Code Smells, Names and More

Admin

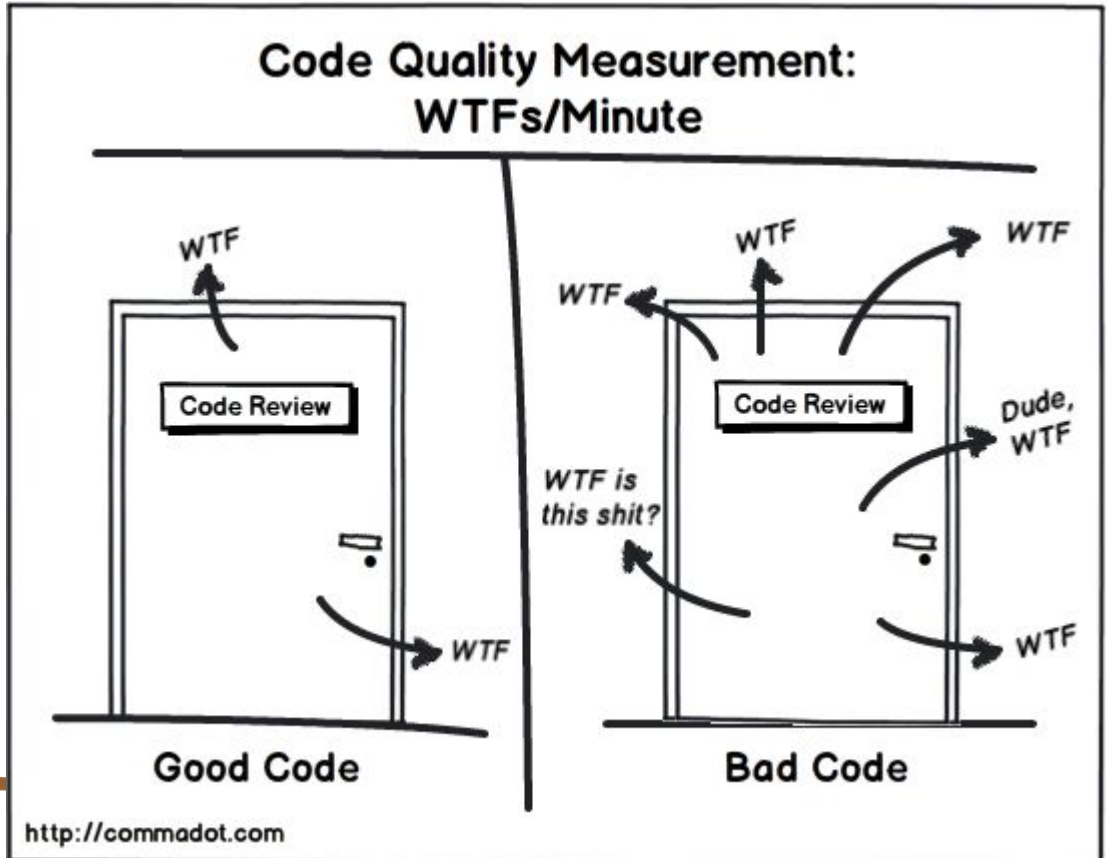
- Retrospective for sprint 2
  - In groups (if week of Feb 14th)
- Last Quiz coming next week
- Reading Assignment
  - Please read Chapter 2 of the Pragmatic programmer for next week

Code Metrics

- There have been many metrics for measuring code over the years
  - Lines/day
    - Or number of lines period
  - Bugs/line
    - Discuss bug bounty economy
  - Cyclomatic Complexity
    - How many paths are there through the code?
    - How hard will this be to test?
  - Class Coupling
  - Churn
    - How often is this part of the code edited/committed in git?
  - Test coverage/’maintainability index’
  - Which is best metric (here or something else)?

Best Code metric

- I agree with several others (including the old clean code book)



Code Review

- Some of you have done Code reviews
  - So what are they?

Code Review

- Some of you have done Code reviews
  - As academics all too often we toss the code together just in time for due date and never return to it.
  - In industry this is only going to happen to companies that go bankrupt very quickly
  - So we have someone(s) look at the code before we let it go to production.
-

Code Review

- Some of you have done Code reviews
  - Code review might be a formal review scheduled by management
    - If defense contractor software this is (was?) required.
    - Original author may or may not be present.
    - For good code original author not needed – it is clear what the code does
  - Code review might be informal – from time to time the team looks over the code it is responsible for.
    - A few team members look over a package every X units of time.
    - Pair programming example
    - Or even pull request approval

Code review

- To be successful most code reviews are peer review.
  - Management should not be present
  - Might not be possible if review is result of epic fail
    - Iowa Caucus app 2020 perhaps
  - Hawaii missile warning of 2018
  - Boeing 737 Max
  - Review should
    - Spot bugs and vulnerabilities
    - Find possible un-clean code
    - Produce recommendations for cleaner code.

Automated Code Review

- Today most of the style code review work is done by automated tools.
  - e.g. flake8 https://flake8.pycqa.org/en/latest/ for python
  - Or checkstyle for java
  - If the style check fails, auto reject a code submission.
  - Benefits?

Automated Code Review

- Today most of the style code review work is done by automated tools.
  - e.g. flake8 https://flake8.pycqa.org/en/latest/ for python
  - Or checkstyle for java
  - If the style check fails, auto reject a code submission.
  - Benefits?
  - No ego
    - No hurt feelings when John used the wrong spacing scheme **again** and tried to submit that code.

Pull Requests

- A very common place for informal code reviews is a pull request
  - Pull requests?

Pull Requests

- A very common place for informal code reviews is a pull request
  - Pull requests?
  - Github introduced workflow
  - Back when I was doing professional software dev we just committed to the source repository after testing a bit ourselves
  - Today almost no one does that. You write your code, then make a ‘pull request’ to have your code integrated with the main codebase.

Code

- Over the last 45 years or so
  - Programming languages of choice → ever higher level of abstraction
  - Top languages of the era
    - Assembler → C → c++ → java → a plethora of specialized higher abstraction languages
    - Of course in each era there were other languages
  - With automatic tools to convert UML to code skeletons and lego brick and MIT app inventor
  - And of course ChatGPT and friends will replace us right?
    - https://blog.bitsrc.io/i-asked-chat-gpt-to-build-a-to-do-app-have-we-finally-met-our-replacement-ad347ad74c51

Code

- Over the last 45 years or so
  - Programming languages of choice → ever higher level of abstraction
  - Top languages of the era
    - Assembler → C → c++ → java → a plethora of specialized higher abstraction languages
    - Of course in each era there were other languages
  - With automatic tools to convert UML to code skeletons and lego brick and MIT app inventor
  - And of course ChatGPT and friends will replace us right?
    - https://blog.bitsrc.io/i-asked-chat-gpt-to-build-a-to-do-app-have-we-finally-met-our-replacement-ad347ad74c51
  - And Gartner says we'll see some development jobs turn into utilities:
  - https://www.gartner.com/newsroom/id/3707317
    - Do we need code any more?

Code

- Over the last 45 years or so
  - Programming languages of choice → ever higher level of abstraction
  - Top languages of the era
    - Assembler → C → c++ → java → a plethora of specialized higher abstraction languages
    - Of course in each era there were other languages
  - And Gartner says we'll see some development jobs turn into utilities:
  - https://www.gartner.com/newsroom/id/3707317
  - Do we need code any more?
    - Yes!! it may not be as code-y as the old days – but it is code.
    - No need for assembler in 2000s
    - Fewer people need to do low level stuff today, but

Code Changes

- I talked about this briefly weeks ago, but when making a code change to a method, what is the activity that you spend the most time on?

Code Changes

- I talked about this briefly weeks ago, but when making a code change to a method, what is the activity that you spend the most time on?
  - Reading other code
    - Need to read the rest of the class
    - Need to read places that call this method
    - Need to read other methods that this one calls
  - Need to understand the context of this code change
  - That code that you write will be read by many people
  - Clean Code motto: leave the code a little better than you found it.
  - Pragmatic Programmer Corollary:
    - Fix the broken windows.

Names

- What do we name?
- What did your (undergrad) software engineering class say about names?
  - I know Professor Matta covers this.

Names

- What do we name?
- We name everything in our code
  - Variables:
    - Local variables, parameters, global variables, member data
  - Functions:
    - Methods, member functions, functions, procedures etc.
  - Classes
  - Filenames
  - Executable and library names
- So pick good names!!!
- What did your (undergrad) software engineering class say about names?

Make clear names

- You've heard this from me and others for years now
- But your names should tell you what is stored there
- int t;
- Is this a good name?

Make clear names

- You've heard this from me and others for years now
- But your names should tell you what is stored there
- int t;
- Is this a good name?
  - Of course not – and you told me why right?

Make clear names

- You've heard this from me and others for years now
- But your names should tell you what is stored there
- int t;
- Is this a good name?
  - Of course not – and you told me why right?
- int timeElapsed;
  - How about this? Is this a good name?
  -

Make clear names

- You've heard this from me and others for years now
- But your names should tell you what is stored there
- int t;
- Is this a good name?
  - Of course not – and you told me why right?
- int timeElapsed;
  - How about this? Is this a good name?
  - Better – but still needs work – how can we fix it?

Make clear names

- You've heard this from me and others for years now
- But your names should tell you what is stored there
- int timeElapsedinSeconds;
- Now this actually tells you what is stored in this variable – no need for a comment.
- If only the Mars Climate Orbiter Team had used such good names.

One Character Variable names

- When and why are one character variable names ok
  - Because usually they are not

Avoid Types in a name

- Avoid having a type mentioned in the variable name
  - AccountList
  - NameString
  - ClassName

Avoid Types in Name

- Avoid having a type mentioned in the variable name
  - AccountList (what happens when we switch to a hashtable?)
  - NameString (what about when it becomes a char array?)
  - ClassName (we swapped out for a struct?)
- Putting type names sets us up for difficulty in maintaining code

Lying Code

- Who ya gonna believe – my comment or that lying code?
  - You'd think I wouldn't have to do this one.
  - Don't make your code lie
  - int name; //the student's bannerID
    - So what if the comment is correct!? It doesn't make up for the lying variable name
    - Or worse yet: int nameString
  - set<account> accountList;
  - String studentClass;
  - Painful – and all too common among students
    - And impossible to maintain.

Spelling Mistakes

- I used joke at the board,
  - But spelling matters. Auto spell check is available everywhere these days
  - Don't rely on spelling mistakes to make it compile
  - int foo(int number, Section klass){
    - int numb3r; //I needed another copy locally
    - ….
    - }
  - No!!!!!!!!!
  - And klass? Really!!??!

Make Autocomplete your friend

- Make your names distinct enough
  - so that when auto complete makes a suggestion, you'll know which one to choose
  - Everyone has auto complete – but make it easy – you've seen how pycharm/intellij/VS code 'helps' sometimes
  - In six months
  - getCurrentStudent()
  - getCurrent()
  - getCurrentStudentInfo()
  - Hmmm which do I call and when?

Autocomplete and AI make longer names fine

- But don't be silly about it

Average  test method name

```
[Test]
public void
WhenClientAttemptsToGetAllCatPicturesWithoutAValidAuthorizationCo
okieHeaderThenTheGetAllMethodOfTheCatPicturesControllerReturnsA40
1UnauthorizedResponseWithInvalidAuthorisationCookieAsAMessage()
{
    ...
}
```

oO0 l1L

- Ok – how about avoiding O and l unless it is very clear in context what they are?
  - What is the title string?
  - If you can't tell instantly, change the variable name

Class and Method names

- You've been told since CS2
  - Classes represent objects/nouns
    - So use a noun to name it
    - ManageDatabase is a bad name for a class
  - Methods represent verbs/actions
    - So use a verb for the methods.

Making the point with humor

Don’t be cute

- Don't be cute and don't pun
  - HolyHandGrenade or anExParrot
    - Everyone with enough geek cred over 30 or 35 should know this – do you?
    - Not likely – don't be cute – it doesn't last
  - Don't pun either
- Its all fun when you are doing it
  - In 20 years when the code is still in use its just annoying
  - Really how about “don’t tase me bro” or the code below, both only 14 years old, or a Tide Pods reference? (what maybe 6 years?)
  - Today it might be a variable called upDog
  -

```
6  var diducomefrom;
7  var didugo;
8  var diducomefromcottoneyedjoe?
```

Don’t Be an Example

- Don’t be an example for a future version of this class

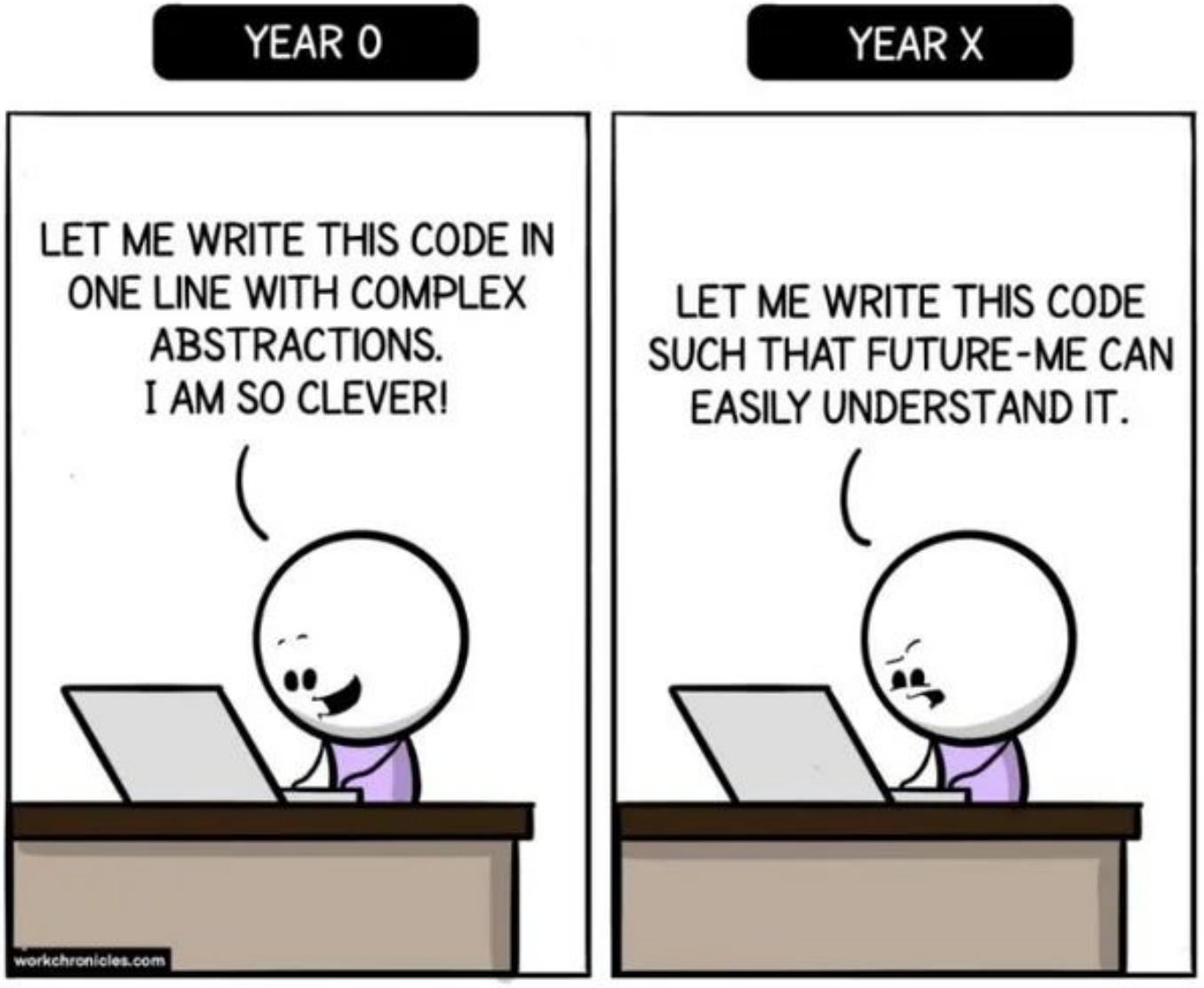
```
for(var key in data)
{
  console.log(key)
  mappingKeys.push(key)
}
//console.log("-----------------------------------------------------------------------------------------------------------")
console.log("-----------------------------------------------------------------------------------------------------------")
mappedKeys = await this.dbinterface.getQuoteFormFields(mappingKeys);
mappedValues = await this.dbinterface.getMappedValues(target,mappingKeys);
console.log(mappedValues)

console.log("-----------------------------------------------------------------------------------------------------------")
//console.log("-----------------------------------------*****-----------------------------------------------------------")
for(var key in data)
{
  for(var gumgum in mappedKeys)
  {

    if(mappedKeys[gumgum].id == key)
    {
      var dumbdumb = mappedKeys[gumgum].name
      for var monkey in mappedValues)
      {
        //console.log("/////*****/////")
            //console.log(mappedValues[monkey].value, "===", data[key], "?????", mappedValues[monkey].valueID)
          //console.log("/////*****/////")
        if(mappedValues[monkey].valueID == data[key] && mappedValues[monkey].fn_qf == key)
        {

          mappedData[dumbdumb] = mappedValues[monkey].value
```

Not just class and variable names

- struct nelson_mandela *NelsonMandela = (struct nelson_mandela *)malloc(sizeof(struct nelson_mandela));
-
- /* a few pages later ... */
-
- free(NelsonMandela);

- Even file names
  - A perl file called
  - doeeeeeeeiiiiiiiit.pl

Final Thoughts on Readability



Related: Software Version Numbers

- You’ve seen version numbers
  - What do they mean?
  - What are the two widely used version numbering systems in use today?
    - Semantic versioning and Calendar Versioning (CalVer)
    - What do these mean?
    - What does Ubuntu version 20.04 mean?
    - What does gcc 7.4.0 mean?

More reading

- Some abridged summaries
- https://blog.aspiresys.pl/technology/express-names-in-code-bad-vs-clean/
- https://hilton.org.uk/blog/naming-smells
- https://www.lesswrong.com/posts/NYaLudjSqsYtZDB2t/bad-names-make-you-open-the-box
- And with a warning that it is reddit, plenty of people sharing nonsense, including code they thought was hilarious as teenagers
  - You have been warned.
  - https://www.reddit.com/r/programming/comments/klhlv/what_is_the_worst_classvariablefunction_name_you/
  -

Last Word

- Last word
  - Conventions in a language are important
  - But they are just that, conventions. The key is to make the software easy to read and therefore easy to change.