Is it worth converting my functional JavaScript code to an object-oriented design?

Asked 16 years, 2 months ago Modified 15 years, 6 months ago

Viewed 1k times

8

I'm currently building a small web application that includes a fair amount of JavaScript. When I was prototyping the initial idea, I just hacked together a few functions to demonstrate how the application would eventually behave intending to go forward re-writing the JavaScript in an object-oriented nature.

Now that I'm getting into the implementation phase, I'm finding that creating object-oriented JavaScript for the sake of being object-oriented seems overkill - the project isn't likely to require any major modifications in the future that would warrant and object-oriented design. Instead, I'm finding that a set of concise, cohesive functions are working well.

So, with that said and with attempting to adhere to the KISS principle, when a set of functions are providing a suitable solution to a problem, are there any other reasons worth considering to convert my code into an object-oriented design?

javascript software-design web-applications

Share

Improve this question

Follow

edited Oct 6, 2008 at 12:11

Adam Haile

31.3k ● 60 ● 195 ● 290

asked Oct 6, 2008 at 12:00

Tom

15.9k ● 5 ● 50 ● 63

8 Answers

Sorted by: Highest score (default)

13

No, although I personally find OOP more tasty, it is a means to an end, and not an end in itself. There are many cases where procedural programming makes more sense than OOP, an converting for the sake of converting, could be, as you said, overkill.

Share Improve this answer

Follow

answered Oct 6, 2008 at 12:05

Vincent McNabb

34.5k ● 7 ● 33 ● 54

11

No, let it be and move forward - that is more productive in my view .

Share Improve this answer

Follow

edited Jun 10, 2009 at 9:49

answered Oct 6, 2008 at 12:02

Johan Bresler

6,508 ● 11 ● 59 ● 77

> That was a bit of a shameless bump of an edit, no? Please refrain from this, it's just artificially increasing noise on the front page. – annakata Jun 10, 2009 at 9:54

6

If your code is well structured, well laid out, and well commented, and does the job that is required of it, then messing with it for any reason other then to add features is ill-advised.

While it might be nice to say that the program is nicely OOP etc, if it doesn't need to be changed to work then I would definetely leave it as it is.

If it aint broke, dont fidgit with it :)

Share Improve this answer

Follow

answered Oct 6, 2008 at 12:13

Jimoc

1,500 ● 1 ● 10 ● 20

3

If this code is already implemented and won't require maintenance or - better yet - upgrades, stick with it. If you are going to implement it now and it could get complex, consider the OO approach.

Experience has shown me that it's pretty easy to write and maintain procedural code while complexity is low, but after a certain threshold it starts getting exponentially more difficult to increase complexity while using procedural programming, whereas OOP, although harder to begin, keeps complexity much more manageable.

Bottom line: if the task is simple enough or has already been implemented, keep it simple. If it might grow more complex, consider OOP.

Share Improve this answer

Follow

answered Oct 6, 2008 at 13:07

schonarth

534 ● 7 ● 13

2

I would say that it is still worth reviewing your code before making a decision. The obvious downside to "re-writing" code is that there is a testing cost to ensure that your code works the same as before. Do you have any Unit tests? If not, then your testing cost is even higher. So in general, I'm against re-writing working code unless it serves another end, which is to allow you to more easily write new functionality that is now required (i.e. refactoring common functions, etc.)

HOWEVER, any time a person says "I hacked together", I suggest it is always worth a second look at your code. Why was it hacked together in the first place? I know plenty of people say that Object Oriented code isn't an end in and of itself, but it is a methodology that after while doesn't have to be thought about either. You just sort of naturally start doing it.

Maybe your js is relatively simple, and therefore OO scafolding is truly extra overhead. Fine. But I still suggest that you should always code review (and especially have someone else review) any code you call "hacked". Perhaps it was a Freudian slip... but it did slip.

Share Improve this answer

Follow

answered Oct 6, 2008 at 13:39

Nick

5,935 ● 1 ● 30 ● 38

@Nick, No Freudian slip - the code was 'hacked' together for prototyping and demonstrating eventual functionality. I'd never release hacked code into a production environment =). Professionally, I write OO code all day so *not* coding in OO feels almost unnatural - that's why I wanted feedback.
– Tom Oct 6, 2008 at 14:04

0

Treat it as legacy code from now on. When you want to change something, refactor it so the code becomes easier on the mind. If you need a bit of OOP, use it. If you don't, don't.

OOP is a hammer, please don't treat a screw-problem as a nail.

Share Improve this answer

Follow

answered Oct 6, 2008 at 12:18

Patrick Huizinga

1,340 ● 11 ● 25

0

If it works, and it's easy to maintain, I wouldn't bother converting it for convertings sake. There must be more interesting things to do.

Share Improve this answer

Follow

answered Oct 6, 2008 at 12:22

Bart Read

2,757 ● 3 ● 22 ● 32

0

Just bare in mind Objects are rather expensive to create in javascript.

Keep construction of objects to a bare minimum.

Share Improve this answer

Follow

answered Oct 6, 2008 at 13:30

Kent Fredric

57.3k ● 14 ● 111 ● 151