# **Golang**

* [Go Code Review](https://github.com/golang/go/wiki/CodeReviewComments)

## **Getting started**

* [How to write Go code](https://golang.org/doc/code.html)
* [Effective Go](https://golang.org/doc/effective_go.html)
* [The Go Programming Language Specification](https://golang.org/ref/spec)
* [Frequently Asked Questions (FAQ)](https://golang.org/doc/faq)
* [Go at Google: Language Design in the Service of Software Engineering](https://talks.golang.org/2012/splash.article)
* [A Tour of Go](https://tour.golang.org/)
* [Go By Examples](https://gobyexample.com/)
* [The Go Blog](https://blog.golang.org/)

## **Understanding Go**

* [Go Details](https://go101.org/article/details.html)
* [Constants](https://blog.golang.org/constants)
* [Strings, bytes, runes and characters in Go](https://blog.golang.org/strings)
* [Arrays, slices (and strings): The mechanics of 'append'](https://blog.golang.org/slices)
* [Go Slices: usage and internals](https://blog.golang.org/go-slices-usage-and-internals)
* [Go maps in action](https://blog.golang.org/go-maps-in-action)
* [Arrays, Slices And Maps In Go](https://go101.org/article/container.html)
* [Structs In Go](https://go101.org/article/struct.html)
* [Pointers In Go](https://go101.org/article/pointer.html)
* [Functions In Go](https://go101.org/article/function.html)
* [Channels In Go](https://go101.org/article/channel.html)
* [Interfaces](https://research.swtch.com/interfaces)
* [Interfaces In Go](https://go101.org/article/interface.html)
* [Go Data Structures](https://research.swtch.com/godata)
* [The Laws of Reflection](https://blog.golang.org/laws-of-reflection)
* [Defer, Panic, and Recover](https://blog.golang.org/defer-panic-and-recover)
* [Broken abstractions in Go](https://research.swtch.com/goabstract)
* [Concurrency Is Not Parallelism](https://youtu.be/cN_DpYBzKso)
* [Share Memory By Communicating](https://blog.golang.org/share-memory-by-communicating)
* [The Go Memory Model](https://golang.org/ref/mem)
* [Go Execution Modes](https://docs.google.com/document/d/1nr-TQHw_er6GOQRsF6T43GGhFDelrAP0NqSS_00RgZQ/edit?pli=1)
* [The Go scheduler](https://morsmachine.dk/go-scheduler)
* [Scalable Go Scheduler Design Doc](https://docs.google.com/document/d/1TTj4T2JO42uD5ID9e89oa0sLKhJYD0Y_kqxDv3I3XMw/edit" \l "heading=h.mmq8lm48qfcw)
* [Scheduling In Go](https://www.ardanlabs.com/blog/2018/08/scheduling-in-go-part1.html)
* [The Scheduler Saga](https://www.youtube.com/watch?v=YHRO5WQGh0k)

## **HowTo**

* [Organizing Go code](https://blog.golang.org/organizing-go-code)
* [Error handling and Go](https://blog.golang.org/error-handling-and-go)
* [Godoc: documenting Go code](https://blog.golang.org/godoc-documenting-go-code)
* [Profiling Go Programs](https://blog.golang.org/profiling-go-programs)
* [Introducing the Go Race Detector](https://blog.golang.org/race-detector)
* [Go Concurrency Patterns: Pipelines and cancellation](https://blog.golang.org/pipelines)
* [Go Concurrency Patterns: Context](https://blog.golang.org/context)
* [Advanced Go Concurrency Patterns](https://blog.golang.org/advanced-go-concurrency-patterns)
* [A Good Makefile for Go](https://azer.bike/journal/a-good-makefile-for-go/)
* [Deploying Go servers with Docker](https://blog.golang.org/docker)

## **Tips, Pitfalls**

* [Go Tips](https://go101.org/article/tips.html)