2019-05-28 - golang-tools session - meeting notes

Previous session notes

2019-04-30 - golang-tools session - meeting notes

Recording

https://youtu.be/qmDsGU0-s7Y

Attendees

- Jay Conrod
- Ian Cottrell
- Rob Findley
- Katie Hockman
- Paul Jolly
- Pontus Leitzler
- Muir Manders
- Daniel Marti
- Michael Matloob
- Bryan Mills
- Rebecca Stambler
- Marwan Sulaiman

Notes

Conference updates

- Michael and Rebecca spoke in Singapore. Rebecca, Katie, and Julie spoke in Japan.
- More talks at Gophercon in San Diego in July

Guest speaker: Katie Hockman

- Module mirror - proxy.golang.org
    - A caching server that sits between the go command running on your machine, and the origin server that hosts code (e.g. github)
    - Helps eliminate the risk of disappearing dependencies, and lowers storage and latency costs by allowing the go command to only request what it needs when fetching new code and resolving transitive dependencies.
- Sum database - sum.golang.org
    - Need a way of verifying that proxy server (or origin server) is giving you the same thing as it gives everyone else.
    - In the past, we relied on go.sum files. Was useful for cross-checking with authors of a module, but you still have to trust first fetch.
    - Sum database is a global database of go.sum entries.

    - Using a Merkle tree, also known as a certificate transparency log, to ensure that sumdb is an append-only log. Will be very obvious to auditors if the log is ever tampered with. Built this using Trillian.
- Module index - index.golang.org
    - A feed of new module versions that are newly available in proxy.golang.org.
- Paul: anything users have to do for sumdb?
    - Katie: no major change. It may catch things that haven't been caught before. Because of the mirror, you may see lower latency and less things in your cache.
    - Katie: There are some differences around private modules. Mirror and sumdb require modules are available on the public internet. You can run your own proxy, or you can go direct to the source in those cases by using GONOPROXY and GONOSUMDB. See the go command documentation at tip for details.
- Marwan: You can install your own proxy, and Athens fills that role. You'll need to set GONOSUMDB to ensure private modules don't leak. By default, Athens proxies the global sumdb (does not mirror; would love to eventually mirror and audit). Need to decide what to do for internal modules when GONOSUMDB is not configured.
- Paul: Is there documentation for private repos, and are you pointing to any implementation in particular?
- Katie: Will document better soon. Not pointing to any implementation yet. Russ has been working on a reference implementation of the proxy. It will end up in x/mod eventually.
- Katie: For the Merkle tree, we're using Trillian, which is open source. Anyone looking into mirroring the checksum database may consider that.
- Daniel: With 1.13, people will start dropping vendoring, what's the plan for companies and private modules?
    - Katie: Will make a note to document that.
    - Ian: We're discussing a lot internally.
    - Jay: The issue of partial vendoring has slipped to 1.14. We didn't reach a consensus on what we should do.
- Paul: Any hosting or managed options for Athens?
    - Marwan: Seems to work okay on AWS and GCP Cloud Run, but not automatic.
- Marwan: GOAUTH?
    - Bryan: Not going to make it into 1.13. Interacts with proxy fallback behavior, too much risk at this point in the cycle. netrc works a lot better though.
- Paul: Is the golang-dev thread where proxy.golang.org was announced a good place to follow up?
    - Katie: Yes, also file issues prefixed with "proxy.golang.org:', "index.golang.org:", ...

gopls update

- Rebecca: Having trouble with hovers on vim.
    - Paul: vim only has a single line balloon, will soon have large multi-line (via new popup window support). Would be good to support single line for now, because

    timeline on popup support being finished is not exact, and people will need time to upgrade in any case.

- Rebecca: in VSCode, do you want diagnostics for all packages on the left or just open files?
    - Paul: If we get all of them, people can write filters in plugins. But maybe gopls should filter them for us?
- Paul: Saw some exciting CLs about fuzzy completion?
    - Muir: Deep completion will suggest completions within objects, not just symbols within the immediate scope. Still in discussion.
    - Muir: Fuzzy completion done on the server side, so all editors can benefit. Fuzzy and deep completion are separate, but they work together.
- Paul (Question from Dominic): what's the Go LSP stance on custom commands that need special editor support? I'd love to see things like keyify and structlayout be part of the LSP somehow.
    - Ian: Tools that aren't controversial and can be written in gopls should be. We've been trying to avoid launching external binaries, but we're also trying to avoid gopls depending on other repositories, and we haven't resolved the conflict. We have some ideas, but it's a large task.
- Paul: Build tags issue got more attention, and one using wire (https://github.com/golang/go/issues/29202#issuecomment-492510254)
    - Rebecca: For the time being, we'll just use the target os / arch, other files will be ignored. We'll try to come up with a better solution.
    - Ian: Wire uses a custom code generator, and the original / generated files are tagged opposite.
- Ian: I split things up so we have a view, session, and low-level cache (gopls concepts, not VSCode). You can have multiple views of the same session, maybe on different build tags. View is where we know about configuration, Session is where we know about workspace, opened files, edited files. No way to expose this to the editor yet, or put another way, editors will need to define UI/UX concepts that map onto views
- Ian: We currently have a lot of unit tests, but need integration tests. If anyone integrating gopls with an editor can provide integration tests, that would be helpful.

go/packages, go/analysis

- Michael: going to try to fix a lot of blocking bugs this week. Going to add support for ranges in go/analysis. Working on a proposal for suggested refactoring. If you have a critical bug that hasn't gotten attention, please reach out.
- Paul: Support for packages defined entirely in overlays?
- Michael: Looking at that.

cmd/go

- Per Roger: any likelihood of progress on the two main module-version-regression issues? As far as I'm concerned, they're all show-stopper issues because they make migrating to modules harder and more error-prone

    - https://github.com/golang/go/issues/27171
    - https://github.com/golang/go/issues/29262
    - https://github.com/golang/go/issues/32161
    - Bryan: Highlighted issues will depend on how big the fix is and where it needs to go. May not make it into 1.13. We're spread pretty thin, so if people have insights, let us know.

Gophercon

    - Paul: Organizing a session, not quite ready yet. Anyone keen to help out, please say!

Gofumpt

- Daniel: gofumpt (https://github.com/mvdan/gofumpt) is a project for experimenting with changes for gofmt. Not sure how to tie this into gopls yet.