What are some things you wish you knew when you started programming?

Ken Mazaika, CTO, Co-founder & Mentor @ thefirehoseproject.com
Updated May 23, 2017 · Featured on Forbes · Upvoted by Oladeji Abubakar Adebowale, Mobile Application Developer, and Nupul Kukreja, Ph.D. Computer Science & Software Engineering, University of Southern California

I wish I knew a ton of stuff, but **27 things** come to mind.

Full disclosure: I put the most important things at **#14** and **#26** (depicted above) because I'd love for you to read the full answer. I think you'll enjoy it.

1. You learn by doing. The only way to get better at programming is to actually program. Don't let an analysis paralysis prevent you from ever starting.

2. Programming isn't like studying for a test. Memorizing stuff doesn't matter all that much.

3. Cheating is completely acceptable. I use Google to solve most of my problems, and so do most programmers.

4. Letting problems pile up without checking is a poor strategy. I used to make endless changes to my code and expect it to work right away. The problem with this approach is that it stacks one problem on top of another, and it becomes difficult to figure what went wrong.

5. Learning by yourself is really tough. Talking about code and collaborating with other aspiring developers is a better way to learn for most people.

6. Let go of your emotions. You're going to see error messages on your journey to becoming a programmer. When screwing up is such an integral part of the process, you need to be ok with making mistakes in order to grow.

7. You don't need 5 monitors. Contrary to what Hollywood might tell you, you don't need an insane external monitor setup to be a programmer. The computer that you own right now is probably good enough to use to start programming. Don't waste your money.

8. There's a BIG difference between a capital letter and a lowercase letter. It took me a while to get used to noticing subtle differences between similar symbols, and it can be pretty frustrating until you get used to it.

9. Trying to understand everything is a lost cause. In the beginning, I tried to chase down the "why" to every problem I encountered. This isn't necessary. Computers are so complex and there is so much to learn, and you're never going to understand everything. That's ok.

10. Pair program as much as possible. There is no faster way to learn how to code.

11. Changing bad code is part of the process. I used to think that every piece of code I wrote needed to be perfect. But making improvements to your code is normal. You're not writing a book that can't be changed once it's published.

12. There's a right way to ask for help. Everybody needs to ask for help at some point. And when you do, make sure that you do these 4 things:

- Overcommunicate details that you see.
- Explain exactly what you think should be happening.
- Explain exactly what is actually happening.
- Explain why you think it should be working differently.

When you go through this process, you'll often come across a solution without ever asking for help. It makes you think about problems in a holistic way.

13. You don't need to be a math genius. If you're not a "math person," that doesn't mean that you can't be a programmer.

14. Always celebrate the small wins. Building stuff with code is really cool. I never would have arrived where I'm at if I hadn't stepped back and admired the awesome things I was building along the way.

15. Meet-ups are incredibly valuable. In the beginning, it was pretty intimidating to get myself to the point where I felt comfortable attending these. But once I did, I realized that there were a ton of developers just like me.

16. Avoiding merge conflicts makes you happier. Merge conflicts are annoying. So when I realized that I could sneak a feature in before a teammate so that he/she had to deal with these instead of me, I was super excited.

17. It's ok to admit what you don't know. When you land your first programming job, you might be inclined to "fake it till you make it." Don't. Nobody expects you to know everything right away.

18. It doesn't take 10,000 hours to be good enough to land a job. In reality, you need to be good enough to know how to self correct and get back on track when problems arise. This takes far less time than 10,000 hours .

19. You're going to wake up in the morning thinking about code. And when it happens, it's really freakin' cool.

20. It's ok to make big mistakes. I once made a mistake that cost my company $10,000 dollars. In the process, I learned the most important lesson of my programming career.

21. An algorithm is like finding a name in a phonebook. Algorithms are a step-by-step approach where there is exactly one correct next step. An easier way to think about this is the strategy that you would take to find one specific name in a phonebook.

22. You're never going to feel like you're ready to program full time. Imposter syndrome is real. Try to remember that it's normal to not know everything. The most important thing is understanding that you can figure out the stuff that you don't know.

23. Programmers never stop learning. New technologies come out all the time, so the programmers that succeed are the ones who continue to learn and develop their craft on an ongoing basis.

24. Make the computer think like a human. Too many people have the impression that you need to think like a computer. It's actually the opposite.

25. Programming is about using the right tool for the job. There are so many different open source libraries, tools, and frameworks at your disposal. So you need to grow your developer toolkit and understand which tool makes sense for each problem that they encounter.

26. It's common to give up right before change is about to happen. Learning to code (especially to the point where you can switch careers) is a lot of work. It takes time and a lot of discipline, but it is possible. Too many people make the mistake of doing almost enough work to to get where they want .

27. Learning to code isn't easy. But that's why it's worth doing.

In hindsight, I'm actually really happy that I was so naive in the beginning. Knowing so little back then gave me the motivation to think critically about everything I learned later on.

Now I get to spend time helping other people achieve their goals through code. What could be better than that?

Too many beginner programmers worry about when they'll be ready to start programming full time. So I wrote about ***the one milestone that means you're ready to make the leap***

If you liked my answer, I'd really appreciate it if you upvoted it by pressing the light blue button below.

654.9k Views · View Upvoters

Upvote · 12.9k Share

Shawn William, senior software engineer
We hate to read others' crappy/spaghetti code :(...

View 100+ Other Answers to this Question

About the Author

Ken Mazaika
CTO @ Co-founder @ thefirehoseproject.com

1 Follower You Know
Ashutosh Gupta

Works at theFirehoseProject

Studied at Northeastern University

Lives in Boston

9.6m answer views
227.2k this month

Session Host
Jan 2017

Follow 9.1k Turn On Notifications

More Answers from Ken Mazaika View More

What is the hardest part about learning to program?
67.4k Views

Will coding ever be obsolete?
163.2k Views

What is the answer to a question you don't know in an interview?
524.4k Views

What are some traits/practices of experienced/good programmers that every beginner programmer should know to make them better at their craft? What would you as an experienced programmer have liked to have known when starting to program?

511.9k Views

What are the programming mistakes most inexperienced programmers make?

810.7k Views