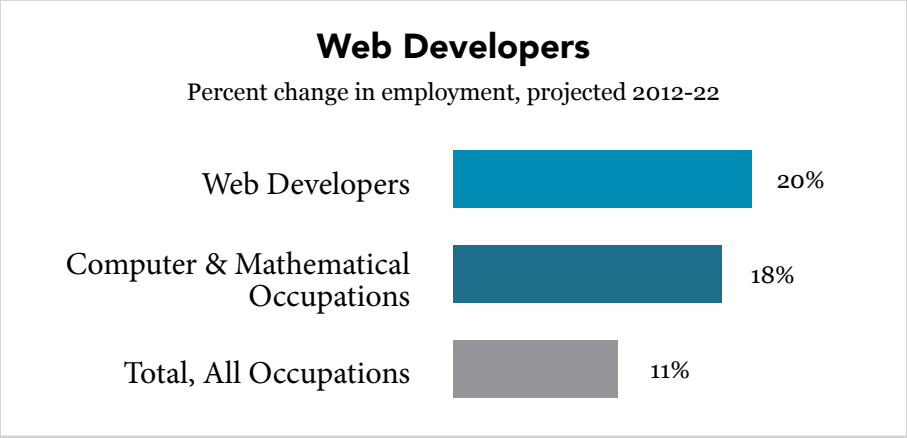


Web development is one of the fastest growing and most in-demand occupations to date, and with no sign that this train will halt anytime soon, there's no better time for new up-and-comers to get into the field. Even *if* the demand for strong developers weren't as high as it is, there are even more reasons beyond the high demand to consider a career in web development, from freedom of scheduling and choice of employer to working/living locale and unavoidable on-the-job training.

Let's explore just a few of the multitude of reasons that a career in web development is a sure bet for many years to come!

Generous Salaries and Rapidly Growing Demand

As our modern society moves deeper and deeper into the digital age, fewer of the standard offline methods of doing things will remain the norm, while instead these methods transition into their more streamlined online counterparts. According to the U.S. Bureau of Labor Statistics (BLS), employment of web developers is projected to increase 20 percent within the decade between 2012 and 2022, nearly **doubling** the growth of all other occupations:

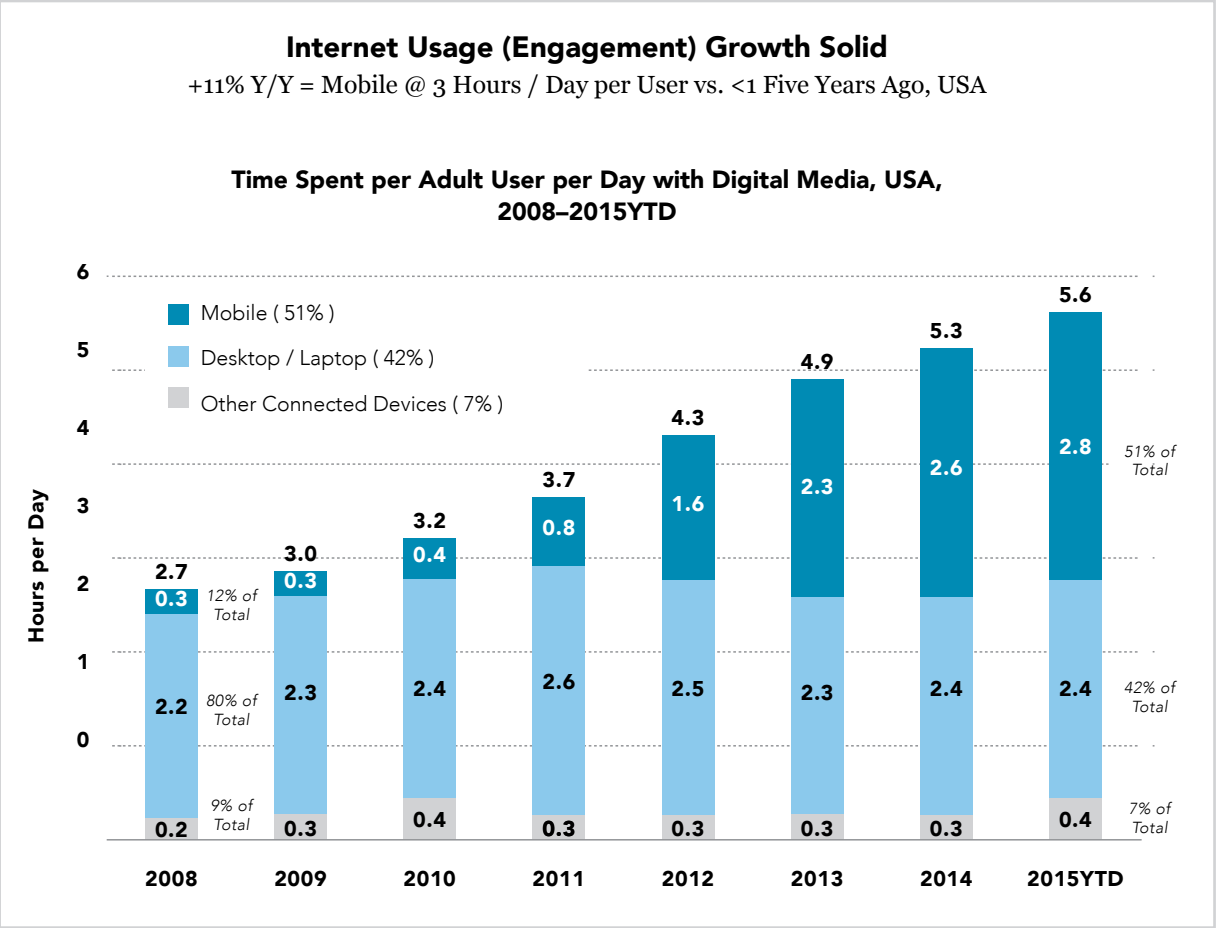


Note: All Occupations includes all occupations in the U.S. Economy.
Source: U.S. Bureau of Labor Statistics, Employment Projections program

The expansion is largely based on the ever-increasing drive toward eCommerce, as online purchasing is expected to grow faster than the retail industry.

An overwhelming surge of mobile device usage is also largely a cause of this projected growth, with a great deal of both new and existing websites requiring designs and development that support a mobile platform.

In fact, according to the 2015 Internet Trends report, which is published annually by the venture capital firm Kleiner Perkins Caufield & Byers, mobile device usage for digital media consumption has handily surpassed that of traditional desktop platforms with no sign of slowing.



Note: Other Connected Devices include OTT and game consoles. Mobile includes smartphone and tablet. Usage includes both home and work. Ages 18+; time spent with each medium includes all time spent with that medium, regardless of multitasking.
Source: eMarketer 9/14 (2008-2010), eMarketer 4/15 (2011-2015).

Perhaps just as critical to those looking to get their feet wet in the field of web development is the pay:

How do web developer salaries match up to other similar occupations?

According to the BLS, the median annual wage in the U.S. for a web developer in May 2012 was $62,500, which as seen in the diagram below, is roughly middle of the pack for similar computer-related occupations, and **well above** the median across all occupations at $34,750.

Median Annual U.S. Wages, May 2012

Computer & Information Systems Managers	$120,950
Software Developers	$93,350
Information Security Analysts	$86,170
Computer System Analysts	$79,680
Database Administrators	$77,080
Computer Programmers	$74,280
Web Developers	$62,500
Multimedia Artists & Animators	$61,370
Computer Support Specialists	$48,900
Graphic Designers	$44,150
Total, all occupations	$34,750

Source: U.S. Bureau of Labor Statistics

Being Choosy: Web Development Offers Unparalleled Freedom

Beyond the financial benefits of a development position are the less measurable but equally important freedoms offered by such a career path. Unlike most traditional occupations, web development positions are so numerous and in such high demand that developers or prospective employees are given far more choice when it comes to factors like living area, hours, working environment, and more.

Self-Employment or Corporate Employment

While there are no shortage of development firms or even corporate businesses looking for talented web developers for hire, due to the digital nature of the work being performed, web developers are often free to choose whether to work for a larger development firm, become contracted by a corporate business, or even become self-employed and begin a small little development business outside of their own garage. According to the BLS, roughly 25 percent of all web developers are self-employed.

Workplace Locale

Similar to choosing whom to work for, web developers are often given far more freedom of *where* to physically do the day-to-day labor of their craft compared to traditional occupations. A 2015 survey of more than 16,000 developers using the very popular StackOverflow discussion platform shows that 29% of developers work remotely at least part-time and often from home – a sizeable increase from the 21% the year prior.

Perhaps surprisingly, the same 2015 StackOverflow survey also indicates that *developers who work remotely full-time earn roughly 40% more than developers who are purely office-bound.*

Brain Power: Constantly Improving Your Technical Prowess

Perhaps one of the least tangible benefits of a career in web development, but certainly one of the most beneficial, is the constant and unavoidable personal growth throughout your career. Both while initially learning the ropes and after being in the industry for many years, web developers will naturally find themselves continuing to grow their breadth of knowledge and technical prowess as new technologies emerge and become popularized, or various coding methodologies are adopted and later thrown by the wayside.

While some may view this constant evolution of personal growth as a bit overwhelming or even scary, it brings with it a great deal of job security and stability that other non-technical positions simply cannot provide. As a relative example, looking back just five years to mid-2010 using Google Trends (a measurement of relative search term frequency), we can compare two currently popular languages of PHP and AngularJS: Based solely on relative search trends, we can see that PHP has dropped in search frequency by roughly 38% over the past five years, while the new technology of AngularJS took hold in late 2012 and has since risen dramatically with roughly 11% of the relative search frequency of PHP.

While these are dramatically different technologies and PHP remains a very popular language, it is a strong indication of the rapid rate of change and evolution within the web development space. As new technologies emerge and frameworks take advantage of those technologies – such as the popular MEAN framework that utilizes AngularJS – then employment and job opportunities will be quick to follow.

> *Intelligent web developers will stay abreast of these latest trends and technologies, thus always ensuring themselves a place at the table of employment for years to come.*

Section 3

Do You Have What It Takes? Qualities of Successful Web Developers

Web development is a discipline that primarily focuses on coding, yet it also requires knowledge and capability in a multitude of similar disciplines such as database administration, user interface/user experience (UI/UX) design, and search engine optimization (SEO). Depending on your background and technical prowess, learning the various disciplines to become a capable web developer *may* feel daunting or even overwhelming, but the wonderful thing about web development – and which makes entering the field so amazing – is that *anyone* can truly do it, with a bit of effort and education.

While anyone that has the desire can become a web developer, there are a handful of qualities and attributes that particularly successful developers will possess.

Like many other occupations, a strong web developer should be **creative** and thus able to brainstorm and come up with the best solutions to a problem. A developer must also be **adaptive**, being both willing and able to quickly shift gears as new technologies are popularized. A reasonable amount of **diligence** is also required, displaying a strong sense of focus and attention to detail.

Beyond those basic concepts we'll explore three additional qualities that are more specific to web development itself. As we delve further into the detail of these attributes, consider how each of these qualities relates to you. If you feel lacking in any given attribute, we'll also outline some brief advice on how to improve yourself in a particular area and thus be better prepared for a career in web development.

Technically Savvy

Perhaps most obviously, a web developer needs to be technically savvy. Most web developers will spend a great deal of their work day using a computer of course, but beyond the simple notion of being adept at computer use, there is also the need for a strong understanding of coding/programming.

Most developers will naturally learn multiple languages over time. The key is not necessarily learning the syntax (i.e. the required format) of any given language, but instead to **understand the principles and concepts of programming**. Concepts like true/false values, IF-ELSE logical statements, and many others are used globally throughout virtually every language, so once a developer learns those basic ideas, they can easily be applied to any new language, thus making the learning process far simpler.

> **Tips:** Education and personal research are the best tools to improve your technical savvy. Once you're comfortable with using the computer overall, begin educating yourself about general programming structures and concepts, and then begin a program to develop your own understanding of a specific language or two in which you can start producing workable code. Once you have something tangible that you've created and that you are building upon and trying to debug, you'll gain a much better appreciation and understanding of code as you move forward.

Problem-Solver

To be an exceptional web developer, one must possess an exceptional ability to consider solutions for and ultimately solve problems. Perhaps surprisingly to those outside the field,

> *programming is largely more about* ***thinking*** *than it is about* ***writing code****.*

Thinking, pondering, and considering how to approach a particular roadblock during development or how to create an appropriate solution to meet a client's need is one of the most common activities for successful developers, and is time that should not be neglected nor denied to oneself.

The manner in which a developer spends his or her time thinking about how to solve problems will of course differ from individual to individual, but it is critical to the process that this time be allotted and allowed to occur. A strong web developer is one who is logical and has a methodical approach to resolving issues that will inevitably rise in the day-to-day process of development.

> **Tips:** For those that feel their problem solving skills are subpar, an easy practice to get into that will help improve your ability to discover solutions is to apply a crude version of the scientific method.

1. **Assess** the situation and identify the problem (or your best guess at the problem if you're uncertain).

2. Create a **hypothesis** for how to solve the issue. In most cases, begin with the most obvious possibilities first then work toward the more complicated options.

3. **Test** your hypothesis to see if the issue is resolved but more importantly, to see if the results you expected were altered.

While this will at first feel a bit like a trial and error methodology, it serves a very important purpose and will eventually lead to insight into the fundamental cause of the problem and ultimately a solution.

Researcher

While a strong technical understanding of the code you are working with and the ability to solve problems for yourself are crucial to becoming a strong developer, a skill that is equally as critical to development is the ability to effectively and thoroughly research.

Due to the rapidly changing nature of the field of web development, it is only natural that at some point in the day, a problem will arise that you simply have no idea how to resolve. You've tried and tried to figure it out yet nothing seems to be working and frustration might be getting the better of you. At times like this, the *absolute best resource* you have available to you is the knowledge, expertise, and efforts of **other developers**.

While there are a multitude of sites out there to request help from fellow web developers such as StackOverflow and A List Apart, the absolute best research tool is Google.

> *Learn it, live it, and love it,*

because Google is likely to be the primary method for helping you solve particularly tricky problems or to find coding resources you suspect may exist, but aren't sure where exactly to locate them.

> **Tips:** While the simplest searches on Google are fairly straightforward, to become an exceptional web developer you should take the time to learn the advanced techniques of Google search to provide far better results. Googleguide.com has a series of particularly thorough and useful guides to improve your Google search expertise.

Section 4

Common Misperceptions About Web Development

Like any group or occupation, web developers and the role web development is susceptible to any number of misperceptions and stereotypes. While these labels are not always harmful nor even applied with ill-intention, it is important that both the general population as a whole as well as individuals interested in a web development career understand the facts about the occupation and the environment that surrounds it.

For those not currently in the industry who wish to get a better understanding of what exactly web development and the people who perform it are like, we'll explore a handful of the most prevalent misperceptions and set them straight.

Misperception: *Web Developers Only Know Code*

The implication here is that web development doesn't require a knowledge or capability to work in other disciplines related to the field such as graphics, design, color palettes, database administration, content management, search engine optimization, and the like. Often, web developers are thought of as purely single-minded creatures that can (and do) only focus on programming without any familiarity with these other secondary practices.

Typically a web developer that only knows code is going to be pigeonholed into employment opportunities, whereas a **full-stack** developer that is trained in and capable of managing every aspect of the design and development process is offered far more in both work opportunities and compensation over a standard web developer salary.

Misperception: *Web Developers are Only Men*

It is true that the percentage of positions in the information technology field held by women over the past few decades had been in decline for a variety of reasons. A report published by the Information Technology Association of America in 2007 shows that the percentage of IT positions held by women had declined from 32.4% in 1996 to 24.9% in 2004.

However, the perception that there are no women in the development field, or that this downward trend continued, is not the case. According to the National Center for Women & Information Technology, there has been a slight increase in the past decade, with women currently holding 26% of all professional computing positions in the U.S. during 2014.

Misperception: *Web Development is Exclusively a Solo Activity*

While it is true that approximately 25 percent of all web developers are self-employed, it is therefore the case that the remaining 75 percent are employed by a development firm or a corporate entity. This brings with it the normal opportunities for social interaction within the work environment, whether we love them or hate them.

More importantly, by the very nature of the task at hand, web development simply *cannot* be an exclusively solo activity, because the end result of development efforts (typically a completed website) are created for and thus dictated by the needs of an employer or client.

A successful web developer will need to be adept at communication with other developers, designers, clients, and managers to accomplish the day-to-day tasks necessary to complete a finished product.

Misperception: *Web Developers are Unnecessary*

With the surge of powerful content creation and publishing tools now available to the general public such as WordPress and Medium, the incorrect conclusion is that the existence of these tools invalidates the need for web developers as a whole.

The reality couldn't be further from the truth. While a handful of individuals would have you believe that web design is dead, in truth what is actually dying off isn't the need for web design, but instead the generic solutions applied all too commonly: an over abundance of themes, skins, templates, and trends.

> *Strong developers are those that focus primarily on personality and product to make an astounding website.*

Misperception: *Web Development Requires a College Degree*

Like most technology occupations, web developers are hired and praised for their skill and expertise rather than accreditations. Development is a prime example where "the proof is in the pudding" – if a web developer shows he or she has the skills to do the work, little else matters to most employers or clients.

According to the U.S. Bureau of Labor Statistics, the standard entry-level education for most people entering the field of web development possess no better than an associate's degree.

Taken even further, what is most vital is the opinions of development firms and hiring managers that are seeking new, talented developers. The vast majority may not require a degree of any kind when hiring. As Mike Feineman, a developer at the successful social media agency Room 214, puts it,

> *"Self-taught programmers generally have a better drive, and are passionate learners. In other words, exactly the kind of people I want on my team."*

Numerous coding boot camps are popping up that offer a focused and streamlined education to learn the ropes and get started with a web development career. Once an individual knows how to make something functional, the expertise and portfolio begins to grow and most employers or clients will require little else beyond that proof.

Section 5

Determining Your Discipline: Front-End, Back-End, or Full Stack Development

As the field of web development continues to grow and expand, with the ever-increasing breadth of technology and tools utilized for building state of the art websites, in recent years the terminology to define a particular occupation or discipline within web development has evolved. As the needs of companies and clients have changed, so to have the labels placed upon, and ultimate focus of, the developers creating these final products of the web.

The Times They Are A Changin'

In the early years of the web, the term **web designer** was most commonly used for the individuals created or working on websites. This label was applicable primarily because web pages were so simple and largely static in content. Thanks to archival tools like Archive.org, we can review what AOL.com – one of the most popular websites in the world at the time – looked like back in 1996.

Image: AOL, courtesy of Archive.org

For those that were using the web back then, we might remember this to be a fairly modern and common style of web design: sidebar navigation with primarily static content including a few hyperlinks mixed in.

Therefore, it comes as no surprise that the first web developers were labeled as web designers because early website creation was largely all front-end code; structuring HTML tables and other tags, along with images, to create a functional if nondynamic site.

As technologies emerged and dynamic content was necessary for most websites, the need for a back end powered by a database arose, typically using database engines like SQL. This is also the period when the term web designer began to shift toward **web developer**, as it was now necessary to do more than simply edit front-end HTML, but to also code dynamic back ends that could query the database and provide relevant information to the user.

The broad term of web developer carried on through the early 2000s until around 2007, when use began to decline. The cause of this drop-off? Modern websites were becoming complex enough that more specialized disciplines were necessary, as only the most well-trained and experienced web developers could manage to do everything that developers of yore could do.

Within the last decade we've seen the emergence of the discipline-specific terms: first **front-end developer**, then **back-end developer**, and only most recently, **full stack developer**.

Front-End Development

While modern front-end development is far more complex and technical than it was back in 1996, the core ideas remain the same: a front-end developer primarily focuses on the **look and feel** of the website. This typically involves coding in three primary languages: HTML5, CSS3, and JavaScript.

Virtually everything to do with front-end development and coding involves altering the *client-side* view of the website: what the user will see and interact with.

HTML (HyperText Markup Language) builds the structure of the page through a variety of self-enclosed "tags" that define the elements of the page and where they should be placed. Every modern browser will then *interpret* these structured HTML tags into a rendered output of what the page should look like.

```html
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Determining Your Discipline</title>
    <meta name="description" content="Determining Your Discipline">
    <meta name="author" content="CodingDojo">
    <link rel="stylesheet" href="css/styles.css?v=1.0">
  </head>

  <body>
    <p>Hello world!</p>
  </body>
</html>
```

Image: HTML5 example

CSS (Cascading Style Sheets) are typically separate documents, which, as the name implies, are in charge of **styling** the page. This is performed by defining *selectors*, which are text tags that match a particular set of HTML tags from the generated HTML of the document. When any matching HTML tags are found, the CSS tells the browser how to style that particular chunk of the page. In the example below, we're altering the look of all the primary headers (*h1*) and all standard paragraph text (*p*).

```css
h1 {
  font-family: Arial, Helvetica, sans-serif;
  font-size: 2em;
  font-weight: bold;
}

p {
  color: #000;
  font-family: "Times New Roman", Times, serif;
}
```

Image: CSS3 example

JavaScript is a massive beast unto itself that can perform almost limitless tasks. For most front-end development uses, JavaScript is a *scripting* language that is purely *client-side*: Once the user's browser loads the page, the JavaScript code then executes and performs the functions it is intended to. These potential functions are far too numerous to list, but effectively JavaScript can alter virtually every aspect of HTML that has been rendered and displayed for the user, however front-end developer desires. For example, a common task is to cause an action when a particular event occurs, such as the user clicking inside a contact form to enter his or her email address. Below is an example of causing (an admittedly annoying) pop-up alert to appear when the 'email' field is clicked.

```
1
2  document.getElementById('email').addEventListener('click', function(){ alert('Email clicked!'); });
3
```

Image: JavaScript example

Best suited for: Individuals with a keen eye for style, interface design, layout, art, color palettes, and structure. Front-end developers are also very likely to interact with non-technical individuals during projects and must be adept at communicating across that knowledge gap about how designer or client desires will manifest into the look and feel of the site.

Back-End Development

Whereas front-end development is all about the visual style and user interaction of a site, the back end is where the dynamic content and core logic of the site is performed. Put simply, *the back-end code provides the data and information that is then utilized by the front-end code to produce the expected webpage.*

Unlike front-end code, which will generally be comprised using just the three standard aforementioned languages, the back end can be coded using any of a *multitude* of server-side languages or frameworks. Just a handful of currently popular examples include PHP, Ruby on Rails, Python, and Node.js. A back-end developer must also typically be adept at understanding databases and how to *query* (retrieve data from) a common database engine such as MySQL, PostgreSQL, and Oracle to name a few. With the appropriate data collected, based on the user request or the function being performed, that data can be inserted into the front-end code and rendered to the user.

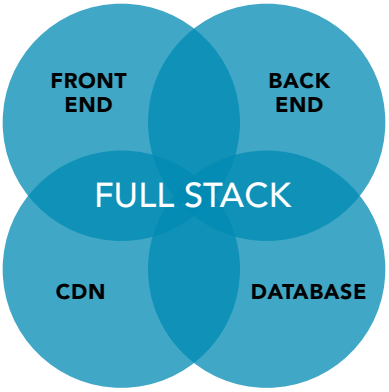
For example, below we see some code from Ruby on Rails for when a user visits the *show* user URL (i.e. http://domain.com/users/1). The back-end code finds the user in the database and that retrieved user record will be used elsewhere to display some user information such as name or email.

```ruby
class UsersController < ApplicationController

  def show
    @user = User.find(params[:id])
  end

  private
    def user_params
      params.require(:user).permit(:name, :email, :password, :password_confirmation)
    end
end
```

Image: Ruby on Rails example

Best suited for: Individuals with very logical minds and methodical practices. Generally far more "code-centric" than front-end development, back-end developers will write the core business logic of every application, defining how each piece or module of the site fits together. Those who enjoy coding, solving puzzles, and have honed their attention to detail are well suited for back-end development.

Full Stack Development



In late 2010 the term full stack developer began to gain popularity in large part due to the development practices at the massive social platform, Facebook. Since then, the concept of a full stack developer has gained such traction that has become synonymous with "the best of the best" – developers who are multi-disciplined and able to tackle nearly every aspect of website development.

A full stack developer is one who combines the ability to code both the front end and back end of a site, from HTML and CSS to server-side business logic and database integration. Full stack development may also include integration of outside third-party tools and resources, such as a content delivery network (CDN) which typically provides images, video, and other "heavy" content for a web application.

> **Best suited for:** Individuals with a desire to learn and to be among the most valued developers in the industry or on their respective teams. Full stack developers are logical, creative, and technically savvy; able to work with (or willingness to learn) a multitude of technologies and languages. Strong research skills are a must have due to the numerous tools that a full stack developer will engage with on a day-to-day basis.

Section 6

Getting Your Feet Wet: Learning the Building Blocks of Web Development

Perhaps one of the most exciting prospects about entering a field like web development is the ability to simply **get into it** right away. Unlike many professions that require months or even years of training, with web development you can easily get started *today* if you feel the fire of learning and discovery smoldering under your feet!

The best place begin getting a feel for how development works are the numerous free, self-guided resources available around the web that cover everything from basic programming introductions to development best practices. Below we'll take a look at the cream of the crop web development resources and highlight what they can each offer you and a selection of the best links or tools to check out.

MIT OpenCourseware (Visit)

An abundance of self-guided courses from none other than one of the best technical schools in the United States, MIT. The MIT OpenCourseware program offers an incredible breadth of topics to learn about, including hundreds of courses relating to programming, development, mathematics, and computer engineering.

> **Don't miss...** Introduction to Computer Science and Programming which is a full, independent study course including all resources and course materials to get you excited about and interested in programming. A Gentle Introduction to Programming Using Python is also a great resource to get insight into that popular and well-designed back-end language.

Khan Academy (Visit)

An extremely useful learning site that covers *all* manner of subject, and the computer programming section of Khan Academy in particular cannot be overlooked. It features a variety of self-guided tutorials, generally with experts providing audio and/or video guidance on the topic while interactive on-screen windows show the code and output the results during narration.

> **Don't miss...** The great introduction to the SQL database engine includes guided narration and even personal challenges that ask you to write SQL yourself to perform simple tasks. A fun series for learning the basics of drawing and animation using JavaScript is great for getting into that prolific front-end language.

Codecademy (Visit)

Codecademy provides a series of self-guided tutorials for beginners to learn the basics of web development programming. An in-browser, self-contained development environment is created where you can learn the basic structures of front-end code like HTML and CSS, before moving on to back-end languages such as Ruby on Rails and Python.

> **Don't miss...** The Make a Website and HTML & CSS programs are great beginning points if you need to learn the basics of web structure and design. For heavier coding, try the Ruby language tutorial followed by the Learn Ruby on Rails guide for making a basic, functional website.

Coursera (Visit)

Similar to MIT OpenCourseware, Coursera offers a plethora of online courses from a variety of universities around the world for free. Each course varies slightly in format and timeline, but numerous courses are available for programming, development, and computer science to get a taste.

> **Don't miss...** A great beginners guide to programming can be found in the Programming for Everyone course from University of Michigan. While the language used throughout the course is specific to Python, the course itself is focused on the concepts of general programming that can be applied to virtually **all** languages you'll encounter for years to come.

HTML5 Rocks (Visit)

While the interface is a bit lackluster and finding relevant articles can be somewhat challenging, the HTML5 Rocks site – that was created by Google – provides a wide assortment of articles and tutorials on all manner of web development topics, with intended audiences ranging from beginners to advanced developers.

> **Don't miss...** Getting Started with CSS Shapes provides a great introduction into advanced css techniques that many beginners may not even realize can be accomplished purely with CSS.

A List Apart (Visit)

One of the most professional and up-to-date online magazines directly aimed at web developers and designers, A List Apart is home to a multitude of exceptional articles dealing with everything from coding and techniques to design and user experience. If you want to do some light reading and learn from the experience and advice of other experts in the field, browsing through the articles here is a great resource.

> **Don't miss...** Building Nonlinear Narratives for the Web offers great insight into the notion that the scattered, modular nature of modern websites requires that narratives about our content are allowed to be free-flowing and not follow the traditional "beginning, middle, end" structure of storytelling. Also check out Reframing Accessibility for the Web, which attacks some of our own inherent prejudices about web users with disabilities and how developers can move forward designing for accessibility, regardless of the user at the other end.

Mozilla Developer Network (Visit)

Mozilla, the team behind the popular Firefox web browser, have created an incredible resource for developers of all skills levels and expertise through the Mozilla Developer Network. These resources, articles, and tutorials are perfect for those who absorb information and learn best using the tried-and-true method of reading words and seeing examples right there on the page. The range of topics is wide, from basic web introductions and front-end languages to common vocabulary and optimization & performance.

> **Don't miss...** Getting started with the Web is a great resource for beginners to learn about how websites function, and the guide then moves onto writing and playing around with front-end technologies such as HTML, CSS, and JavaScript.

Conclusion

Together in this book we began by examining what web development actually is and what you might expect to be doing on a day-to-day basis with a career in web development. Understanding the basic network communications between a user's computer and the server computer that houses the website, we could then better understand where web development ties into that process and how you yourself might soon be able to create a functional site accessible by millions of people online.

We also looked closely in Chapter 2 at the astounding growth and employment opportunities in the web development field, from a 20% increase in hiring opportunities over the next seven years to the exceptional starting salaries and limited education requirements to get into the industry. We also explored the freedom available to web developers with regards to workplace locale and even self-employment opportunities for the ultimate freedom employment and self-reliance.

In Chapter 3 we examined the critical qualities and attributes necessary to be a successful developer, from technical savvy and creativity to being a logical problem-solver and becoming adept at research and information gathering. For those feeling slightly inadequate to any extent, we also offered potential methods to improve your skills and increase your competence in each area.

As with any profession, web development and the individuals who work in the field are not immune to misperceptions and misinformation, so we looked at clearing up a number of the most popular misperceptions in Chapter 4. We learned about just how abundant the need for web developers truly is in the current marketplace and how traditional rules of requiring a four-year college degree for hiring eligibility simply do not apply to the web development field.

In Chapter 5 we examined the three primary disciplines most web developers will consider when making a career move: front-end, back-end, and full stack development. We explored each in detail including small code snippets and examples of how the development needs for the industry have evolved over the past few decades to a more nuanced and specialized series of roles for developers.

Finally, in the last chapter we explored some incredible websites and resources around the web for getting your feet wet in web development, from interactive coding tutorials through Codecademy and Khan Academy to articles from professionals at A List Apart and full, free university courses from MIT OpenCourseware.

Web development as a career choice has never been a more promising, exciting, and *actually possible* than it is right now. Your potential for success in the field of web development is only limited by your willingness to grow and learn. The industry is **full** of success stories of regular individuals just like you who, with little to no education or training, took that first step down the road to a career in web development and are now successful developers with a wide portfolio and great salary to show for it.

Start your future now with Coding Dojo's 14-week full stack development program, which offers a rapid coding bootcamp to train you in a variety of full stack technologies of your choice including LAMP, MEAN, Ruby on Rails, Python, and even Swift for iOS app development. These astounding programs pack in everything a beginner will need to get started with a web development career, including both front- and back-end technologies, providing you with the tools and knowledge necessary to label yourself as a true **full stack developer**.

With Coding Dojo's proven track record of **92% job placement** for black belt certified students within 60 days of graduation, now is the time to apply and discover what an amazing career opportunity awaits you in the incredible field of web development, with the help of the coding ninjas at Coding Dojo!