How to mentor a junior programmer [closed]

Asked 16 years, 4 months ago Modified 6 years, 4 months ago

Viewed 6k times

47

As it currently stands, this question is not a good fit for our Q&A format. We expect answers to be supported by facts, references, or expertise, but this question will likely solicit debate, arguments, polling, or extended discussion. If you feel that this question can be improved and possibly reopened, visit the help center for guidance.

Closed 12 years ago.

Does anyone have any suggestions on how to mentor a junior programmer ? If you have mentored someone did you follow any process or was it quite informal ?

If you've been mentored in the past what kind of things did you find most helpful ?

language-agnostic

Share

Improve this question

Follow

edited Aug 23, 2013 at 15:18

user1228

asked Aug 24, 2008 at 14:05

David

14.2k ● 25 ● 82 ● 102

9 most helpful: a rolled-up newspaper – Steven A. Lowe Dec 24, 2008 at 17:48

@Steven A. Lowe: You brought back so many memories from when I trained my dogs. :) – Esteban Araya Feb 1, 2010 at 2:41

12 Answers

Sorted by: Highest score (default)

48

Try to set aside between 30-60 minutes a day to review their code together. If you can't do this, then try to get together to review their code whenever they make a code commit, unless it was very basic. Have them explain why they chose the approach they took in lieu of others. A process like this helps to establish a great relationship, as well as really stimulate the student to think on their own and be able to defend their decisions. Not only does the student end up with someone approachable whom they can trust, but you'll notice an improvement in their quality of code and logic almost immediately.

Edit: Also, If you are unable to commit this much time to co-review with your junior, then you probably shouldn't be mentoring them and instead see if anyone else has a schedule that would allow it. The whole point of mentoring is to actively aid in the professional development of the

student, and they're not going to learn much if proper attention and guidance is not given to them.

Share Improve this answer Follow

edited Aug 24, 2008 at 14:33

answered Aug 24, 2008 at 14:14

Aaron

23.8k ● 10 ● 50 ● 48

18

I had the chance to work as an intern (one of two) in a small software company and had the opportunity to work on an "almost new" project they had. They had me set up with everything needed and gave me an introduction into what the project actually was (basic stuff like what the requirements were etc).

At first we got to do minor tasks like researching things that mattered to the project (they had given us a list of topics). This was, I think, to see how much we could handle ourselves, as the things we needed to look up and research were not that trivial and it took a good 2 weeks or so (counting the basic demos we had to create for it). That testing phase was actually done really without much 'coaching'.

However, after that period we could work on the actual project itself. This was also the moment we began to be coached together, in a similar style to pair programming, except there were three of us (2 interns and 1 'coach').

We learned a lot from him, but it was in an informal manner, and he didn't act like the 'all-knowing-listen-to-me' guy. When we had suggestions he would listen and think through with us whether they were any good. or give his view on why an idea should not be done in that way... Now that I think of it, he actively encouraged us to make suggestions, and to think about better ways to do things, instead of just sitting there 'taking orders' from someone who probably knows what to do better then you.

So in short:

- Let the junior programmer work (mostly) on his own to study the materials at hand, give him a list of minor TODO things like looking up information, or building small demos.
- Check the work he has done regularly and advise him if there are better ways to do things. Also point out the items he actually did well, that way he'll remember those for later.
- Let him work on a real project, and mentor him by working together at the same project, giving him advice when he has questions.
- The effort has to come from both directions: encourage him to ask questions, to challenge 'the way it is currently done'. Ask him questions on how he thinks it should be done and give him your opinion as well.
- Make it 'enjoyable' - don't let it look like you are giving orders.

Share Improve this answer

Follow

edited Mar 15, 2011 at 16:00

ire_and_curses

70k ● 23 ● 119 ● 141

answered Aug 24, 2008 at 14:37

sven

18.3k ● 10 ● 52 ● 62

> This is good. One of the problems I have observed in mentoring is that you have some people that will run with the ball, and some who won't. I am currently of the opinion that you need more ball runners to be successful. – EvilTeach Nov 30, 2009 at 17:21

14

During an internship w/ a large company that had a lot of in house IT, I was paired w/ a mentor. The practice definitely aided my career development, both in terms of technical skills and business skills. Here are some of the reasons the mentoring worked out so well:

- **Credible**: The mentor had 8+ years of experience and an accomplished background to draw upon in leading and training. He'd been through different challenges, worked in different environments, so he had a great perspective.
- **Genuine**: The mentorship was encouraged by the supervisor, but not so formal as to make it an exercise in going through the motions. The mentor wanted to mentor, and I wanted someone to learn from.

- **Passion**: The mentor loved the field he was in, the problems he was solving, and the technologies he was using. When I came under his wing, I found this to be infectious.
- **Sharp and Articulate**: The mentor approached issues critically and framed them concisely. There wasn't a lot of fuzziness in our discussions; we got to the root of the matter and he directed me on wise courses of problem solving and action.
- **Meaningful**: The work I was doing w/ the mentor was meaningful work, not just an exercise to keep busy or ramp up in a skill set. By jointly working on a task that tangibly aided the organization, that helped focus my interest and legitimize the mentoring process.

Share Improve this answer

Follow

answered Aug 24, 2008 at 20:48

user290

6

At my first place of employment there was this really **patient** dude that would always help me solve my immediate problem, and then teach me some important underlying principle. I loved this because he would help me stay productive while teaching me how to become a better programmer.

Share Improve this answer

Follow

answered Feb 1, 2010 at 2:42

Esteban Araya

29.6k ● 25 ● 111 ● 141

3

I'd be the junior, I guess :) I think I'd value an informal approach. It probably depends a lot on your and your mentee's characters, but I'd say that you learn best if you don't have egos in the way. Break the ice, make sure there's feedback in *both* directions. Things like code reviewing (both ways?) and occasional pair programming may work, and if there's a good match, it may be a lot of fun, as well!

Share Improve this answer

Follow

answered Aug 24, 2008 at 14:16

onnodb

5,261 ● 1 ● 33 ● 41

3

Because I had to explain *why* I wanted to co-op (besides needing the money) during my interview, my manager made sure my first project allowed me to work on what I had identified as weak areas: very little Linux experience (I chose a Linux-only R&D team so I would be forced to learn), not knowing a useful text editor (I really wanted to learn Vim), and how to learn another programming language (very different approach than learning a language as you learn to program). He told me I was being paid to study for a while.

I learn best by reading books, so after chuckling over *Unix for Dummies* (yay! I wasn't the only one who thought this was obscure and knuckleheaded sometimes) I started with *Unix in a Nutshell* and Sobell's *Practical*

Guide to Linux Commands. After that I printed out the Vim documentation and started going through it. Then I looked through a couple books on Python, the language of my first project. I was given all the time I needed to feel comfortable about these things (which was the real problem, as I now understand) and then began adding functionality to a previous co-op's project.

I realize now it would have been terrific to meet with someone every day or two for code review, as Kamikaze Mercenary said.

Share Improve this answer

Follow

edited Dec 24, 2008 at 17:50

answered Dec 24, 2008 at 17:45

kajaco

2,577 ● 3 ● 24 ● 34

3

In my experience, when mentoring someone, it is very important for the mentee to really WANT to learn more.

Never ever spoon feed them. Instead point them towards things of value and have them utilize the new information they are learning in projects that they are using. Knowledge is useless if not put into practice. So encourage your mentee to code, code, code.

Share Improve this answer

Follow

edited Aug 1, 2018 at 7:44

Gaurav

406 ● 1 ● 9 ● 22

answered Feb 1, 2010 at 2:27

Alex Baranosky

50k ● 45 ● 104 ● 151

2

Here would be my short list:

Pair programming - This is helpful for many things, like reinforcing various ideas and practices. Getting used to Resharper is much easier when you pair with someone that uses it often.

Informal chats - This is where we would go get a drink, go outside for someone to have a smoke break, go for lunch together, etc. While away from the desks, the discussion may be related to work immediately being done or it may be abstract philosophical stuff that can help bring someone's game up a notch or two. Talking about various upcoming technologies or changes in what coming can be exciting and help form bonds as well.

Feedback and suggestions - This is what occured in both above cases. Books like "How to Win Friends and Influence People" by Dale Carnegie can help in understanding various human relationship dynamics, which while that sounds quite technical is really just about how to motivate someone else in various ways. A key point here is to know how to leave a trail of breadcrumbs to pick up some practices, like giving hint after hint about something rather than just give the answer. I have had

various Math teachers that had a gift for this for how I developed some of this skill.

So part of this is merely motivating the other person and trying to guide them as when someone figures something out for themself it can be an empowering and enlightening experience. The, "I did it! That's right, moi, yours truly!" kind of self-talk is quite nice when it happens.

Share Improve this answer

Follow

answered Dec 1, 2009 at 17:11

JB King

11.9k ● 4 ● 40 ● 49

2

Ask them what would they try next to complete the task. This can give an idea of where from the "I don't know what to do" to "Well, I would try this but..." category are they in terms of having their own idea that may be useful for a starting point.

Take a quick look at what they want to do and offer hints so that they figure out the problem. This is rather than give the answer of, "Just take out this line of code," suggest they look at what is there and is it all necessary

Share Improve this answer

Follow

answered Apr 15, 2012 at 16:57

game avatar hd

21 ● 1

I would recommend start giving parts of real assignments you have and make everything to be able to use his code.

1

In other words train him as replacement for yourself.

This way you will commit yourself to allocating time to work with junior and he will be able to see "real life". By working on real assignments and hearing lively feedback he will be able to get p to speed rather quickly.

Drawback of this approach is that it is possible that it will have too narrow focus on you particular project. So be sure to show the trainee possible alternatives and encourage trade-offs analysis to widen his professional horizon.

Share Improve this answer

Follow

answered Aug 24, 2008 at 14:52

Dima Malenko

2,835 ● 2 ● 27 ● 24

1

Couple of years ago I worked for a small company, where on the first day I was given a list of small task to complete - do some little changes in code, find and fix a small bug in the project. It really helped me to ask the right questions from my mentor and familiarize myself with the environment, the code base. These tasks were easy to complete, so I had a little bit of self confidence, before turning to the bigger tasks.

This way of mentoring really worked with me very well, so I am planning to do the same with our new colleague.

Share Improve this answer

Follow

answered Aug 24, 2008 at 15:06

gyurisc

11.5k ● 17 ● 70 ● 106

0

I have mentored several junior people under me before. My approach varied slightly based on the person a bit based on how they learned.

In short, I gave the junior people small, self-contained projects when I could and gave them a relatively fixed time to complete the task. Once the task was complete I would review their approach, code and solution and made suggestions for improvements or a better way to handle the problem. I think this way they don't feel overwhelmed being part of a much larger project.

Hope this helps a bit.

Share Improve this answer

Follow

answered Aug 24, 2008 at 14:37

Rob Bazinet

1,020 ● 1 ● 8 ● 12