How do you incentivize good code? [closed]

Asked 16 years, 3 months ago Modified 16 years, 3 months ago

Viewed 1k times

5

Closed. This question is opinion-based. It is not currently accepting answers.

Want to improve this question? Update the question so it can be answered with facts and citations by editing this post.

Closed 5 years ago.

Improve this question

Are there any methods/systems that you have in place to incentivize your development team members to write "good" code and add comments to their code? I recognize that "good" is a subjective term and it relates to an earlier question about measuring the maintainability of code as one measurement of good code.

sdlc

Share

edited May 23, 2017 at 11:57

Improve this question

Follow

Community Bot

1 ● 1

asked Sep 24, 2008 at 17:39

Guy

67.1k ● 101 ● 265 ● 331

15 Answers

Sorted by: Highest score (default)

7

This is tough as incentive pay is considered harmful. My best suggestion would be to pick several goals that all have to be met simultaneously, rather than one that can be exploited.

Share Improve this answer

Follow

answered Sep 24, 2008 at 17:44

Jason Z

13.4k ● 15 ● 51 ● 62

6

While most people respond that code reviews are a good way to ensure high quality code, and rightfully so, they don't seem to me to be a direct incentive to getting there. However, coming up with a positive incentive for good code is difficult because the concept of good code has large areas that fall in the realm of opinion and almost any system can be gamed.

At all the jobs I have had, the good developers were intrinsically motivated to write good code. Chicken and egg, feedback, catch 22, call it what you will, the best way to get good code is to hire motivated developers.

Creating an environment where good developers want to work is probably the best incentive I can think of. I'm not sure which is harder, creating the environment or finding the developers. Neither is easy, but both are worth it in the long term.

I have found that one part of creating an environment where good developers want to work includes ensuring situations where developers talk about code. I don't know a skilled programmer that doesn't appreciate a good critique of his code. This helps the people that like to be the best get better. As a smaller sub-part of this endeavor, and thus an indirect incentive to create good code, I think code reviews work wonderfully. And yes, your code quality should gain some direct benefit as well.

Another technique co-workers and I have used to communicate good coding habits is a group review in code. It was less formal and allowed people to show off new techniques, tools, features. Critiques were made, kudos were given publicly, and most developers didn't seem to mind speaking in front of a small developer group where they knew everyone. If management cannot see the benefit in this, spring for sammiches and call it a brown bag. Devs will like free food too.

We also made an effort to get people to go to code events. Granted, depending on how familiar you all are with the topic, you might not learn too much, but it keeps people thinking about code for a while and gets people talking in an even more relaxed environment. Most devs

will also show up if you offer to pick up a round or two of drinks afterwards.

Wait a second, I noticed another theme. Free food! Seriously though, the point is to create an environment where people that already write good code and those that are eager to learn want to work.

Share Improve this answer

Follow

answered Sep 24, 2008 at 18:31

Chuck

8,272 ● 2 ● 31 ● 25

5

Code reviews, done well, can make a huge difference. No one wants to be the guy presenting code that causes everyone's eyes to bleed.

Unfortunately, reviews don't always scale well either up (too many cooks and so on) or down (we're way too busy coding to review code). Thankfully, there are some tips on Stack Overflow.

Share Improve this answer

Follow

edited May 23, 2017 at 12:28

Community Bot

1 ● 1

answered Sep 24, 2008 at 17:43

Jon Ericson

21.5k ● 12 ● 102 ● 151

I disagree. What you describe is called negative reinforcement. Effective in the short term, but not good for

building a team. Causes a stressful work environment.
– kemiller2002 Sep 24, 2008 at 17:55

@Kevin: I think you missed the "done well" qualifier. ;-)
– Jon Ericson Sep 24, 2008 at 20:36

4

I think formal code reviews fill this purpose. I'm a little more careful not to commit crappy looking code knowing that at least two other developers on my team are going to review it.

Share Improve this answer

Follow

answered Sep 24, 2008 at 17:42

Bill the Lizard

405k ● 211 ● 572 ● 889

3

Make criteria public and **do not connect incentives with any sort of automation**. Publicize examples of what you are looking for. Be nice and encourage people to publicize their own bad examples (and how they corrected them).

Part of the culture of the team is what "good code" is; it's subjective to many people, but a functioning team should have a clear answer that everyone on the team agrees upon. Anyone who doesn't agree will bring the team down.

Share Improve this answer

Follow

answered Sep 24, 2008 at 17:49

davetron5000
24.8k ● 11 ● 72 ● 99

2

I don't think money is a good idea. The reason being is that it is an extrinsic motivator. People will begin to follow the rules, because there is a financial incentive to do so, and this doesn't always work. Studies have shown that as people age financial incentives are less of a motivator. That being said, the quality of work in this situation will only be equal to the level you set to receive the reward. It's a short term win nothing more.

The real way to incent people to do the right thing is to convince them their work will become more rewarding. They'll be better at what they do and how efficient they are. The only real way to incentivize people is to get them to want to do it.

Share Improve this answer

Follow

answered Sep 24, 2008 at 17:52

kemiller2002

115k • 28 • 199 • 253

1

This is advice aimed at you, not your boss.

Always remind yourself of the fact that if you go that extra mile and write as good code as you can now, that'll pay off later when you don't have refactor your stuff for a week.

Share Improve this answer

Follow

answered Sep 24, 2008 at 17:47

Hans Sjunnesson

22.2k • 17 • 56 • 63

1

I think the best incentive for writing good code is by writing good code together. The more people write code in the same areas of the project, the more likely it will be that code conventions, exception handling, commenting, indenting and general thought process will be closer to each other.

Not all code is going to be uniform, but upkeep usually gets easier when people have coded a lot of work together since you can pick up on styles and come up with best practice as a team.

Share Improve this answer

Follow

answered Sep 24, 2008 at 17:49

nyxtom

2,929 • 2 • 24 • 24

You get rid of the ones that don't write good code.

I'm completely serious.

1

Share Improve this answer

Follow

answered Sep 24, 2008 at 17:59

core

33k • 45 • 140 • 195

1 Someone who is a great problem solver, might not write the best code. That's why code reviews, peer reviews, academy courses, etc. are important to our industry. A great problem solver can be taught how to write code...a great code writer can't be taught how to problem solve. – MaTT Sep 24, 2008 at 18:05

That may be fine for a Jr.-level developer. But a great problem-solver who writes poor code can be more dangerous than a worker bee who writes good code. If one is going to be more than Jr.-level developer, he or she must be a great problem solver AND write good code. – core Sep 24, 2008 at 19:37

Valid point, Chris. That's why I suggested mechanisms for teaching good problem solvers how to write good code. Both are a must if you want to have a very strong software developer...we are both right :-) – MaTT Sep 25, 2008 at 14:13

I agree with Bill The Lizard. But I wanted to add onto what Bill had to say...

1

Something that can be done (assuming resources are available) is to get some of the other developers (maybe 1 who knows something about your work, 1 who knows your work intimately, and maybe 1 who knows very little about it) together and you walk them through your code. You can use a projector and sit them down in a room and you can drive through all of your changes. This way, you have a mixed crowd that can provide input, ask questions, and above all make you a better developer.

There is no need to have only negative feedback; however, it will happen at times. It is important to take negative as constructive, and perhaps try to couch your feedback in a constructive way when giving feedback.

The idea here is that, if you have comment blocks for your functions, or a comment block that explains some tricky math operations, or a simple commented line that explains why you are required to change the date format depending on the language selected...then you will not be required to instruct the group line by line what your code is doing. This is a way to annotate changes you have made and it allows for the other developers to keep thinking about the fuzzy logic you had in your previous function because they can read your comments and see what you did else-where.

This is all coming from a real life experience and we continue to use this approach at my job.

Hope this helps, good question!

Share Improve this answer

Follow

answered Sep 24, 2008 at 18:14

MaTT

175 • 1 • 11

0

Hm.

Maybe the development team should do code-reviews of each other codes. That could motivate them to write better, commented code.

Share Improve this answer

Follow

answered Sep 24, 2008 at 17:43

Nils Pipenbrinck

86.2k • 33 • 155 • 223

> That'd fall nicely under pair-programming. Something I've tried to get going "here" to no avail. – hometoast Sep 24, 2008 at 17:44

0

Code quality may be like pornography - as the famous quote from the Justice Potter Stewart goes, "I know it when I see it"

So one way is to ask others about the code quality. Some ways of doing that are...

Code reviews by their peers (and reviews of others code by them), with ease of comprehension being one of the criteria in the review checklist (personally, I don't think that necessarily means comments; sometimes code can be perfectly clear without them)

Request that issues caused by code quality are raised at retrospectives (you do hold retrospectives, right?)

Track how often changes to their code works first time, or whether it takes several attempts?

Ask for peer reviews at the annuak (or whatever) review time, and include a question about how easy it is to work with the reviewee's code as one of the questions.

Share Improve this answer

Follow

answered Sep 24, 2008 at 17:47

The Archetypal Paul

41.7k ● 20 ● 106 ● 135

0

Be very careful with incentivizing: "What gets measured gets done". If you reward lines of code, you get bloated code. If you reward commenting, you get unnecessary comments. If you reward lack of bugs found in the code, your developers will do their own QA work which should be done by lower-paid QA specialists. Instead of incentivizing parts of the process, give bonuses for the whole team's success, or the whole company's.

IMO, a good code review process is the best way to ensure high code quality. Pair programming can work too, depending on the team, as a way of spreading good practices.

Share Improve this answer

Follow

answered Sep 24, 2008 at 17:47

Christian Oudard

49.8k ● 25 ● 70 ● 70

0

The last person who broke the build or shipped code that caused a technical support call has to make the tea until somebody else does it next. The trouble is this person probably won't give the tea the attention it requires to make a real good cuppa.

Share Improve this answer

Follow

answered Sep 24, 2008 at 17:58

Scott Langham

60.3k ● 42 ● 136 ● 210

0

I usually don't offer my team monetary awards, since they don't do much and we really can't afford them, but I usually sit down with each team member and go over the code with them individually, pointing out what works ("good" code) and what does not ("bad" code). This seems to work very well, since I don't get nearly as much junk code as I did before we started this process.

Share Improve this answer

Follow

answered Sep 24, 2008 at 18:03

Iwasakabukiman

1,453 ● 4 ● 15 ● 17