Quality Code Notes

Prof Bill - Apr 2017

Here are my notes on writing quality code for our Capstone2017 project.
“Quality” means code that works and is easily understood by others.
“Code” means Rails, Ruby, ERB, HTML, and CSS files.
These are guidelines that are applicable in most cases; not hard and fast rules.

1. Rails Tutorial - Generally, follow the Rails tutorial. Hartl writes quality code.

2. Idiom - Follow Rails idioms, where possible. We’re new at this, so again, the tutorial is a good starting point. I haven’t found a better source than Hartl.

3. Comments - There are four flavors of comments:

➔ For a new file, comment yourself as author and a brief description

```
# author: Prof Bill
# Controller for the automatic code review facility
```

➔ Comment at the top of each method

```
# Returns true if the given token matches the digest
def authenticated?(attribute, token)
  ...
end
```

➔ Comment complex code sections, include URL if solution is from the Internets

```
# This is weird. We delicately retrace our steps in log N time
# I found this solution here.
# http://www.example.com/retrace_this.htm
```

➔ Identify yourself in a comment if you make a significant change to someone else’s code

```
# Changed by: Prof Bill
# I changed the spark plugs and rotated the tires
```

4. Document references - If you find a solution on the Internets (google, stackoverflow, tutorial, etc), please document the URL source as a comment in your code.

5. No SQL, please - One of Rails’ more important idioms is accessing the database without explicit/native SQL commands. If you think you **must** use SQL to get the functionality you need, then please show me.

6. ERB, HTML, and CSS - Yes, this is code! Someone in the future will have to understand it. So, please use comments to ID authorship, explain tricky parts, etc. For these more structured languages, consistent indentation and spacing is critical.

7. Regression tests - Include at least 5 tests with your code in each Sprint. Yes, this number is arbitrary, and it's a minimum. Doing more is better. It's like wearing more flair at Chotchkie's: youtu.be/KJtrLKGZZFg

8. Code reviews - We'll do code reviews at the end of each sprint. You'll review someone else's code. And vise versa. Use these notes to help you give constructive feedback when reviewing code. Example: "Hey, this method doesn't have a comment". "Please comment this block of complex code. I don't understand it."

Well, my quality list is small, but it's critical. Quality code looks good and is understandable by others. Again, Hartl writes quality code. That's a good bogey.

It's hard to overstate these two emotions related to quality code:

- ❏ **JOY** - You're debugging some problem. You visit someone else's code, and it's clear and commented and easy to follow. Huzzah!
- ❏ ***despair*** - You visit code to change it... it's complex, hard-to-follow and comment-free. Wah.

Owners write quality code.
thanks… yow, bill