Go 1.28 Release Notes

- Go 1.28 Release Notes
  - DRAFT RELEASE NOTES — Introduction to Go 1.28
  - Tools
    - Vet
  - Bootstrap
  - Standard library
    - Minor changes to the library
      - crypto/x509
      - debug/pe
      - embed
      - encoding/base32
      - encoding/base64
      - flag
      - go/build
      - go/parser
      - go/types
      - log/slog
      - math
      - net
      - net/http
      - net/url
      - regexp
      - strconv
      - syscall
      - testing/fstest
      - testing/synctest
      - time

Go 1.28 Release Notes

DRAFT RELEASE NOTES — Introduction to Go 1.28

Go 1.28 is not yet released. These are work-in-progress release notes. Go 1.28 is expected to be released in February 2027.

Tools

Vet

The new `scannererr` analyzer checks for failure to handle scanner errors after a loop around `bufio.Scanner.Scan`, which may cause scanning or I/O errors to go unreported.

The `sqlrowserr` analyzer performs a similar check for loops around `sql.Rows.Next`, so that iteration errors are correctly distinguished from a smaller result.

Bootstrap

As mentioned in the Go 1.26 release notes, Go 1.28 now requires Go 1.26.0 or later for bootstrap. We expect that Go 1.30 will require a minor release of Go 1.28.0 or later for bootstrap.

Standard library

Minor changes to the library

crypto/x509

`ParseCertificate`, `ParseCertificateRequest`, `ParseRevocationList` and `ParsePKIXPublicKey` now reject trailing data after the parameters of an AlgorithmIdentifier, after the RSAPublicKey structure of an RSA public key and after its public exponent, and after the named curve of an ECDSA public key.

`ParseCertificate` and `ParseCertificateRequest` now reject trailing data after the value of the key usage, basic constraints, subject alternative name, extended key usage and certificate policies extensions.

debug/pe

The `ImportDirectory` type is now marked as deprecated. It was exported by accident, is an internal detail of the package, and should not be used directly.

embed

The `ReadAt` methods of files returned by `embed.FS` now return `io.EOF` when called with an offset at or beyond the end of the file, as required by `io.ReaderAt`.

encoding/base32

`Encoding.EncodedLen` now panics if the encoded length overflows int, rather than returning a negative number.

encoding/base64

`Encoding.EncodedLen` now panics if the encoded length overflows int, rather than returning a negative number.

flag

The new `FlagSet.All` method returns an iterator over all the flags in the FlagSet; the function `All` does the same for the global `CommandLine` flag set. The new `Flag.IsSet` field indicates whether the flag has been set.

go/build

The `build.ArchChar` function is deprecated, as it no longer varies by architecture.

go/parser

The `ResolveFile` function performs legacy syntactic identifier resolution on an `ast.File` syntax tree. This resolution is deprecated; new applications should instead use `go/types`. This function is provided to ease the transition away from the old resolution in applications that mostly parse files in `SkipObjectResolution` mode but later encounter some cases where the old resolution is needed.

go/types

Added `NewTypeList` and `NewTypeParamList` constructors to create `TypeList` and `TypeParamList` instances from slices of their respective element types.

The new `Scope.Objects` method returns the sequence of Objects in the Scope.

log/slog

In Go 1.27.1, direct calls to `Handler.WithGroup` that passed `""`, including those inside `MultiHandler`, would erroneously insert a group with name `""` instead of doing nothing. This bug is now fixed.

math

The new `FMA32` function computes x*y + z for float32 values with only one rounding. It uses a hardware fused multiply-add instruction where available.

net

The `IPMask`, `IPNet`, and `HardwareAddr` types each now implement a `encoding.TextUnmarshaler` method. This method will recognize the current marshaling of those types, as well as a planned future marshaling format. The future marshaling format will be readable in JSON format, unlike the current format which is a base64 encoding. The future marshaling format will become the default in a later Go release. This is being done as a two-step process so that JSON generated by this release of Go can still be read by earlier releases of Go.

The unused `DNSConfigError` type is deprecated.

net/http

The long-deprecated `Transport.CancelRequest` now does nothing. Use `NewRequestWithContext` (available since Go 1.13) to create a cancellable request.

`ProxyFromEnvironment` now prioritizes lowercase proxy environment variables (`http_proxy`, `https_proxy`, and `no_proxy`) over the uppercase versions if both are set to different values. This change also affects default proxy lookups in `DefaultTransport` and `DefaultClient` by extension.

The new `MethodQuery` constant represents the HTTP QUERY method defined in RFC 10008.

The HTTP/2 server now lets a connection's serve goroutine exit while the connection has nothing to do, restarting it on demand. This reduces memory usage for servers with many idle connections or many handlers idling mid-response (such as long polls).

net/url

The new `MustParse` function returns the parsed `URL` or panics on error. It can simplify some common uses of `Parse`. For instance, initializing variables with valid URL string constants.

regexp

New `Regexp.All`, `Regexp.AllIndex`, `Regexp.AllString`, `Regexp.AllStringIndex`, `Regexp.AllStringIndex`, `Regexp.AllStringSubmatchIndex`, `Regexp.AllSubmatch`, and `Regexp.AllSubmatchIndex` iterator methods on `Regexp` allow incremental matching of the input.

The new `Regexp.SplitSeq` method iterates over the results of splitting the input with the regular expression.

strconv

When `Unquote` is applied to a string containing a single-quoted rune literal, it now correctly returns the rune's UTF-8 encoding, even when the literal uses a hexadecimal or octal escape, as in `Unquote(`'\xff'`)`. (Previously it would return a string whose sole byte had the same numeric value as the rune, which is incorrect for non-ASCII.)

syscall

On Linux, `SysProcAttr` now supports enforcement of Landlock policies and the `PR_SET_NO_NEW_PRIVS` flag for newly spawned subprocesses.

testing/fstest

The `ReadAt` methods of files returned by `testing/fstest.MapFile.Open` now return `io.EOF` when called with an offset at or beyond the end of the file, as required by `io.ReaderAt`.

testing/synctest

The new `Subtest` convenience function combines [testing.(*T).Run] and `testing/synctest.Test`.

time

On Windows, the monotonic clock used by `Now` now has a resolution of 100 nanoseconds. Previously, its resolution was on the order of milliseconds. This improves the precision of elapsed-time measurements, such as those made with `Since`.

□□□ #3
□ □□□ □□□ 2026-10-26 20:26:29 UTC
□ cocobook-sync □□□ 2026-11-05 04:08:25 UTC