How can I improve my coding skills?

cmw.osdude | *Apr 3 2013* | *Comment (1)* | *Visits (8414)*

I got the title of this article as a question through a private Facebook message. I decided that this is probably something that a lot of people are thinking about, so I figured I'd post my thoughts here. (I've also been very sparse at writing here so hopefully this will get me back on track.) My thoughts on this are not specific to code but about skill-building in general. I'll talk about code, but I'll also talk about other technical areas where I hear people say that they want to grow.

"How can I improve my coding skills?"

The short answer is simply "Write more code." That's not a very satisfying answer, though. My own immediate response to an answer like that would probably be something like "If I could do that then I wouldn't have asked the question." So, how do you get passed the obstacles? How do you grow in something where you feel you need improvement.

Surely the first step must be to understand what you think needs improving. What is the difference between what you *can* do and what you *want* to do? Do you have a goal? A role model? I don't think that you can get anywhere if you don't know where you want to go.

People act like setting goals is easy, but it's really not. Often our goals are vague or open-ended. In fact, "improve my programming skills" is both vague and open-ended! So how do we narrow that down?

What are goals?

Setting reasonable goals is one of the key secrets to achieving just about anything. You set a destination and you go there. Yet how do you tell the difference between an achievable goal and just a wish. I've repeatedly heard that the best way to set goals is to use the S.M.A.R.T. system. The letters stand for:

- **S** - Specific... Identify a particular thing that you want to do
- **M** - Measurable... Have some way of determining if you have accomplished your task
- **A** - Attainable... Do something that is within your resources
- **R** - Relevant... Make sure that it fits with what you want to achieve
- **T** - Time-bound... Give yourself a time limit to get it done

These aren't hard and fast rules, but they are good guidelines for keeping you directed. So, how would we apply this to "improving programming skills"?

Specific - What skills do you want to improve? Do you want to learn a particular language? Do you want to work with a particular kind of application? Your overall goal may have subgoals depending on your current skill set. It's a great goal to want to program on-line video games, but if you have no coding or network experience then you have some foundations to build. It is likely that you have a pyramid of goals that have to be achieved in order to reach your final destination.

Measurable - What are you going to set as your test for each goal? For programming, it makes sense that your goals would include writing specific functions and applications which build toward your overall goal.

Attainable - What resources do you have at your disposal? If your goal is to be a master mainframe programmer but you don't have any way to access a mainframe you are going to get frustrated. Start with things that you can do. The open source world has made this easier than ever to learn technologies without having to buy expensive hardware and software. Your circumstances might allow you to gain access to other resources by simply asking, or bartering. It's true that what you need for your goal may not be easy, and you shouldn't always go for the low-hanging fruit, but if it does not seem possible for you to do something then that's not really a goal. It's a desire waiting to become a goal.

Relevant - How will this activity help you accomplish your goal? This makes more sense for sub-goals in your pyramid. Will learning PHP really help you with embedded system programming? Will developing your drawing skills really help you create a phone app? Sometimes our goals will take us into strange places that really are useful in ways that we hadn't anticipated. Other times we distract ourselves by shiny things that just seem interesting at the time. It doesn't mean that you shouldn't be multifaceted with lots of varied interests, but your curiosity can drive you off-track. If you feel you are

About this blog

Chris Walden's thoughts and adventures as an open-source kind of guy. The ideas stated here are his own and don't necessarily represent the positions, strategies or opinions of IBM or its partners.

Related posts

The NASA Space App C...
UpdatedMay 21 0 0

Announcing a New 4-m...
UpdatedMay 12 1 0

Want to learn node.j...
UpdatedApr 15 0 0

SCCD 7.5 Using Black...
UpdatedApr 10 0 0

New youtube video - ...
UpdatedApr 9 0 0

Links

Clonezilla
Pidgin

Tags

Find a Tag

3d android articles austin blender blogging business community culture data desktop developerworks development droid education email encryption ffmpeg freedom future gimp google government gpg

struggling too much with your goal you may have a lot of irrelevant things that need to be set aside for a while.

Time-bound - When will I do this? I can attest that the things that I have to do with deadlines usually get done while the things "I mean to do some day" wait. Setting a time goal will help you keep on track. If you are like me, you curse the clock and the calendar a little and would rather play with ideas and technologies. However, that won't help you get it done. Set some time constraints on your goals and you'll be better off.

I think it's also important to remember that you ~~may~~ will fail in some of your goals as you move forward. You'll try something that is beyond your skills and have to back up a little. This is all part of the process. You *cannot* give up because of a failure here and there. Persistence and refactoring your path as you go is what will bring you success.

Where do I start?

As I said above, this is a great time to be in technology if you want to learn. More options are available less expensively than ever before. The Internet also provides a lot of ways to interact with others who share your interest to share resources and knowledge. Even if your interests are very specialized, odds are that you will find some group that shares them. This is important to deal with they nay-sayers who try to talk you out of your goals.

I would begin by finding something that was a model for my goal. For example, if I wanted to be a game programmer, I'd start looking at games that I liked and looking behind the scenes into the people who made them happen. You will likely find a blog or something where they have talked about what they think is important to their work. You'll also find various sites that recommend the sorts of skills that you need for a particular industry. Get a feel for what you need to learn and start putting together a picture of what you need to do. **Write this stuff down!** Don't just ponder it, but lay out written goals and plans for what you want to do. You'll forget pieces otherwise. This is a serious project. It's your future. Put together the same kind of material that you would to explain this to your boss, but do it for your own benefit.

Once you have an idea of what you want to achieve, then start building your subgoals. If you are doing video games you should probably learn something about C/C++. If you are focused on Mobile games then you might need to be more focused on Java. You might need to collaborate with people who have other skills, such as a graphic artist or musician. This may require to you become more social.

I would also recommend that you look at open-source projects that are relevant to what you want to learn. One of the nice things about these projects is they usually need work done and they are happy to have contributions. You don't have to build an entire application yourself, which can be hard, and you get to write code which will be reviewed by others. Even if your code isn't accepted as the final solution in the project, the experience you gain by trying is equal to what you would have doing exercises in a classroom. You also might make a real difference to the project with your contribution, which will be *highly motivating*. You can find repositories of open-source projects on sourceforge, github, ourproject.org and any number of other sites.

Also remember that many projects need contributions besides just coding. You might be able to contribute something where you are more skilled, such as editing documentation while you become socially connected with people who have the skills you want to learn. It's a lot easier for me to give my time to someone who is helping me do something I find difficult than someone who feels like my knowledge is theirs for the taking. Become a contributing part of a team in whatever way you can and resources will open up to you.

Not just programming

The question I addressed was improving programming skills, but I think these ideas are relevant to improving just about any technology skill... and probably non-technology skills as well. As our world has more and more chaos because there are more and more resources with more and more choices it is difficult to follow a simple proscribed path. An emerging skill is the ability to become our own teachers, to determine our own curriculum and to find our own resources. Technology is moving pretty fast and it takes a lot of effort to put together a series of classes and get it through the approval process for a University program. The ability to set your own goals and plan to achieve them will keep you agile, even when you have left the University far behind you.

Tags: *programming open_source opensource coding skills open-source education*

Add a Comment | More Actions

Comments (1)

Add a Comment | *More Actions*

1 kelleydg *commented Apr 9 2013* *Permalink*

Good insights, Chris. Thanks for sharing! Another idea would be to participate in contests, such as The Great Mind Challenge: https://www.ibm.com/developerworks/mydeveloperworks/groups/service/html/communityview?communityUuid=f870215a-82d8-4701-88a5-7937fb3c73c0 and the Master the Mainframe student contests: http://www.ibm.com/developerworks/university/students/contests/. You can also see who else in your school participates in these student contents, and you can get some advice from them about what has worked well.

Show: 10 | 20 | 30 items per page 1 Previous | Next

Previous Entry | Main | Next Entry