



mboom/TI2806

Last updated: May 27, 2016




YOUR TASKLIST



Write Short Units of Code

✖⁰

Guideline

- Small units are easier to understand, reuse, and test.
- When writing new units, don't let them grow above 15 lines of code.
- When a unit grows beyond 15 lines of code, you need to shorten it by splitting it in smaller units of no longer than 15 lines of code.
- The list on the right side contains a sorted by severity selection of units that violate this guideline and the colors indicate the severity of the violation:  more than 60 lines of code,  more than 30 lines of code,  more than 15 lines of code.

Refactoring candidates




- ☐ jquery.js:DefaultUnit
- ☐ qunit.js:DefaultUnit
- ☐ jquery.js:Sizzle.setDocument
- ☐ jquery.js:\$anonymousobject.ajax
- ☐ qunit.js:QUnit.load
- ☐ time-and-pr-size.js:\$anonymousobject.body
- ☐ jquery.js:defaultPrefilter
- ☐ qunit.js:\$anonymousobject.finish
- ☐ Gruntfile.js:module.exports
- ☐ filled-graph.js:\$anonymousobject.body
- ☐ jquery.js:Sizzle



Write Simple Units of Code

✖⁰

Guideline

- Keeping the number of branch points (if, for, while, etc.) low makes units easier to modify and test.
- Try to keep the number of branch points in a unit below 5.
- You can reduce complexity by extracting sub-branches to separate units of no more than 5 branch points.
- The list on the right side contains a sorted by severity selection of units that violate this guideline and the colors indicate the severity of the violation:  more than 25 branch points,  more than 10 branch points,  more than 5 branch points.

Refactoring candidates

- ☐ jquery.js:DefaultUnit
- ☐ qunit.js:DefaultUnit
- ☐ jquery.js:Sizzle.setDocument
- ☐ jquery.js:\$anonymousobject.ajax
- ☐ jquery.js:defaultPrefilter
- ☐ qunit.js:\$anonymousobject.finish
- ☐ jquery.js:Sizzle
- ☐ jquery.js:\$anonymousobject.buildFragment
- ☐ jquery.js:\$anonymousobject.domManip
- ☐ jquery.js:setMatcher
- ☐ jquery.js:matcherFromGroupMatchers.superMatcher



Write Code Once

✖⁰

Guideline

Refactoring candidates

- › When code is copied, bugs need to be fixed in multiple places. This is both inefficient and error-prone.
- › Avoid duplication by never copy/pasting blocks of code.
- › Reduce duplication by extracting shared code, either to a new unit or to a superclass.
- › The list on the right side contains sets of modules (grouped by highlighting) which contain the same duplicated code block.

SIGN OUT






Keep Unit Interfaces Small



0



Guideline

- › Keeping the number of parameters low makes units easier to understand and reuse.
- › Limit the number of parameters per unit to at most 4.
- › The number of parameters can be reduced by grouping related parameters into objects.
- › The list on the right side contains a sorted by severity selection of units that violate this guideline and the colors indicate the severity of the violation:  more than 7 parameters,  more than 4 parameters,  more than 2 parameters.

Refactoring candidates

- ☐ jquery.js:setMatcher
- ☐ jquery.js:matcherFromGroupMatchers.superMatcher
- ☐ jquery.js:\$anonymousobject.on
- ☐ jquery.js:\$anonymousobject.access
- ☐ jquery.js:augmentWidthOrHeight
- ☐ OctopeerHelper.js:OctopeerHelper.area
- ☐ jquery.js:condense
- ☐ OctopeerHelper.js:OctopeerHelper.line
- ☐ materialize.js:\$anonymousobject.easeOutBounce
- ☐ jquery.js:\$anonymousobject.init
- ☐ materialize.js:\$anonymousobject.easeInOutElastic






Separate Concerns in Modules



0



Guideline

- › Keep the codebase loosely coupled, as it makes it easier to minimize the consequences of changes.
- › Identify and extract responsibilities of large modules to separate modules and hide implementation details behind interfaces.
- › Strive to get modules to have no more than 10 incoming calls.
- › The list on the right side contains a sorted by severity selection of modules that violate this guideline and the colors indicate the severity of the violation:  more than 50 incoming calls,  more than 20 incoming calls,  more than 10 incoming calls.

Refactoring candidates

- ☐ OctopeerHelper.js
- ☐ require.js



Couple Architecture Components Loosely



0



Guideline

- › Having loose coupling between top-level components makes it easier to maintain components in isolation.

Refactoring candidates

- › Do this by minimising the amount of code in a component that has dependencies on modules in other components.
- › You can hide a component's implementation details through various means, e.g. using the "abstract factory" design pattern.
- › The list on the right side contains a sorted by severity selection of modules that are being called from modules in other components (incoming), or have incoming calls as well as calls to modules of other components.

SIGN OUT



Keep Architecture Components Balanced



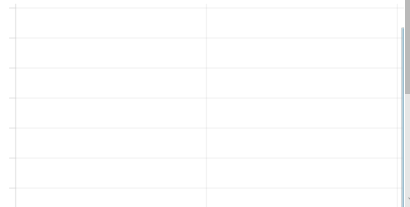
Guideline

- › Balancing the number and relative size of components makes it easier to locate code.
- › Organize source code in a way that the number of components is between 2 and 12, and ensure the components are of approximately equal size (keep Gini coefficient less than 0.71).
- › Organising components based on functionality makes it easier to divide your code into components.

Components overview

0.52
Gini coefficient

3
Components



Keep Your Codebase Small

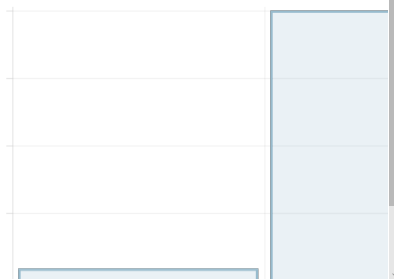


Guideline

- › Keeping your codebase small improves maintainability, as it's less work to make structural changes in a smaller codebase.
- › Avoid codebase growth by actively reducing system size.
- › Refactor existing code to achieve the same functionality using less volume, and prefer libraries and frameworks over "homegrown" implementations of standard functionality.
- › Strive to keep volume below 20 Man-years.

Volume overview

11
Man-months



Automate Tests



Guideline

- › Automating tests for your codebase makes development more predictable and less risky.
- › Write unit tests that amount to at least 80% coverage.
- › Add tests for existing code every time you change it.

Testing overview

- ☐ average-comment-size-compared.js
- ☐ average-comment-size-compared.js
- ☐ average-comment-size-compared.js
- ☐ average-comment-size-compared.js

N/A


Test coverage

- › To update the status on the right, regularly send coverage information to Coveralls.io

Write Clean Code

Guideline

- › Clean code is more maintainable.
- › Proactively search and remove code smells.
- › Remove useless comments, commented code blocks, and dead code. Refactor poorly handled exceptions, magic constants, and poorly names units or variables.
- › The list on the right side contains a selection of violations for this guideline.

- ☐ time-and-pr-size.js
- ☐ time.js
- ☐ animated-bar-chart.js
- ☐ filled-graph.js
- ☐ pr-size.js
- ☐ time-and-pr-size.js  0
- ☐ average-comment-size-compared.js
- ☐ average-comment-size-yours.js

Refactoring candidates

- ☐ time-and-pr-size.js
- ☐ main.js
- ☐ time.js
- ☐ qunit.js
- ☐ average-comment-size-compared.js
- ☐ average-comment-size-yours.js
- ☐ average-comment-size-total.js
- ☐ average-comment-size-total.js
- ☐ average-comment-size-compared.js
- ☐ average-comment-size-compared.js
- ☐ graph1.js
- ☐ graph1.js
- ☐ pr-size.js
- ☐ time-and-pr-size.js
- ☐ average-comment-size-compared.js
- ☐ average-comment-size-total.js
- ☐ average-comment-size-compared.js
- ☐ pr-size.js
- ☐ pr-size.js
- ☐ time-and-pr-size.js
- ☐ time-and-pr-size.js
- ☐ time.js
- ☐ average-comment-size-compared.js
- ☐ average-comment-size-yours.js
- ☐ time-and-pr-size.js
- ☐ time-and-pr-size.js
- ☐ pr-size.js
- ☐ time-and-pr-size.js
- ☐ pr-size.js
- ☐ time.js
- ☐ BitbucketAPI.js
- ☐ GithubAPI.js
- ☐ BitbucketAPI.js
- ☐ OctopeerAPI.js
- ☐ GithubAPI.js
- ☐ pr-size.js
- ☐ time-and-pr-size.js
- ☐ jquery.js
- ☐ jquery.js
- ☐ time.js
- ☐ average-comment-size-yours.js
- ☐ pr-size.js
- ☐ average-comment-size-total.js
- ☐ average-comment-size-compared.js

[SIGN OUT](#)