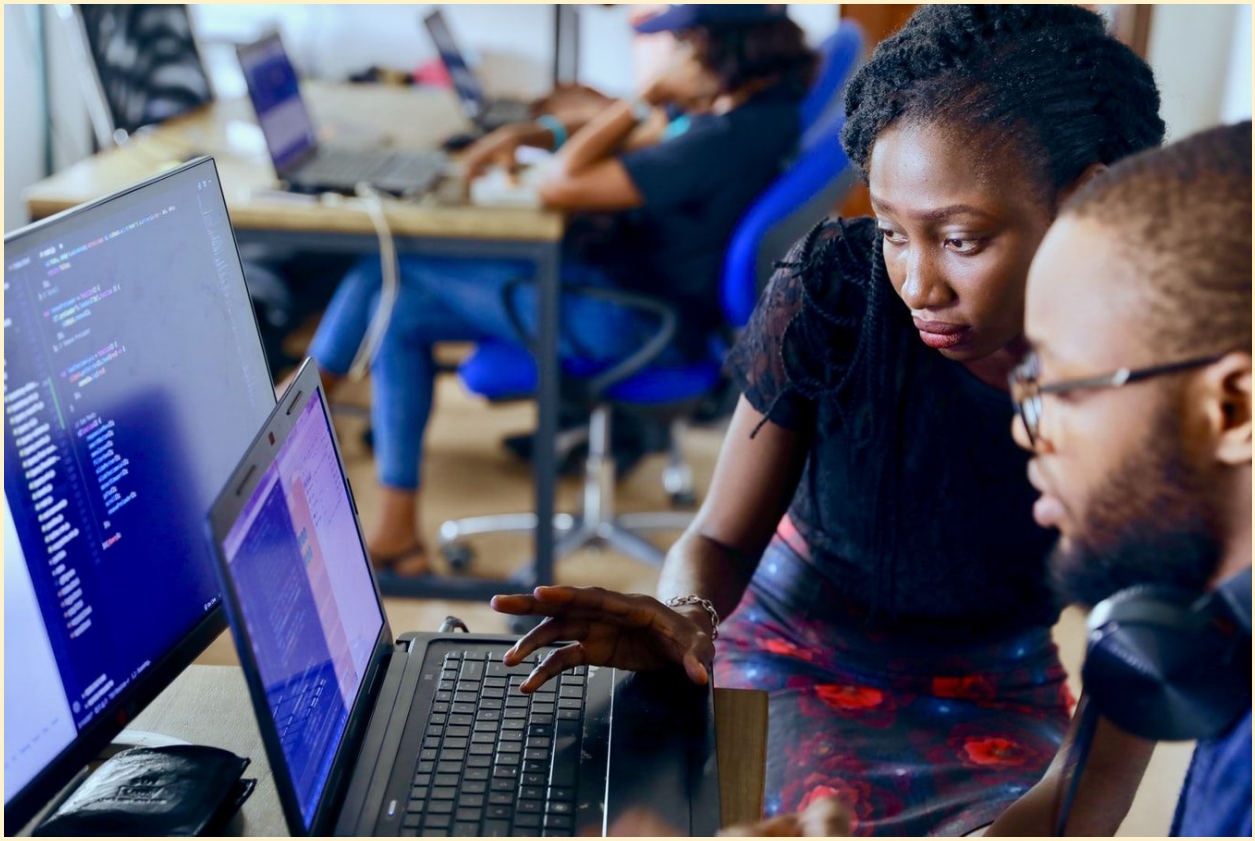
How to Become a Great Software Developer [with Actionable Tips]



Starting a new career in software development is both exciting and frightening. There is a lot to learn, and it takes years of practice to become really good at it, like with other crafts.

To make it a little bit easier, especially for beginners, I want to share some knowledge I've gained over the years in the form of actionable tips with you.

1) Find a great mentor -

One of the best things you can do to become a great developer is to find an excellent mentor. A good mentor will provide sound advice on everything from architecture to design, and even how to improve your career path.

A person who is not in the same company or location may be the best mentor for you. You can talk to them and get their advice on the phone, via email, or online. A mentor doesn't have to work at your company nor be an expert developer.

Find someone who understands software development and has had success in their career that you respect.

2) Learn to communicate well -

Being a great developer means that you can communicate effectively with users and other developers. In fact, a large part of being an excellent programmer is sharing your ideas effectively and efficiently.

If you can present your ideas clearly and concisely, others will want to listen to what you have to say.

But communication isn't just about talking. It's also about listening to what others have to say. Studies show that good listeners were more successful in their careers than good talkers.

So you should never underestimate the power of a good listener, especially when working with clients or other developers on your team.

Learning how to listen can be invaluable, especially if you're planning on working with other people.

3) Learn how to use Google effectively -

When I started as a software developer, I remember thinking that my fellow developers were some sort of super humans because they could always point me in the right direction when solving problems or implementing features.

That's when I started Googling everything, learning new tools and techniques along the way.

It turns out that knowing how to use Google effectively doesn't make you any less of a programmer – it makes you more effective at what you are doing.

Take the time to learn about all different kinds of tools so that if an issue comes up that requires something outside your realm of experience, you can quickly determine whether it is worth spending time learning about or not.

4) Start writing code every day -

As soon as you finish reading this article, write some code! It may sound obvious, but you've got to start writing code every day.

I know you're busy, and everyone else is too. Trust me – there's a reason we all keep telling you to write code every day. It's because it works.

Writing code every day keeps your skills sharp, helps you learn new languages and technologies, and allows you to work on stuff that interests you.

If it doesn't interest you, switch it up – change languages or projects at least once a month. This will keep things interesting (and stop them from getting boring) and make sure that you don't get stuck in a rut for months on end.

5) Don't be afraid to make mistakes -

No one gets this right on the first try, so don't feel bad when you mess up. Everyone else is still learning too! It is essential to learn from your mistakes so that you can avoid making them again in the future and so that you can help others learn from them as well.

In fact, if you make a mistake and then fix it before anyone notices, what have you learned? If someone points out that there was a problem with the code you wrote or tells you how to improve a particular design choice, thank them for spotting it. They are doing you a favor by pointing out something that needs improvement or clarification.

Even if they are wrong about what needs fixing, ask questions and discuss the issue with them to clarify why they think it is an issue. Remember that just because someone does not agree with your solution does not mean it's wrong – it just means they have different needs or expectations than you do.

The main takeaway here is: be open-minded and willing to learn!

6) Work on projects outside of work -

Being a great developer doesn't mean you know everything there is to know about programming and software development. Staying humble and willing to learn are both traits that great programmers have.

If you are interested in JavaScript I have collected some great **JavaScript projects** that you can add to your portfolio.

Working on side projects allows you to apply what you have been learning at work outside of work, which will help you learn even more. You can learn a lot from doing something wrong, but it's also essential to see how things are supposed to work. So try to learn from other people's solutions and prevent yourself from making mistakes when working on new projects.

It's essential to never stop learning or improving because the world of tech is always changing. There is no "set it and forget it" mentality when it comes to software development.

7) Follow industry leaders -

Follow industry leaders via blogs, social media, email newsletters, and so on, and try reading up on what they are writing about.

Lots of developers jump into new technologies right after they are released without really understanding what problems these new technologies solve or why they would ever need them – and this is a mistake!

Newer technologies are not always better than older ones. There is usually a good reason why specific tools and techniques exist in our respective industries. And if you

don't understand those reasons, chances are that you will fall behind your peers who understand the individual decisions that their organization or the industry as a whole are making.

Reading what influential developers are writing about helps put things into perspective so that we can avoid falling into the proverbial "black-hole of techno-babble".

While I wouldn't call myself an industry leader (yet), I would be humbled if you decided to follow me on **LinkedIn**. I'll try to regularly post content to support developers, especially beginners. :)

7) Get involved in open source projects -

Contributing back to the community that provides you with incredible tools and frameworks makes you appreciate them even more!

You can learn a lot from seeing other people's code (on GitHub), fixing bugs, writing documentation, and so on. Sometimes finding issues or making pull requests can lead to opportunities for mentorship or employment if others see your contributions.

A good way to get started is to search for GitHub projects with the tag "good first issue".

I've personally had many great experiences while working on open source projects:

I've learned about industry best practices from reviewing other peoples' code.
I've learned how to write better documentation because others were not sure how something worked.
I've received lots of positive feedback for my contributions, which helps make me feel good about myself as a developer, and
I've made many new friends in the process!

It's important to remember that just because we are using someone else's software for free does not mean we should take advantage of this fact by doing nothing but complaining about problems without offering solutions.

If we don't like something, we should try fixing it or offering ideas for improvement. Maybe your solution will be implemented by the project maintainers themselves.

8) Learn from peers and mentors – AND teach them, too -

While working with mentors is excellent, you don't have to wait until you have 5+ years of experience under your belt before becoming one yourself. We can all teach each other new things every day.

You should always be willing to help others out when they ask questions or need advice. Teaching others gives us a greater perspective on specific topics/situations that may otherwise seem foreign (or intimidating) if we are only observing via text/video alone.

For example, I once wrote a post about Java vs. JavaScript explaining the difference between the two programming languages.

As our peers progress through their career paths and become senior-level developers, they need to continue asking questions to continue learning once they reach their current knowledge level. By helping each other out in this way, we can all improve at an accelerated rate without ever being left behind due to missed opportunities for mentorship along the way.

9) Learn how to ask for help -

If there's one skill that will take you far as a programmer, it's learning how to ask questions. Don't ever be embarrassed or ashamed if you don't know something. Instead, use those moments as opportunities for growth and improvement.

Asking someone else for help shows them that you value their opinion and expertise. Most people will be flattered by such an offer, which makes them more willing to help out when they can.

10) Final Thoughts -

Don't forget that the best way to learn new things is to find something you're interested in – nothing will stick with you if it feels like an obligation or chore rather than a source of joy and reward. Learning should be fun, so seek out ways to make it more enjoyable.

And remember: don't be afraid of failure! The more ways we try at something, the more likely we are to succeed. You only fail when you give up on learning something. So get out there and start learning!