# 2019-08-27 - golang-tools session - meeting notes

## Previous session notes

The two sessions at GopherCon San Diego

- https://docs.google.com/document/d/1-RVyttQ0ncjCpR_sRwizf-Ubedkr0Emwmk2LhnsUOmE/edit
- https://docs.google.com/document/d/1ZI_WqpLCB8DO6teJ3aBuXTeYD2iZZZlkDptmcY6Ja60/edit

## Details

15:30 UTC
https://meet.google.com/xuq-tcoc-dkp

## Agenda

- cmd/go
    - Vendor proposal https://github.com/golang/go/issues/33848
- Updates from GopherCon action items
    - gopls
        - Strawman proposal on what v1.0.0 is/isn't (i.e. extensions not in, for now)
        - Establishing list of "certified" editors/plugins
        - Documenting error reporting process
    - build tags
        - Enumerate and document required use cases for build tags: https://github.com/golang/go/issues/33389
    - Debugger support (DAP)
        - Understand state of debugger support in various editors, start process similar to gopls "certified" plugins
    - go/packages
        - Any outstanding problems with go/packages?
    - Discovery site update
        - Public update on discovery site
        - Trusted testers programme extended - sign up at https://golang.org/s/discovery-tester-signup
    - Refactoring
        - Ian/Rebecca to share their ideas on this
- gopls
    - Status update
    - Blockers for "v1.0.0"
    - Any developments/progress on extensions? (ref Slack chat)
    - Workspace + build tags support (ref https://github.com/myitcv/govim/issues/428#issuecomment-516803065)
    - Any update on "the big rewrite"? (Ian)
- go/packages, go/analysis
- Discovery site
- AOB
    - Proposal: move mature wiki content behind Gerrit - mvdan
    - gopls documentation CL: https://go-review.googlesource.com/c/tools/+/191741
    - Code generation, extensions and gopls
    - Delete file support in gopls
    - With Go 1.13, how are we flagging to people that they're now interacting with a proxy by default?

## Recording

https://youtu.be/OTKPu0kZ6sQ

## Notes [WIP]

- Attendees
  - Andy Linderman
  - Carmen Andoh
  - Daniel Marti
  - David Chase
  - Duco van Amstel
  - Edward Muller
  - Giovanni Bajo
  - Ilan Pillemer
  - Jeffrey Talbot Kowalczyk
  - Kris Brandow
  - Marwan Sulaiman
  - Muir Manders
  - Bryan Mills
  - Pontus Leitzler
  - Ian Cottrel
  - Rebecca Stambler
  - Michael Matloob
  - Julie Qiu
  - Rob Findlay
  - Peter Weinberger
  - … others?
- Vendor and modules proposal: https://github.com/golang/go/issues/33848
  - Background: previous proposal in 1.13
  - New proposal for 1.14:
    - gets rid of partial vendoring
    - gets rid of automatic updates to vendor directory (in response to user feedback)
    - more about verifying consistency than producing consistency, i.e. error out when things are not consistent
  - Details about how the proposal will be implemented are also included
  - CL with implementation upcoming which will allow people to try this out (especially in the context of gopls and other tools)
  - Giovanni asked a question about the "Dependency analysis" point in the proposal, specifically what analysis will not be supported? Bryan indicated the answer to this point is precisely known at this stage, that's something that will likely become clearer with an implementation CL
  - Process from here:
    - Part of the standard proposal process; already run by key stakeholders on Go team
    - Gather feedback from the wider Go community, especially once there is a CL so that people can try this out
    - Everyone is encouraged to provide feedback and ask questions in https://github.com/golang/go/issues/33848
    - FAQ and any changes to the proposal will be made in the linked issue; any communication about a working CL will also be posted there
- Updates from GopherCon tools sessions
  - gopls
    - https://github.com/golang/go/issues/33389 has been created to enumerate and document required use cases for build tags; Rebecca will summarise at some point
    - Milestone created for "blockers" pre gopls v1.0.0 (https://github.com/golang/go/issues?q=is%3Aopen+is%3Aissue+milestone%3A%22gopls

+v1.0%22), where gopls v1.0.0 means "a stable cut that we can announce to editor plugin authors"

            - People should flag any issues that they consider blockers
        - Rebecca has also spent time working with VSCode thinking about how to make the user experience better
        - Ian has been focussing on documentation; starting point is https://go-review.googlesource.com/c/tools/+/191741. This is also an experiment in moving some documentation out of the wiki (see Daniel's proposal later)
        - Ian has also been speaking to people about better debugger (Delve) integration; no specific conclusions for now
        - Ian's documentation is a starting point for the initial entry point for users wanting a Go-enabled editor, for "blessed" editor plugins, error reporting process etc
    - go/packages
        - The main feedback had been that the load modes were not specific enough, but this is a tough balance because more modes introduce more edge cases
        - So the consensus is to leave things as they are for now (people should shout if this is strawman is unacceptable for any reason)
        - Update from Jay (not on the call): there is elbow room to improve the performance of go list but it will require a lot of work hence the case will have to be compelling. The use case of gopls, for example, is not a sufficiently compelling case because we are not currently memory/performance bounded by the go list calls. If anyone does have a compelling cases they should raise it
        - Marwan mentioned that a couple of code generators have had issues in the migration to modules, and are now significantly slower than before. Marwan will raise any issues either in the thread or via Go issues. Ian remarked that cases where he has helped generally centred around people treating go/packages as a direct replacement for go/build or go/loader, and a simple refactor to "load everything up front" generally fixed most situations, rather than being a go list problem per se.
        - Michael to provide an update on all of the points raised in this thread later this week
    - Discovery site
        - Trusted tests launched at GopherCon
        - Feedback has been submitted via the anonymous feedback form since then
        - Updates posted to various GitHub issues in order to signpost people who might be looking for updates on the "story" that is go doc, godoc, godoc.org and the discovery site:
            - Umbrella issue for godoc, godoc.org module support: https://github.com/golang/go/issues/26827#issuecomment-521427618
            - Tracking issue for godoc module support: https://github.com/golang/go/issues/33655
            - Tracking issue for discovery site: https://github.com/golang/go/issues/33654
        - These tracking issues will be used for announcements, updates etc in the future
        - Main focus is still working towards public launch:
            - Performance
            - Module and directory views, e.g. what to show when the user navigates to the equivalent of https://godoc.org/golang.org/x/tools or https://godoc.org/go. The former is a module but not a package, the latter is an empty package
            - General cleanup, in particular around edge cases in the standard library
- gopls
    - v1.0.0 milestone
        - Main update above; everything critical at this point is marked as https://github.com/golang/go/issues?q=is%3Aopen+is%3Aissue+milestone%3A%22gopls+v1.0%22
        - People should shout if things (bugs, requirements) are not marked as v1.0.0 but in their opinion should be
        - Duco to follow up on the details of file watching (which is a v1.0.0 "blocker") and history within VSCode
    - gopls extensions

        - No progress (on discussions) since GopherCon
        - Confirmed that this is parked until post v1.0.0; nothing critical depends on this for v1.0.0, and nothing we need to change about v1.0.0 in order to design for it
    - "The big rewrite"
        - Flowing up the stack, Rebecca has taken up the reins on this
        - Types information is now being cached in this new structure, but cache invalidation part is not changed over yet. Part of this involves switching from go/token positions to LSP positions
        - All of this falls under the milestone of v1.0.0
- go/packages, go/analysis
    - Suggested fixes are no longer "really extremely experimental" so people are encouraged to try them out (link to Michael's talk here when it becomes available)
    - Variant of checker tool that lives in go/analysis now supports a -fix flag to apply all suggested fixes
        - Was added in https://github.com/golang/tools/commit/97f12d73768f0481c12570ed574cf04a8879b04b
        - Flag is declared in https://github.com/golang/tools/blob/master/go/analysis/internal/checker/checker.go#L64
        - Which means it is part of all checkers, but Michael was implying you use https://godoc.org/golang.org/x/tools/go/analysis/singlechecker to try it out
- Discovery site
    - Covered above
- AOB
    - Daniel - proposal to move mature wiki content behind a Gerrit CL
        - Link to draft proposal: https://docs.google.com/document/d/1PwVEOkWMPBxGY_Du8Uwms8jakti8fgyFFYpg9aloknE/edit
        - Looking for feedback: proposal is that more mature, well-established content that lives in the wiki be migrated to another home and changes to said content be guarded by a Gerrit CL. Goal being to protect such content an ultimately improve its quality by ensuring good reviews on changes
        - Plan to submit towards the end of the week (post coordinate with Carmen who has a related proposal)
        - Per Ian: the move of gopls documentation is in effect an experiment in this regard. Good because of code review, but also because contributions then require signing of the CLA which makes it possible to cleanly redistribute the docs
    - gopls documentation
        - Main update covered above
        - It's a start: feedback, suggestions, contributions all welcome
    - cmd/go 1.13 and module proxy interaction
        - Point submitted by thepudds: are we comfortable that with the release of 1.13 and the default "on" of the module proxy that error messages to users are clear and informative enough? Do we perhaps want to include a link in such messages to "see more details http://here.com?"
        - Per Bryan: situation with error reporting in 1.13 is no less useful than in 1.12 because in 1.13 there is the fallback https://proxy.golang.org,direct: hence the error messages that people see will likely be coming from the direct fallback. And those error messages are at least as useful as 1.12 (Jay and Bryan have made a number of improvements)
    - Prometheus modules migration
        - Duco has been speaking with folks from the Prometheus project about their difficulties with Go modules (context: https://groups.google.com/forum/#!topic/prometheus-developers/F1Vp0rLk3TQ).
        - They follow semver for their system-level APIs but make breaking changes between minors in their implementations libraries which many users also depend on.
        - Duco has been helping to talk through various options to bring the project in line with proper semver, focussing on keeping things as minimal as possible in order to maintain contribution velocity.

        - Duco is writing a proposal for a tool to help them with this, and potentially other larger projects in similar situations; will post a link to the golang-tools community looking for feedback, any other ideas etc
    - 1.14 tree open
        - Request from Daniel: if there are any major changes landing in the tree, please flag them to the golang-tools community where appropriate so we can get more eyes on them, people testing stuff out etc, e.g. the modules+vendor changes