Changelog

1.19.0 (9 Aug 2021)

Enhancements:

- #975: Avoid panicking in Sampler core if the level is out of bounds.
- #984: Reduce the size of BufferedWriteSyncer by aligning the fields better.

Thanks to @lancoLiu and @thockin for their contributions to this release.

1.18.1 (28 Jun 2021)

Bugfixes:

- #974: Fix nil dereference in logger constructed by `zap.NewNop`.

1.18.0 (28 Jun 2021)

Enhancements:

- #961: Add `zapcore.BufferedWriteSyncer`, a new `WriteSyncer` that buffers messages in-memory and flushes them periodically.
- #971: Add `zapio.Writer` to use a Zap logger as an `io.Writer`.
- #897: Add `zap.WithClock` option to control the source of time via the new `zapcore.Clock` interface.
- #949: Avoid panicking in `zap.SugaredLogger` when arguments of `*w` methods don't match expectations.
- #943: Add support for filtering by level or arbitrary matcher function to `zaptest/observer`.
- #691: Comply with `io.StringWriter` and `io.ByteWriter` in Zap's `buffer.Buffer`.

Thanks to @atrn0, @ernado, @heyanfu, @hnlq715, @zchee for their contributions to this release.

1.17.0 (25 May 2021)

Bugfixes:

- #867: Encode `<nil>` for nil `error` instead of a panic.
- #931, #936: Update minimum version constraints to address vulnerabilities in dependencies.

Enhancements:

- #865: Improve alignment of fields of the Logger struct, reducing its size from 96 to 80 bytes.
- #881: Support `grpclog.LoggerV2` in zapgrpc.
- #903: Support URL-encoded POST requests to the AtomicLevel HTTP handler with the `application/x-www-form-urlencoded` content type.
- #912: Support multi-field encoding with `zap.Inline`.
- #913: Speed up SugaredLogger for calls with a single string.
- #928: Add support for filtering by field name to `zaptest/observer`.

Thanks to @ash2k, @FMLS, @jimmystewpot, @Oncilla, @tsoslow, @tylitianrui, @withshubh, and @wziww for their contributions to this release.

1.16.0 (1 Sep 2020)

Bugfixes:

- #828: Fix missing newline in IncreaseLevel error messages.
- #835: Fix panic in JSON encoder when encoding times or durations without specifying a time or duration encoder.
- #843: Honor CallerSkip when taking stack traces.
- #862: Fix the default file permissions to use `0666` and rely on the umask instead.
- #854: Encode `<nil>` for nil `Stringer` instead of a panic error log.

Enhancements:

- #629: Added `zapcore.TimeEncoderOfLayout` to easily create time encoders for custom layouts.
- #697: Added support for a configurable delimiter in the console encoder.
- #852: Optimize console encoder by pooling the underlying JSON encoder.
- #844: Add ability to include the calling function as part of logs.
- #843: Add `StackSkip` for including truncated stacks as a field.
- #861: Add options to customize Fatal behaviour for better testability.

Thanks to @SteelPhase, @tmshn, @lixingwang, @wyxloading, @moul, @segevfiner, @andy-retailnext and @jcorbin for their contributions to this release.

1.15.0 (23 Apr 2020)

Bugfixes:

- #804: Fix handling of `Time` values out of `UnixNano` range.
- #812: Fix `IncreaseLevel` being reset after a call to `With` .

Enhancements:

- #806: Add `WithCaller` option to supersede the `AddCaller` option. This allows disabling annotation of log entries with caller information if previously enabled with `AddCaller` .
- #813: Deprecate `NewSampler` constructor in favor of `NewSamplerWithOptions` which supports a `SamplerHook` option. This option adds support for monitoring sampling decisions through a hook.

Thanks to @danielbprice for their contributions to this release.

1.14.1 (14 Mar 2020)

Bugfixes:

- #791: Fix panic on attempting to build a logger with an invalid Config.
- #795: Vendoring Zap with `go mod vendor` no longer includes Zap's development-time dependencies.
- #799: Fix issue introduced in 1.14.0 that caused invalid JSON output to be generated for arrays of `time.Time` objects when using string-based time formats.

Thanks to @YashishDua for their contributions to this release.

1.14.0 (20 Feb 2020)

Enhancements:

- #771: Optimize calls for disabled log levels.
- #773: Add millisecond duration encoder.
- #775: Add option to increase the level of a logger.
- #786: Optimize time formatters using `Time.AppendFormat` where possible.

Thanks to @caibirdme for their contributions to this release.

1.13.0 (13 Nov 2019)

Enhancements:

- #758: Add `Intp` , `Stringp` , and other similar `*p` field constructors to log pointers to primitives with support for `nil` values.

Thanks to @jbizzle for their contributions to this release.

1.12.0 (29 Oct 2019)

Enhancements:

- #751: Migrate to Go modules.

1.11.0 (21 Oct 2019)

Enhancements:

- #725: Add `zapcore.OmitKey` to omit keys in an `EncoderConfig` .
- #736: Add `RFC3339` and `RFC3339Nano` time encoders.

Thanks to @juicemia, @uhthomas for their contributions to this release.

1.10.0 (29 Apr 2019)

Bugfixes:

- #657: Fix `MapObjectEncoder.AppendByteString` not adding value as a string.
- #706: Fix incorrect call depth to determine caller in Go 1.12.

Enhancements:

- #610: Add `zaptest.WrapOptions` to wrap `zap.Option` for creating test loggers.
- #675: Don't panic when encoding a String field.
- #704: Disable HTML escaping for JSON objects encoded using the reflect-based encoder.

Thanks to @iaroslav-ciupin, @lelenanam, @joa, @NWilson for their contributions to this release.

v1.9.1 (06 Aug 2018)

Bugfixes:

- #614: MapObjectEncoder should not ignore empty slices.

v1.9.0 (19 Jul 2018)

Enhancements:

- #602: Reduce number of allocations when logging with reflection.
- #572, #606: Expose a registry for third-party logging sinks.

Thanks to @nfarah86, @AlekSi, @JeanMertz, @philippgille, @etsangsplk, and @dimroc for their contributions to this release.

v1.8.0 (13 Apr 2018)

Enhancements:

- #508: Make log level configurable when redirecting the standard library's logger.
- #518: Add a logger that writes to a `*testing.TB` .
- #577: Add a top-level alias for `zapcore.Field` to clean up GoDoc.

Bugfixes:

- #574: Add a missing import comment to `go.uber.org/zap/buffer` .

Thanks to @DiSiqueira and @djui for their contributions to this release.

v1.7.1 (25 Sep 2017)

Bugfixes:

- #504: Store strings when using AddByteString with the map encoder.

v1.7.0 (21 Sep 2017)

Enhancements:

- #487: Add `NewStdLogAt` , which extends `NewStdLog` by allowing the user to specify the level of the logged messages.

v1.6.0 (30 Aug 2017)

Enhancements:

- #491: Omit zap stack frames from stacktraces.
- #490: Add a `ContextMap` method to observer logs for simpler field validation in tests.

v1.5.0 (22 Jul 2017)

Enhancements:

- #460 and #470: Support errors produced by `go.uber.org/multierr` .
- #465: Support user-supplied encoders for logger names.

Bugfixes:

- #477: Fix a bug that incorrectly truncated deep stacktraces.

Thanks to @richard-tunein and @pavius for their contributions to this release.

v1.4.1 (08 Jun 2017)

This release fixes two bugs.

Bugfixes:

- #435: Support a variety of case conventions when unmarshaling levels.
- #444: Fix a panic in the observer.

v1.4.0 (12 May 2017)

This release adds a few small features and is fully backward-compatible.

Enhancements:

- #424: Add a `LineEnding` field to `EncoderConfig`, allowing users to override the Unix-style default.
- #425: Preserve time zones when logging times.
- #431: Make `zap.AtomicLevel` implement `fmt.Stringer`, which makes a variety of operations a bit simpler.

v1.3.0 (25 Apr 2017)

This release adds an enhancement to zap's testing helpers as well as the ability to marshal an AtomicLevel. It is fully backward-compatible.

Enhancements:

- #415: Add a substring-filtering helper to zap's observer. This is particularly useful when testing the `SugaredLogger`.
- #416: Make `AtomicLevel` implement `encoding.TextMarshaler`.

v1.2.0 (13 Apr 2017)

This release adds a gRPC compatibility wrapper. It is fully backward-compatible.

Enhancements:

- #402: Add a `zapgrpc` package that wraps zap's Logger and implements `grpclog.Logger`.

v1.1.0 (31 Mar 2017)

This release fixes two bugs and adds some enhancements to zap's testing helpers. It is fully backward-compatible.

Bugfixes:

- #385: Fix caller path trimming on Windows.
- #396: Fix a panic when attempting to use non-existent directories with zap's configuration struct.

Enhancements:

- #386: Add filtering helpers to zaptest's observing logger.

Thanks to @moitias for contributing to this release.

v1.0.0 (14 Mar 2017)

This is zap's first stable release. All exported APIs are now final, and no further breaking changes will be made in the 1.x release series. Anyone using a semver-aware dependency manager should now pin to `^1`.

Breaking changes:

- #366: Add byte-oriented APIs to encoders to log UTF-8 encoded text without casting from `[]byte` to `string`.
- #364: To support buffering outputs, add `Sync` methods to `zapcore.Core`, `zap.Logger`, and `zap.SugaredLogger`.
- #371: Rename the `testutils` package to `zaptest`, which is less likely to clash with other testing helpers.

Bugfixes:

- #362: Make the ISO8601 time formatters fixed-width, which is friendlier for tab-separated console output.
- #369: Remove the automatic locks in `zapcore.NewCore`, which allows zap to work with concurrency-safe `WriteSyncer` implementations.
- #347: Stop reporting errors when trying to `fsync` standard out on Linux systems.
- #373: Report the correct caller from zap's standard library interoperability wrappers.

Enhancements:

- #348: Add a registry allowing third-party encodings to work with zap's built-in `Config`.
- #327: Make the representation of logger callers configurable (like times, levels, and durations).
- #376: Allow third-party encoders to use their own buffer pools, which removes the last performance advantage that zap's encoders have over plugins.
- #346: Add `CombineWriteSyncers`, a convenience function to tee multiple `WriteSyncer`s and lock the result.
- #365: Make zap's stacktraces compatible with mid-stack inlining (coming in Go 1.9).
- #372: Export zap's observing logger as `zaptest/observer`. This makes it easier for particularly punctilious users to unit test their application's logging.

Thanks to @suyash, @htrendev, @flisky, @Ulexus, and @skipor for their contributions to this release.

v1.0.0-rc.3 (7 Mar 2017)

This is the third release candidate for zap's stable release. There are no breaking changes.

Bugfixes:

- #339: Byte slices passed to `zap.Any` are now correctly treated as binary blobs rather than `[]uint8`.

Enhancements:

- #307: Users can opt into colored output for log levels.
- #353: In addition to hijacking the output of the standard library's package-global logging functions, users can now construct a zap-backed `log.Logger` instance.
- #311: Frames from common runtime functions and some of zap's internal machinery are now omitted from stacktraces.

Thanks to @ansel1 and @suyash for their contributions to this release.

v1.0.0-rc.2 (21 Feb 2017)

This is the second release candidate for zap's stable release. It includes two breaking changes.

Breaking changes:

- #316: Zap's global loggers are now fully concurrency-safe (previously, users had to ensure that `ReplaceGlobals` was called before the loggers were in use). However, they must now be accessed via the `L()` and `S()` functions. Users can update their projects with

```
gofmt -r "zap.L -> zap.L()" -w .
gofmt -r "zap.S -> zap.S()" -w .
```

- #309 and #317: RC1 was mistakenly shipped with invalid JSON and YAML struct tags on all config structs. This release fixes the tags and adds static analysis to prevent similar bugs in the future.

Bugfixes:

- #321: Redirecting the standard library's `log` output now correctly reports the logger's caller.

Enhancements:

- #325 and #333: Zap now transparently supports non-standard, rich errors like those produced by `github.com/pkg/errors`.
- #326: Though `New(nil)` continues to return a no-op logger, `NewNop()` is now preferred. Users can update their projects with `gofmt -r 'zap.New(nil) -> zap.NewNop()' -w .`.
- #300: Incorrectly importing zap as `github.com/uber-go/zap` now returns a more informative error.

Thanks to @skipor and @chapsuk for their contributions to this release.

v1.0.0-rc.1 (14 Feb 2017)

This is the first release candidate for zap's stable release. There are multiple breaking changes and improvements from the pre-release version. Most notably:

- **Zap's import path is now "go.uber.org/zap"** — all users will need to update their code.
- User-facing types and functions remain in the `zap` package. Code relevant largely to extension authors is now in the `zapcore` package.
- The `zapcore.Core` type makes it easy for third-party packages to use zap's internals but provide a different user-facing API.
- `Logger` is now a concrete type instead of an interface.
- A less verbose (though slower) logging API is included by default.
- Package-global loggers `L` and `S` are included.
- A human-friendly console encoder is included.
- A declarative config struct allows common logger configurations to be managed as configuration instead of code.
- Sampling is more accurate, and doesn't depend on the standard library's shared timer heap.

v0.1.0-beta.1 (6 Feb 2017)

This is a minor version, tagged to allow users to pin to the pre-1.0 APIs and upgrade at their leisure. Since this is the first tagged release, there are no backward compatibility concerns and all functionality is new.

Early zap adopters should pin to the 0.1.x minor version until they're ready to upgrade to the upcoming stable release.