Create Your Personalized Coding Game Plan

A Free Guide With

7 *Proven* Tips to Kick-Start Your Coding Skills

From **Alex Coleman** of

Your First Web Development

yourfirst.io

When you're starting out, learning to code can be really overwhelming.

People throw all kinds of technical information and resources at you, but no one ever takes the time to help you develop a good plan of attack in order to achieve your goals.

And that's exactly what this guide is aimed to help you with. It contains proven tips based on my years and years of experience as a professional coder and code instructor. I still use these same exact tips and techniques every single day. They're simple, but extremely effective.

Worksheets are included throughout to help you easily put the tips into practice, right away. (Individual worksheet download links are included within the text, and all of the worksheets are also attached to the end of this guide). Again, these aren't philosophical musings. This is tried and true advice that you can use, immediately.

Ready to jump-start your coding skills? Follow these tips and you'll be well on your way.

1. Develop a Plan of Attack

Does the following sound familiar?

"Alright, I'm going to learn to code. Let's see, where to start ... *:: browsing the boatloads of materials on the web ::* ... HTML! That sounds like the right place to start."

There are *so* many resources available these days that — after you've decided you want to learn to code — one of the most difficult things is just determining what path to take.

But it's absolutely *crucial* to develop a good plan of attack. It will help you stay on track on ensure everything you're learning is getting you closer to your ultimate end-goal.

With that in mind, you need to ask yourself: **what are you ultimately trying to learn?**

- To build web applications?
- To build simple websites?
- To fix/enhance your existing WordPress site?

I very, very strongly favor resources that serve a practical end-goal. What do I mean by that? I like courses (or books, or whatever) that help you build something *real*.

My idea of a (largely) **unhelpful** course:

"Learn the basics of PHP"

My idea of a **helpful** course:

"Learn to build your own web application using PHP" (Pro Tip: I designed and built this very course.)

The former just teaches you a few "skills", without helping you put them into practice to actually *do* something with them.

Whereas the latter not only teaches you skills, but how to use them to ***create something real***. From my experience — and the experience of a great number of my students — resources like this lead to a much more

engaging, much more fulfilling experience.

After completing the “Learn the Basics of PHP” course, you may know “what a loop is in PHP,” but after completing the second, you know “how to build a complete application.” And that translates to building your very *own* applications — a *much* more valuable real-life skill-set.

Use the **“Discover Your Coding Path”** Worksheet to determine your coding Path and find the course best-suited to your goals:

Download the “Discover Your Coding Path” Worksheet

2. Create a Schedule

This is simple. (But extremely important. *Don’t skip this!*)

Schedule a **predictably recurring time slot** for your coding learning and practice. **For example, every Monday, Wednesday, Thursday at 7:00pm, after dinner**.

Here are some tips to make your schedule as effective as possible:

- Shoot for **three sessions a week**.
- Start with a **small, easy-to-complete amount of time** (e.g. 10 minutes).
- If at all possible, do each session at the **same time of day** (e.g. do *every* session from 7:00 - 7:10pm).

Starting with such a small barrier — e.g. a 10-minute session — makes it almost impossible for you to fail. Can you do one thing for 10 minutes? For sure.

As you progress, you can gradually increase the length of each session. (In fact, you'll likely see it happen naturally, as you get more into it and more excited.) But starting out with a small commitment helps develop a strong habit, which is crucial for your continued success.

Don't worry — the next worksheet (coming up in a minute) will help you establish, and follow, a set schedule.

3. Set yourself up for success!

When you're trying to accomplish something, there are always outside influences acting against you, as I'm sure you're well aware.

Some are unpredictable: if the basement starts flooding as soon as you sit down to practice your coding, you couldn't have done much to prevent that.

But, on the other hand, some outside influences are *predictable*. And *those* you *can* prepare for.

Basically, you want **remove (predictable) barriers ahead of time**:

- Set up your workspace beforehand.
- Remove distractions (e.g. leave your phone in the other room).
- Work in a closed-off environment (e.g. a separate room; a coffee shop) — one without distractions.
- Close all unrelated windows on your computer (more on this in a minute).

When your scheduled time arrives, don't think — just *go*. Go to the place that you have already set up for your success.

4. Use “Full-Screen” Mode

As I mentioned, it’s important to close all windows on your desktop unrelated to coding before you start each session.

To extend that, whenever it’s possible, if you only need to use *one* program — say, your source code editor — set it to full-screen mode.

That will help block out *any* possible distractions and will drastically increase your productivity.

But, with that said, I realize there are times when you must view more than one window. For example, if you’re following along with a course, you may require two windows/programs to be open:

1. A browser window, to view the course content
2. Your source code editor, to do your coding

So I’ll revise the rule to:

Display as few windows as possible, and have them take up the entire screen.

You should still simulate a sort of revised full-screen mode with those two windows: resize window #1 so it takes up the entire left-hand side of your screen; and resize window #2 to take up the entire right-hand side.

That way, you can see both windows, and distractions are still minimized.

Pro Tip: *If you have a tablet, consider setting it up next to your computer with the course you’re following along with and open your source code editor in full-screen mode on your computer.*

Use the **"Set Yourself Up For Coding Success"** Worksheet to set your coding session schedule and prepare your Zen working environment:

Download the "Set Yourself Up For Coding Success" Worksheet

5. Focus on Making Small, Consistent Progress

Let's face it: it's so easy to get wrapped up in the overwhelming amount of coding-related information available to you.

One minute you're working on an exercise and the next minute, out of the blue, you suddenly feel utterly lost. If you haven't experienced that yet, you will. I don't mean this to sound scary — it's just part of the learning process when tackling *any* new skill.

BUT, I have good news: there *are* way to greatly minimize those overwhelming feelings and inevitable roadblocks. And one of the best ways to do this is to use one of the tips discussed earlier: **focus on making small, consistent progress**.

That's exactly why I recommended setting the length of your initial coding sessions at just 10 minutes each. Is it easy to do something for just 10 minutes? Yup. There's really no excuse for not doing something for just 10 minutes.

And by starting at such a small, achievable amount of time, you will rapidly develop a very strong habit of *actually completing* those sessions.

With that habit in place, you can gradually progress to longer sessions and work into a schedule that suits you. But that's only possible after you've laid a good foundation of consistently completing those small sessions.

If you stick to this general philosophy, you'll be in great shape:

Start small ->
Build good habits ->
See progress ->
Gradually revise your game plan

Use the **"Track Your Coding Sessions"** Worksheet to keep track of all of your completed coding sessions to ensure you make consistent progress:

Download the "Track Your Coding Sessions" Worksheet

6. Take Breaks!

Here's the thing: coding can be tough. And inevitably, there will be points where you get stuck. Again, this is just a fact of life when you're working on *anything* of a complex nature.

But guess what? Because this is such a common occurrence in life, there's a tried and true tactic that almost always works:

When you get stuck, *take a break*.

Take a walk around the block. Go in the other room and read a few pages of a book.

I can't count the number of times I've gotten stuck and taken a short break, only to have the answer pop right into my head, mid-walk-around-the-block.

Your brain is a muscle. It gets fatigued. It needs breaks. When it gets tired, it may *feel* different than when, say, your legs get tired from a several-mile run, but nevertheless, it still experiences that same sort of exhaustion.

Knowing when to stop and give your brain a rest is crucial, not only to get "unstumped," but also to prevent yourself from getting overly frustrated. When you're coding, take breaks, often.

Use the **"Reflect On Your Coding Session Breaks"** Worksheet to monitor the breaks you take:

Download the "Reflect On Your Coding Session Breaks" Worksheet

7. Above All, Have Fun

I truly believe learning to code should, first and foremost, be *fun*.

It's incredibly rewarding to be able to start with an idea, and through coding, make it come to life right before your eyes, seemingly from thin air. It's still something, to this day, that brings me incredible joy.

So, throughout the entire process, always remember to enjoy yourself, and have some freaking fun with everything! Don't take yourself too seriously. Enjoy the process. Revel in the results.

It's a great journey, and, if you stick with it, you'll be immensely rewarded.

Get To It

This advice is based on my years and years of experience.

I've experimented with a lot, and these are the tips, tricks, and techniques that continue to work best for me, personally, time and time again. I hope that they'll save you some time and help jump-start your coding learning.

But I can't do the coding for you. So with that in mind, I'll leave you with a quote from one of the best teachers of all time:

> *"Do. Or do not. There is no try."*
>
> -Yoda

The rest is up to you...

Worksheets

Discover Your Coding Path

This worksheet will help you determine the coding "Path" to take based on the goals you plan to achieve. After you determine your Path, you'll see a list of recommended resources and learning materials catered to that Path. Remember, I always recommend using courses that help you create something real.

Below you'll find information on different possible learn-to-code Paths you can take. **Read through each, and determine which Path is best for you, based on what you ultimately plan to achieve.** Then come back and fill in your chosen path below.

My chosen Coding Path is:

_____

PATH #1

Create an application based on your own ideas

This is the path to take if you want to learn to build your own applications. Do you have an idea for an application that you want to bring to life? This is the path for you.

Next, **choose the Sub-Path** that matches your goals:

SUB-PATH A

Build web applications you can access via a browser

Do you want to learn to build web applications that anyone can access via a web browser? This is my personal wheelhouse. In my opinion, these skills are the most diverse and powerful, and enable you to get the most "bang for your buck."

Recommended Courses:

1. **Your First Laravel App** *(recommended)*
   The online course I created, containing easy-to-understand lessons that guide you through the creation of a fully-featured Laravel web application, teaching you all of the fundamentals of web development along the way. You all get free on-demand assistance from me at any time.
2. One Month Rails
   A video-based course that walks you through the creation of a basic Ruby on Rails application.

SUB-PATH B

Build mobile applications for smartphones

Do you want to build native iOS or Android mobile applications? Mobile development is booming, but, fair warning, it can be a tough market to find space in. With that said, it will only continue to become more and more prolific as the years go by.

Recommended Courses:

1. **Treehouse — iOS Swift Courses** *(recommended)*
   Treehouse, the popular web development course platform, offers multiple courses (e.g. "Build a Weather App") that walk you through the creation and deployment of complete iOS Swift applications.
2. **Treehouse — Android Courses** *(recommended)*
   Treehouse also offers those same Swift courses in Android flavors.
3. The Complete iOS8 and Swift Course
   This highly regarded udemy course guides you through all of the fundamentals of the Swift language and walks you through the creation of 15 real applications along the way.

4. The Complete Android Lollipop Development Course
   Another very highly rated udemy course that walks you through the creation of 14 real-world Android applications.

PATH #2

Learn to work on, or make changes to, an existing web application

See Path #1.

In order to understand and work with an existing web application, you'll want to learn how web applications work in general and how to build them on your own.

One thing: I'd recommend following a course that involves the same programming language(s) as your existing application. (E.g. If you'll be working with an existing PHP application, do a course on Laravel; if you'll be working on an existing Rails application, follow a course on Rails.)

PATH #3

Create a simple website (e.g. an "About Me" page, or a simple business website)

The core languages typically required to create simple websites are HTML, CSS, and JavaScript. If you're creating a website from scratch, I highly recommended using the WordPress platform.

Recommended Courses:

1. **Treehouse — HTML, CSS, JavaScript, and WordPress Courses**
   (recommended)
   The folks over at Treehouse offer a ton of courses on building simple websites and on WordPress. Browse their course library to see everything they have to offer.

PATH #4

Fix, or enhance, your existing website

It depends on the technologies your existing website is built with, but this Path is **likely very similar to Path #3**. Simpler websites are all built with a combination of HTML, CSS, and JavaScript. If you're looking to work with an existing web application, see Paths #1 and #2.

Recommended Resources and Materials:

1. **Treehouse — HTML, CSS, JavaScript, and WordPress Courses**
   (recommended)
   The folks over at Treehouse offer a ton of courses on building simple websites and on WordPress. Browse their course library to see everything they have to offer.

If you feel like you need more assistance

… to develop your plan of attack, send me an email. I'm happy to help you develop a good, solid plan.

Set Yourself Up For Coding Success

Use this worksheet to set yourself up for success. Determine where and when, specifically, you'll do your coding sessions, and make sure you prepare your environment to be absolutely distraction-free.

1. Choose your a distraction-free coding environment:

_____

2. The days of the week will you consistently commit to coding:

[ ] M [ ] Tu [ ] W [ ] Th [ ] F [ ] Sa [ ] Su

3. The *specific* time slot for your coding sessions (e.g. 7:00 - 7:30pm)**:**

_____

4. Do the following before you start every coding session (add additional items, specific to you, to the end of the list, and *keep this list handy!*):

- put cellphone out of sight (or *at least* on airplane/distraction-free mode)
- close all unnecessary windows on your computer
- make a list of necessary programs/windows you'll need for your session
- arrange the windows in "full-screen" mode

Add any additional, personalized things to do before each session here:

_____

_____

Track Your Coding Sessions

Use this worksheet to track your completed coding sessions. Sticking to a consistent schedule will help you make consistent progress. Multiple forms are included; you can print out more, as needed.

***At the start of each week, fill in the dates for Monday and Sunday (above the form), and then fill in all of the "time slot" values according to your coding session schedule**, established in the "Set Yourself Up For Coding Success" Worksheet.*

An example of a completed week's worksheet is shown below.

EXAMPLE:

Week of **Monday,** _January 12_ through **Sunday,** _January 18_

Coding session time slot (e.g. Mon 7:00-7:10pm)	**Coding session completed?** (Y or N)
Mon 7:15 - 7:30pm	Y
Wed 7:15 - 7:30pm	Y
Fri 7:15 - 7:30pm	Y

Your Coding Sessions

Week of **Monday,** _____ through **Sunday,** _____

Coding session time slot (e.g. Mon 7:00-7:10pm)	**Coding session completed?** (Y or N)

Week of **Monday,** _____ through **Sunday,** _____

Coding session time slot (e.g. Mon 7:00-7:10pm)	**Coding session completed?** (Y or N)

Reflect On Your Coding Session Breaks

*Use this worksheet to track the breaks you take during coding sessions. It will help you determine when you're getting stuck and, over time, the most effective way for you to take breaks. Multiple forms are included (as **you should be taking breaks fairly often**), and you can print out more, as needed.*

1. What I got stuck on:

2. What I did to take a break (e.g. "took a 10-minute walk around the block"):

3. How it helped (e.g. "Helped me relax and relieve some frustration"; or "I figured it out during my break!"):

---

1. What I got stuck on:

2. What I did to take a break (e.g. "took a 10-minute walk around the block"):

3. How it helped (e.g. "Helped me relax and relieve some frustration"; or "I figured it out during my break!"):