Go 1.25

@peterhellberg

There are no language changes that affect Go programs in ***Go 1.25****.*

However, in the language specification the notion of core types has been removed in favor of dedicated prose.

Tools

GO command

Go command

The `go build -asan` option now defaults to doing leak detection at program exit.

*This will report an error if memory allocated by **C** is not freed and is not referenced by any other memory allocated by either C or Go.*

Go command

The Go distribution will include fewer prebuilt tool binaries.

Core toolchain binaries such as the compiler and linker will still be included, but tools not invoked by ***build*** *or* ***test*** *operations will be built and run by* `go tool` *as needed.*

Go command

The new **`go.mod` ignore** directive can be used to specify directories the go command should ignore.

Files in these directories and their subdirectories will be ignored by the go command when matching package patterns, such as all or `./...`, but will still be included in module zip files.

Go command

The new `go doc -http` option will start a documentation server showing documentation for the requested object, and open the documentation in a browser window.

Go command

The new `go version -m -json` option will print the **JSON** encodings of the `runtime/debug.BuildInfo` structures embedded in the given Go binary files.

Go command

The `go` command now supports using a subdirectory of a repository as the path for a module root

when resolving a module path using the syntax ***`<meta name="go-import" content="root-path vcs repo-url subdir">`*** *to indicate that the root-path corresponds to the subdir of the repo-url with version control system vcs.*

Go command

The new **`work`** package **pattern** matches all packages in the work *(formerly called **`main`**)* modules:

- *either the single work module in module mode*
- *or the set of workspace modules in workspace mode*

Go command

When the `go` command updates the **`go`** line in a **`go.mod`** or **`go.work`** file, it no longer adds a **`toolchain`** line specifying the command's current version.


Vet

Vet

The `go vet` command includes new analyzers:

- ***waitgroup***
  - *Reports misplaced calls to* ***`sync.WaitGroup.Add`***
- ***hostport***
  - *Reports uses of* ***`fmt.Sprintf("%s:%d", host, port)`*** *to construct addresses for* ***`net.Dial`****, as these will not work with IPv6; instead it suggests using* ***`net.JoinHostPort`****.*

Runtime

Container-aware GOMAXPROCS

The default behavior of the `GOMAXPROCS` has changed.

In prior versions of **Go**, `GOMAXPROCS` defaults to the number of logical CPUs available at startup (`runtime.NumCPU`).

1. On Linux, the runtime considers the CPU bandwidth limit of the cgroup containing the process, if any.
2. On all OSes, the runtime periodically updates `GOMAXPROCS` if the number of logical CPUs available or the cgroup CPU bandwidth limit change.

New experimental garbage collector

A new garbage collector is now available as an **experiment**.

The new garbage collector may be enabled by setting `GOEXPERIMENT=greenteagc` at build time. We expect the design to continue to evolve and improve.

Trace flight recorder

Runtime execution traces have long provided a powerful, but **expensive** way to understand and debug the low-level behavior of an application.

The new **`runtime/trace.FlightRecorder`** API provides a lightweight way to capture a runtime execution trace by continuously recording the trace into an in-memory ring buffer.

Change to unhandled panic output

The message printed when a program exits due to an **unhandled** panic that was recovered and repanicked no longer repeats the text of the panic value.

Compiler

nil pointer bug

This release fixes a compiler bug, introduced in **Go 1.21**, that could incorrectly delay nil pointer checks.

The following program is incorrect because it uses the result of **`os.Open`** before checking the error.

nil pointer bug

```
package main

import "os"

func main() {
    f, err := os.Open("nonExistentFile")
    name := f.Name()
    if err != nil {
        return
    }
    println(name)
}
```

DWARF5 support

The compiler and linker in **Go 1.25** now generate debug information using **DWARF version 5**.

The newer DWARF version reduces the space required for debugging information in Go binaries, and reduces the time for linking, especially for large Go binaries.

Can be disabled by setting the environment variable `GOEXPERIMENT=nodwarf5` at build time.

Faster slices

The compiler can now allocate the backing store for slices on the stack in more situations, which **improves** performance.

This change has the *potential* to amplify the effects of incorrect **`unsafe.Pointer`** usage.

Linker

Linker

The linker now accepts a `-funcalign=N` command line option, which specifies the alignment of function entries.

The default value is platform-dependent, and is **unchanged** in this release.

Standard library

New testing/synctest package

The new `testing/synctest` package provides support for testing concurrent code:

- **`Test`** function runs a test function in an isolated “bubble”.
  - Within the bubble, **`time`** is virtualized.
- **`Wait`** function waits for all goroutines in the current bubble to block.

New experimental encoding/json/v2 package

Go 1.25 includes a new, **experimental** JSON implementation, which can be enabled by setting the environment variable `GOEXPERIMENT=jsonv2` at build time.

New experimental encoding/json/v2 package

When enabled, two new packages are available:

- **`encoding/json/v2`**
  - A major revision of the **`encoding/json`** package.
- **`encoding/json/jsontext`**
  - Provides lower-level processing of JSON syntax.

New experimental encoding/json/v2 package

In addition, when the “jsonv2” `GOEXPERIMENT` is enabled:

- **`encoding/json`**
  - uses the new JSON implementation.

    Marshaling and unmarshaling behavior is unaffected, but the text of errors returned by package function may change.
  - contains a number of new options which may be used to configure the **marshaler** and **unmarshaler**.

Minor changes to the library

As usual, there has been **minor** changes to a number of standard library packages, those packages are:

archive/tar encoding/asn1 crypto crypto/ecdsa crypto/ed25519 crypto/elliptic crypto/rsa crypto/sha1 crypto/sha3 crypto/tls crypto/x509 debug/elf go/ast go/parser go/token go/types hash hash/maphash io/fs log/slog mime/multipart net net/http os reflect regexp/syntax runtime runtime/pprof sync testing testing/fstest unicode unique

Ports

Ports

- **Darwin**: Go 1.25 requires macOS 12 Monterey or later.
- **Windows**: Go 1.25 is the last release that contains the broken 32-bit windows/arm port (**`GOOS=windows GOARCH=arm`**).
- **Loong64**: Now supports:
  - The race detector
  - Gathering traceback information from C code using **`runtime.SetCgoTraceback`**
  - Linking CGo programs with the internal link mode
- **RISC-V**: Now supports the **`plugin`** build mode.

Learn more

Please read the Go 1.25 Release Notes

https://go.dev/doc/go1.25