# styleguide

# Go Style

https://google.github.io/styleguide/go

Overview | Guide | Decisions | Best practices

## About

The Go Style Guide and accompanying documents codify the current best approaches for writing readable and idiomatic Go. Adherence to the Style Guide is not intended to be absolute, and these documents will never be exhaustive. Our intention is to minimize the guesswork of writing readable Go so that newcomers to the language can avoid common mistakes. The Style Guide also serves to unify the style guidance given by anyone reviewing Go code at Google.

| Document | Link | Primary Audience | Normative |
|---|---|---|---|
| **Style Guide** | https://google.github.io/styleguide/go/guide | Everyone | Yes |
| **Style Decisions** | https://google.github.io/styleguide/go/decisions | Readability Mentors | Yes |
| **Best Practices** | https://google.github.io/styleguide/go/best-practices | Anyone interested | No |

### Documents

1. The **Style Guide** outlines the foundation of Go style at Google. This document is definitive and is used as the basis for the recommendations in Style Decisions and Best Practices.

2. **Style Decisions** is a more verbose document that summarizes decisions on specific style points and discusses the reasoning behind the decisions where appropriate.

   These decisions may occasionally change based on new data, new language features, new libraries, or emerging patterns, but it is not expected that individual Go programmers at Google should keep up-to-date with this document.

3. Best Practices documents some of the patterns that have evolved over time that solve common problems, read well, and are robust to code maintenance needs.

   These best practices are not canonical, but Go programmers at Google are encouraged to use them where possible to keep the codebase uniform and consistent.

These documents intend to:

- Agree on a set of principles for weighing alternate styles
- Codify settled matters of Go style
- Document and provide canonical examples for Go idioms
- Document the pros and cons of various style decisions
- Help minimize surprises in Go readability reviews
- Help readability mentors use consistent terminology and guidance

These documents do **not** intend to:

- Be an exhaustive list of comments that can be given in a readability review
- List all of the rules everyone is expected to remember and follow at all times
- Replace good judgment in the use of language features and style
- Justify large-scale changes to get rid of style differences

There will always be differences from one Go programmer to another and from one team's codebase to another. However, it is in the best interest of Google and Alphabet that our codebase be as consistent as possible. (See guide for more on consistency.) To that end, feel free to make style improvements as you see fit, but you do not need to nit-pick every violation of the Style Guide that you find. In particular, these documents may change over time, and that is no reason to cause extra churn in existing codebases; it suffices to write new code using the latest best practices and address nearby issues over time.

It is important to recognize that issues of style are inherently personal and that there are always inherent trade-offs. Much of the guidance in these documents is subjective, but just like with `gofmt`, there is significant value in the uniformity they provide. As such, style recommendations will not be changed without due discourse, Go programmers at Google are encouraged to follow the style guide even where they might disagree.

## Definitions

The following words, which are used throughout the style documents, are defined below:

- **Canonical**: Establishes prescriptive and enduring rules

  Within these documents, "canonical" is used to describe something that is considered a standard that all code (old and new) should follow and that is not expected to change substantially over time. Principles in the canonical documents should be understood by authors and reviewers alike, so everything included within a canonical document must meet a high bar. As such, canonical documents are generally shorter and prescribe fewer elements of style than non-canonical documents.

https://google.github.io/styleguide/go#canonical

- **Normative**: Intended to establish consistency

  Within these documents, “normative” is used to describe something that is an agreed-upon element of style for use by Go code reviewers, in order that the suggestions, terminology, and justifications are consistent. These elements may change over time, and these documents will reflect such changes so that reviewers can remain consistent and up-to-date. Authors of Go code are not expected to be familiar with the normative documents, but the documents will frequently be used as a reference by reviewers in readability reviews.

  https://google.github.io/styleguide/go#normative

- **Idiomatic**: Common and familiar

  Within these documents, “idiomatic” is used to refer to something that is prevalent in Go code and has become a familiar pattern that is easy to recognize. In general, an idiomatic pattern should be preferred to something unidiomatic if both serve the same purpose in context, as this is what will be the most familiar to readers.

  https://google.github.io/styleguide/go#idiomatic

## Additional references

This guide assumes the reader is familiar with Effective Go, as it provides a common baseline for Go code across the entire Go community.

Below are some additional resources for those looking to self-educate about Go style and for reviewers looking to provide further linkable context in their reviews. Participants in the Go readability process are not expected to be familiar with these resources, but they may arise as context in readability reviews.

**External References**

- Go Language Specification
- Go FAQ
- Go Memory Model
- Go Data Structures
- Go Interfaces

- Go Proverbs

- Go Tip Episodes - stay tuned.

- Unit Testing Practices - stay tuned.

### Relevant Testing-on-the-Toilet articles

- TotT: Identifier Naming
- TotT: Testing State vs. Testing Interactions
- TotT: Effective Testing
- TotT: Risk-driven Testing
- TotT: Change-detector Tests Considered Harmful

### Additional External Writings

- Go and Dogma
- Less is exponentially more
- Esmerelda’s Imagination
- Regular expressions for parsing
- Gofmt’s style is no one’s favorite, yet Gofmt is everyone’s favorite (YouTube)