The Go Blog

Go 1.20 is released!

Robert Griesemer, on behalf of the Go team
1 February 2023

Today the Go team is thrilled to release Go 1.20, which you can get by visiting the download page.

Go 1.20 benefited from an extended development phase, made possible by earlier broad testing and improved overall stability of the code base.

We're particularly excited to launch a preview of profile-guided optimization (PGO), which enables the compiler to perform application- and workload-specific optimizations based on run-time profile information. Providing a profile to `go build` enables the compiler to speed up typical applications by around 3–4%, and we expect future releases to benefit even more from PGO. Since this is a preview release of PGO support, we encourage folks to try it out, but there are still rough edges which may preclude production use.

Go 1.20 also includes a handful of language changes, many improvements to tooling and the library, and better overall performance.

Language changes

- The predeclared `comparable` constraint is now also satisfied by ordinary comparable types, such as interfaces, which will simplify generic code.
- The functions `SliceData`, `String`, and `StringData` have been added to package `unsafe`. They complete the set of functions for implementation-independent slice and string manipulation.
- Go's type conversion rules have been extended to permit direct conversion from a slice to an array.
- The language specification now defines the exact order in which array elements and struct fields are compared. This clarifies what happens in case of panics during comparisons.

Tool improvements

- The `cover` tool now can collect coverage profiles of whole programs, not just of unit tests.
- The `go` tool no longer relies on pre-compiled standard library package archives in the `$GOROOT/pkg` directory, and they are no longer shipped with the distribution, resulting in smaller downloads. Instead, packages in the standard library are built as needed and cached in the build cache, like other packages.
- The implementation of `go test -json` has been improved to make it more robust in the presence of stray writes to `stdout`.

- The `go build`, `go install`, and other build-related commands now accept a `-pgo` flag enabling profile-guided optimizations as well as a `-cover` flag for whole-program coverage analysis.
- The `go` command now disables `cgo` by default on systems without a C toolchain. Consequently, when Go is installed on a system without a C compiler, it will now use pure Go builds for packages in the standard library that optionally use cgo, instead of using pre-distributed package archives (which have been removed, as noted above).
- The `vet` tool reports more loop variable reference mistakes that may occur in tests running in parallel.

Standard library additions

- The new `crypto/ecdh` package provides explicit support for Elliptic Curve Diffie-Hellman key exchanges over NIST curves and Curve25519.
- The new function `errors.Join` returns an error wrapping a list of errors which may be obtained again if the error type implements the `Unwrap() []error` method.
- The new `http.ResponseController` type provides access to extended per-request functionality not handled by the `http.ResponseWriter` interface.
- The `httputil.ReverseProxy` forwarding proxy includes a new `Rewrite` hook function, superseding the previous `Director` hook.
- The new `context.WithCancelCause` function provides a way to cancel a context with a given error. That error can be retrieved by calling the new `context.Cause` function.
- The new `os/exec.Cmd` fields `Cancel` and `WaitDelay` specify the behavior of the `Cmd` when its associated `Context` is canceled or its process exits.

Improved performance

- Compiler and garbage collector improvements have reduced memory overhead and improved overall CPU performance by up to 2%.
- Work specifically targeting compilation times led to build improvements by up to 10%. This brings build speeds back in line with Go 1.17.

When building a Go release from source, Go 1.20 requires a Go 1.17.13 or newer release. In the future, we plan to move the bootstrap toolchain forward approximately once a year. Also, starting with Go 1.21, some older operating systems will no longer be supported: this includes Windows 7, 8, Server 2008 and Server 2012, macOS 10.13 High Sierra, and 10.14 Mojave. On the other hand, Go 1.20 adds experimental support for FreeBSD on RISC-V.

For a complete and more detailed list of all changes see the full release notes.

Thanks to everyone who contributed to this release by writing code, filing bugs, sharing feedback, and testing the release candidates. Your efforts helped to ensure that Go 1.20 is as stable as possible. As always, if you notice any problems, please file an issue.

Enjoy Go 1.20!

Previous article: Share your feedback about developing with Go
Blog Index