2019-09-24 - golang-tools session - meeting notes

Previous session notes

- https://docs.google.com/document/d/1NiIbz1h4-UaavdL-SC2hTp54Y87p-1joaLa-r5HgKaE/edit#

Details

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

Agenda

- Welcome, especially those for whom this is their first call!
- /community proposal (https://docs.google.com/document/d/1dc3Gq2R4lCpvaB0skGqZU2-sGL1pkiesnOmX2O71n0M/edit)
    - add /community repo on github.com/golang
- gopls
    - Status update
    - Update on recent work with VSCode
    - Blockers for "v1.0.0" (https://github.com/golang/go/issues?q=is%3Aopen+is%3Aissue+milestone%3A%22gopls+v1.0%22)
    - Documentation update
        - How well is it working?
        - Has anyone seen it? Do they care?
    - Recent release
        - Huge thanks to Rebecca, Ian, Muir and everyone who contributed for getting it out the door!
    - New release(s)
        - what frequency should we be aiming for?
    - Current/next focus
        - Multiple concurrent gopls clients: https://github.com/golang/go/issues/34111. Significant for editors like Vim which start/stop regularly
- cmd/go
    - Status update
    - 1.14 - modules on by default, are we ready?
        - Major blockers: #32027 (behavior outside of a module), #26092 (svn support), #33848 (automatic use of vendor)
        - Some uncertainty: 1.14 cycle is short due to 1.13 delays.
    - proposal: cmd/go: ignore +incompatible versions as of Go 1.14 - https://github.com/golang/go/issues/34217
        - Background: +incompatible versions are a major pain for both module maintainers and users.
        - General idea: can we just make them go away, and replace them with pseudo-versions?
        - Specific proposal is probably too aggressive for 1.14 — we're looking at more incremental alternatives.
    - Quick update on the vendor proposal (https://github.com/golang/go/issues/33848)?
        - Still awaiting proposal review, but looking uncontentious so far — likely to start prototyping this week so that we can start getting feedback / experience on a draft CL.
        - Only open question right now is when exactly tro trigger `-mod=readonly` by default. (`go 1.14` in the `go.mod` file, or any version? Presence of needed annotations in `vendor/modules.txt`, or presence of a `vendor` directory at all? Leaning toward `go 1.14` + any vendor directory, as suggested by thepudds.)

    - Clarify comments on proxy not keeping all modules - https://twitter.com/_myitcv/status/1172136207440121856
    - Moving forward the gobin-like proposal: where do we go from here? https://github.com/golang/go/issues/33518#issuecomment-532643149
    - Is the "global install" issue https://github.com/golang/go/issues/30515 likely to happen for 1.14? Is any input needed?
    - #34506 - Considering new flags for manipulating go.mod files: -modfile, -sumfile
        - Intended to stop gopls from editing go.mod (via go list). It can copy go.mod to a temp directory and point to it with -modfile.
        - -modfile=none would cause go command to act as if no go.mod were present.
- go/packages, go/analysis
    - Status update
- Discovery site
    - Status update
    - godoc getting module support before the discovery site is public?
- AOB
    - Update on proposal to move mature wiki content behind Gerrit (https://github.com/golang/go/issues/34038)
    - Update on the modularise tool proposal to manage existing larger projects in the context of Go modules. (https://docs.google.com/document/d/1g7wIlxn9JBJkc-2lHAb2MfCYLltWZKyNIIO6SBW1x-8/edit#)

Next call

Tue 29 Oct 15:30 UTC

(note this date is straddled by the daylight savings changes in the UK and US, so this call will be at 15:30 in the UK)

Calendar invite already sent out

WIP agenda: https://docs.google.com/document/d/1F3T58Nj_Ft3bu15Wd4hAZAW6kLI_M1EH5XP43l_3CpY/edit#

Recording

https://youtu.be/E5w02B62oqc

Notes

- Carmen Andoh - /community proposal
    - Proposal is currently in draft form: https://docs.google.com/document/d/1dc3Gq2R4lCpvaB0skGqZU2-sGL1pkiesnOmX2O71n0M/edit
    - Motivation:
        - Wiki is huge, hard to manage, no review/control process
        - Not a natural home for many and various other resources like Go meetup organisers sharing best practices, documents, etc
        - No clear path to content becoming more mature etc
    - There have been a number of related proposals:
        - Move more mature wiki content behind Gerrit #34038
        - Restructure module documentation #33637
        - Move gopls docs behind Gerrit CL 191741
    - Plan is to submit this draft "soon" - it's been well circulated amongst various folks
    - Beyond the more specific proposals above is the idea that there is a lot more content that would more naturally sit under a /community repo, with owners of specific subparts of that repo

        - GDN
        - Go meetup organisers material
        - Tooling working group minutes, content etc
        - ...
    - Carmen would like feedback on the structure proposed in the proposal
    - Ultimately content in the /community repo can then be linked from golang.org as well as required
    - Jay:
        - Daniel previous suggested x/doc as a repo
        - Feedback from Andy and Dmitri is that documentation that is on the website should live under x/website
        - Should /community sit under x/website?
    - Carmen:
        - Benefits of this is that it's a single source of truth
        - If the gerrit bot can still make the PR workflow "work" here then great
        - We would need a separate repo altogether on GitHub if we only wanted the PR workflow
        - Having a separate /community "space" is important for drawing the distinction between what is in the official Go distribution vs the wider ecosystem, meetups network etc
    - Paul:
        - Does living under x/website suggest that all this material is "website" material?
        - Because meetup organisers material is more a random grab bag of stuff that is almost exclusively _not_ website material
    - Carmen:
        - Andy and Dmitri to offer thoughts on this later in the week
    - Paul:
        - No reason why tooling group meeting minutes, material etc can't move over. It's only lived in the wiki because that was the easiest home to start with
        - Google docs can easily become markdown files etc
    - Everyone is encouraged to offer feedback in the doc
    - Once Andy and Dmitri have responded, Carmen will create this as an official proposal
- gopls
    - Status updates
        - 2 release since the last call
        - Folding ranges and prepareRename support added by Suzy
        - Suggested fixes are now turned on by default
        - staticcheck support has been merged (but is not on by default)
            - This will likely bring lots of suggested fixes
        - Fixes added for completion after go and defer statements
        - You can run go mod tidy using a source.organizeImports code action on go.mod files (at master)
    - Rebecca has been focussing on fixing caching issues to iron out remaining race conditions
    - Rebecca has been trying to improve the integration with VSCode
        - Specifically VSCode users are now using @latest instead of master
        - More plans for the future like making it easier to file issues etc
    - Blockers for v1.0.0
        - Many of the items are in progress, but current focus is sorting caching issues; features will follow
        - Goal is to hit v1.0.0 by end of year
    - Ian:
        - Don't feel like we can turn modules on by default (in Go 1.14) until tooling is ready
        - Switching everyone to gopls is the right answer for that
        - So this should be the guiding principle behind deciding what goes into the v1.0.0 milestone
        - Hence if we want to turn on modules by default in Go 1.14, we have to have gopls v1.0.0 ready and tested in plenty of time
        - Timeline on plugins for other editors is a bit later (e.g. Vim, Emacs etc): the call on whether modules can be turned on by default is going to be made with reference to VSCode
    - Documentation update:

        - https://github.com/golang/tools/tree/master/gopls
        - Does anyone have any feedback on the move of docs from the wiki to the x/tools/gopls module?
        - Can we delete the wiki now, simply leaving a "redirect"? No objections
        - Is everyone happy with doing Gerrit CLs to update the docs? No objections
    - How frequently should we release gopls?
        - Rebecca has now made the change for VSCode to use @latest
        - Hence we can't really ask people to upgrade more regularly than once a month, say
        - Until we get to a point where VSCode will allow people to be upgraded without asking them, then the releases will need to be at a "sensible" pace
        - Plan for release process:
            - Update gopls go.mod file to refer to master in x/tools on a given day per month
            - We test this for a couple of weeks and only merge bug fixes
            - Then release
        - No objections this approach
    - Multiple concurrent gopls clients: https://github.com/golang/go/issues/34111
        - This is important for Vim/Emacs etc users who regularly have multiple instances of an editor open
        - With imports cache pre-warming, this would mean each instance needs to perform a massive pre-warm (see https://github.com/golang/go/issues/32750#issuecomment-528314462 for examples of ~40 sec cache warms)
        - Hence having a single instance of gopls would help
        - Billie would also like the gopls instance to be shareable between different Vim plugins
        - Ian:
            - implementing not too hard
            - discovery is a bigger question
            - how is lifetime managed?
        - Paul:
            - discovery rough first cut could follow same approach as gocode did, which presumably worked for people?
    - BIG THANK YOU to Rebecca, Ian, Muir and everyone for all the work that has gone into gopls over the last month
- cmd/go (Bryan)
    - Current focus fixing up a number of regressions in Go 1.13
    - Will modules be on by default in 1.14?
        - Three main blockers:
            - Behaviour outside of modules (re-resolving dependencies again and again)
            - Subversion support
            - Vendor proposal (see below)
        - 1.14 is a really short cycle - still trying to figure out whether this list is feasible in that time pre-freeze
    - Vendor proposal update (https://github.com/golang/go/issues/33848):
        - Seems to have been very well received); no objections. Hence seems to suggest proposal will be accepted
        - Open question: when to trigger using vendor directory by default
        - Currently leaning towards go1.14 in go.mod + presence of vendor directory; get feedback via implementation
        - Bryan is going to start implementation so that's ready to be tested once proposal approved, giving as much time as possible pre freeze
    - proposal: cmd/go: ignore +incompatible versions as of Go 1.14 - https://github.com/golang/go/issues/34217
        - +incompatible versions are a thorn for everyone
        - Feels too aggressive for 1.14 based on feedback
        - Looking at less aggressive solutions for 1.14
        - So proposal will likely not happen as proposed

    - Request for clarification of when modules will disappear from the module mirror
        - Mirror is operated by Google, hence must abide by the law, which by implication means that licenses (or a lack thereof) need to be honoured
        - Modules without a "suitable license" are not guaranteed to remain in the mirror
        - What does "suitable" mean?
        - This is a legal answer
        - The discovery site needs to apply the same restriction, so in effect a lack of documentation in the discovery site will be an indicator that a module doesn't have a "suitable" license
        - Modules that do not have a recognised license will be classified as "non distributable"
        - Goal is to then publish a "best practice" guide on licenses
        - Using a local proxy or vendoring (in some form) remains the way to defend against lack of availability of the module mirror or a module within the mirror
        - Documentation will be improved to be clear on the scenarios in which modules can disappear from the mirror
        - Suggestion to create an issue (locked perhaps, just for publishing updates?) so that people can track progress
    - Moving forward the gobin-like proposal: where do we go from here? https://github.com/golang/go/issues/33518#issuecomment-532643149
        - Still on the radar, not likely to land for Go 1.14
        - People still considering
    - Is the "global install" issue https://github.com/golang/go/issues/30515 likely to happen for 1.14? Is any input needed?
        - Jay has a proposal for this: https://github.com/golang/go/issues/34506
- go/(packages|analysis)
    - Suggested fixes are now ready for use; staticcheck (integrated with gopls) is already using this
- Discovery site (Julie)
    - Still targeting a beta release by end of year
    - Module and package directory views have now been added
    - Also fixed some edge cases with standard library
    - Paperwork being finished up, as well as some work on reliability, monitoring etc
    - Looking to move to a permanent new home
    - More regular updates to be posted to the locked issue: https://github.com/golang/go/issues/33654
    - godoc getting module support is separate from the discovery site; they are separate projects
    - The discovery site being launched GA is something the team is looking to do before Go 1.14
- Update on proposal to move mature wiki content behind Gerrit (https://github.com/golang/go/issues/34038)
    - Mostly positive feedback
    - Currently blocked on understanding the overlap with x/website
    - Modules related documentation is likely going to move to x/website
    - Huge overlap here with Carmen's /community proposal
    - We are at a minimum missing a way of mapping links from the wiki to URLs under golang.org
- Update on the modularise tool proposal to manage existing larger projects in the context of Go modules. (https://docs.google.com/document/d/1g7wIlxn9JBJkc-2IHAb2MfCYLItWZKyNIIO6SBW1x-8/edit#)
    - Scope here is practical workflows for migrating to and maintaining large projects to modules
    - Those interested are encouraged to review and offer feedback
- Ian travelling
    - London at https://www.meetup.com/LondonGophers/events/264779288/ on Oct 16
    - Florence at GoLab https://golab.io/ Oct 20-22

Date for next call

Tue 29 Oct 15:30 UTC

WIP agenda: https://docs.google.com/document/d/1F3T58Nj_Ft3bu15Wd4hAZAW6kLl_M1EH5XP43l_3CpY/edit#

(note this date is straddled by the daylight savings changes in the UK and US, so this call will be at 15:30 in the

UK)