RUBY STYLE GUIDE

The personal style guide of Xavier Shay. Any potential exceptions must be justified and documented. Code examples can be revealed by clicking ↩ , and these imply more rules than are explicitly labeled. Rules that I often use editor features or scripts to apply are notated with ⚙ and a link to details.

Older code bases should migrate to this style, rather than stay consistent with themselves. Style upgrades should be in separate commits from behaviour changes.

The canonical URL for this document is http://xaviershay.com/ruby-style-guide.

ARRAYS

- Use a trailing comma on final entry of multi-line definitions. ↩
- Prefer `%w[]` for construction of string arrays. It is easier to write, and harder to mess up syntactically. ↩
- Prefer `%i[]` for construction of symbol arrays. ↩

ASSIGNMENT

- Prefer using instance variables rather than writer methods in constructors. ↩
- Prefer separating assignment from conditionals. ↩
- If using an assignment as a boolean value in a conditional, enclose in parentheses to indicate it's not a `=` vs `==` typo. ↩
- Only use parallel assignment for short variable names or when splitting the return value of a method. ↩

BLOCKS AND PROCS

- Use curly-braces when block is being chained. ↩
- Use stabby syntax with parenthesis. ↩

- Omit parenthesis for zero-argument lambdas. ↩
- Prefer `Symbol#to_proc` (the &preztel operator) where applicable. ↩
- Use `.()` to call. ↩

CLASSES

- Assign empty subclasses rather than inheriting. ↩
- Assign struct subclasses so that they can be re-opened. ↩
- Alias `[]` to `new` for value objects. ↩
- Nest namespaced definitions. ↩

CONDITIONALS

- Only use ternary operator for very short conditionals. When in doubt, avoid. ↩
- Use `&&` and `||` in preference to `and` and `or`. The latter have looser precedence and often lead to unexpected bugs. ↩

DEPENDENCIES

- Explicitly require third-party code in each file it is used.
- Prefer constant stubbing when depending on code that you own. ↩
- Prefer constructor or parameter injection when depending on code you do not own. When providing code to others, include fake versions of your classes. ↩

HASHES

- Use a trailing comma on final entry of multi-line definitions. ↩
- Prefer 1.9 syntax, though stay consistent within a single definition. ↩
- Construct from an array using `each_with_object`. ↩
- Omit `{}` when passing as the last argument to a method. ↩
- Prefer `fetch` for providing default values. ↩

ENUMERATION

- Prefer lisp-style enumeration methods. ↩

LINE LENGTH

- 80 characters is good enough for anyone. ⚙
- Long URLs are exempt, though consider shortening them.
- Line up method arguments when they do not fit on one line. ↩
- Squeeze multiline strings if you need to remove newlines. ↩
- Use `sprintf` (the `%` operator) to interpolate variables into long strings. ↩
- One line per method for long chains. ↩

METHODS

- Use parentheses to enclose parameters in method definitions. ↩
- Use a single line for blocks of trivial methods. ↩
- Use `_` for ignored arguments. ↩
- Use `*` when all arguments are ignored. ↩
- Prefer using parentheses to send messages with arguments. ↩
- Uses parentheses when using return value of a call. ↩
- Omit parentheses when message has no arguments. ↩
- Omit parentheses for DSLs when the return value is not used. ↩

NAMING

- Use short variable names for simple and obvious blocks. ↩
- Use short variable names for math. ↩
- Use `?` suffix for methods rather than `has_` or `is_` prefix. ↩
- Use modules or classes for namespacing rather than suffixes, except when necessarily following Rails conventions. ↩
- Avoid double namespacing. ↩

NIL VALUES

- Prefer `||` to ternary for default values. ↩
- Prefer `||=` for memoization. ↩
- Prefer `&&` for nil guards. ↩

REGEXES

- Make with `%r{}` when escaping of slashes would otherwise be required. ↩
- Use `\A` and `\z` rather than `^` and `$` for start and end of line matching. ↩
- Use `String#[]` for extraction. ↩

WHITESPACE

- No trailing whitespace. ⚙
- Two space indentation. ↩
- No indent after `private` or `protected` modifiers. ↩
- Align `if` blocks with left indent. ↩
- Align 1.8 style hashes along hash rocket. ⚙ ↩
- Align 1.9 style hashes along right-hand side. ⚙ ↩
- Align assignment along equals sign. ⚙ ↩
- No space between block brace and parameters. ↩
- No space between method definition name, argument, and bracket. ↩
- No space between method call name, argument, and bracket. ↩

Source for this document