Vendoring & Versioning with Go

4 November 2016

Chris Roche
Software Engineer, Lyft

Go's Vendoring History

In the beginning, there was the mono-repo

- Project lived in Google's mono-repo
- Mercurial(ish) system
- Vendoring not an issue in this world
- Using internal tooling (Bazel) in any case
- Each directory considered distinct package regardless of hierarchy
- go tools built out to support mutliple repos and VCS

Go 1.0 - 1.4 : WYSIWYG

- All non-stdlib dependencies must live under `$GOPATH/src`
- Works perfectly fine for subpackages in the same repo
- External dependency versions completely neglected
- Tools built to change refs before running build/tests, etc.
- Workaround: use a project-specific `$GOPATH`

Path to ./vendor

Go 1.5 (Aug 2015)
- Enabled use of the `./vendor` directory
- Disabled by default
- Enable: `GO15VENDOREXPERIMENT=1`

Go 1.6 (Feb 2016)
- Enabled by default
- Disable: `GO15VENDOREXPERIMENT=0`

Go 1.7 (Aug 2017)
- Cannot disable

How does it work?

First: ./vendor

- From: *"github.com/foo/bar"*
- To: *"github.com/fizz/buzz/vendor/github.com/foo/bar"*
- Never import vendor (compilation will fail or behave strangely)
- Commit or not to commit? (up to preference)

Second: $GOPATH/src

- This is how subpackages of a project are loaded
- Useful for testing/debugging libraries in dependent's context

Finally: $GOROOT/src

- Standard Library exclusively
- Never ever ever touch/add to these

Sidebar: Versioning Go with Gimme

github.com/travis-ci/gimme (https://github.com/travis-ci/gimme)

```
brew install gimme # other ways (it's just a shell script)
gimme 1.7.3
go version
go version go1.7.3 darwin/amd64
```

Configure in your bash/zsh profile:

```
source $HOME/.gimme/envs/latest.env
```

Configure in IntelliJ (per project):

- Module settings (⌘↓)
- Platform Settings / SDKs
- + in top left / Go SDK
- (turn on hidden)
- "/Users/gopher/.gimme/versions/go.1.7.3.darwin.amd64"

Vendoring/Versioning Tools

- No blessed Go tool (working on it (https://docs.google.com/document/d/18tNd8r5DV0yluCR7tPvkMTsWD_lYcRO7NhpNSDymRr8), though)
- GoDep: one of the first, `JSON` based, scrubs vendored code
- gpm: no support for `./vendor`, just update VCS refs in `$GOPATH`
- gvp: change $GOPATH to a localized `./.godeps`, works with gpm
- gb: replaces go toolchain, does not use `./vendor` (correctly)

Glide

Features

- Correctly uses `./vendor`
- Leaves `$GOPATH` alone
- YAML based configuration file
- Locking system similar to Composer (PHP), Bundler (Ruby), and Cargo (Rust)
- Aware of Godep config files
- Respects configuration of transient dependencies
- Useful CLI, but certainly not perfect

Ecosystem: $HOME/.glide

- The VCS cache
- Independent of `$GOPATH` and `./vendor`
- Don't touch!

Ecosystem: ./glide.yaml

- Human's Configuration
- YAML-based
- Override VCS behavior
- Commit / Branch / Tag / SemVer aware
- Only include your direct dependencies

```
package: github.com/example/pkg

import:
- package: github.com/external/dep
  version: unstable-branch
- package: github.com/example/other-pkg
  version: ^1.2.3
  repo:    git@github.com:example/other-pkg

testImport:
- package: github.com/stretchr/testify
  subpackages:
  - assert
```

Ecosystem: ./glide.lock

- Computer's Configuration
- Snapshot of what the crazy human wants
- Look Don't Touch!
- Always pinned to a SHA (or equivalent)
- Includes hash of glide.yaml
- All explicit and transient dependencies
- Must be committed

```
hash: d8dc02f36d3bd58163dfc37dfd022a8539e31258d8f2c2ad417ef8f3d6d76d2a
updated: 2016-10-06T17:33:31.683461401-04:00
imports:
- name: github.com/external/dep
  version: 74a703abb31ea9faf7622930e5daba1559b01b37
```

glide init: Starting a New Project

- walks any existing code and creates a glide.yaml
- will prompt about versioning if semver is detected
- **nothing is installed!**
- allows modification of glide.yaml before install

```
glide init
[INFO]  Generating a YAML configuration file and guessing the dependencies
[INFO]  Attempting to import from other package managers (use --skip-import to skip)
[INFO]  Scanning code to look for dependencies
[INFO]  --> Found reference to github.com/fizz/buzz
[INFO]  --> Found reference to github.com/foo/bar
[INFO]  Writing configuration file (glide.yaml)
```

glide update: Update All Deps and Lock

- Walks the code and glide.yaml for dependencies (including transitive)
- YAML: guidelines, not rules. Your code is glide's source of truth
- Updates **EVERYTHING**, so pin
- Generates new glide.lock

```
glide update
[INFO]  Downloading dependencies. Please wait...
[INFO]  --> Fetching updates for github.com/fizz/buzz.
[INFO]  --> Fetching github.com/foo/bar.
[INFO]  Resolving imports
[INFO]  --> Fetching github.com/baz/dep-of-bar.
[INFO]  Downloading dependencies. Please wait...
[INFO]  Setting references for remaining imports
[INFO]  Exporting resolved dependencies...
[INFO]  --> Exporting github.com/fizz/buzz
[INFO]  --> Exporting github.com/foo/bar
[INFO]  --> Exporting github.com/baz/dep-of-bar
[INFO]  Replacing existing vendor dependencies
[INFO]  Project relies on 3 dependencies.
```

glide install: Dependencies from Lock

- Installs what's specified in **the lock file**
- Does not check branches, tags, phase of the moon
- **Gaurantees reproducibility**
- If no lock present, runs update instead

```
glide install
[INFO]  Downloading dependencies. Please wait...
[INFO]  --> Found desired version locally github.com/fizz/buzz 77ed1c8a01217656d2080ad51981f6e99adaa177!
[INFO]  --> Found desired version locally github.com/foo/bar d15fa2e2a63dd52104bc96d8ea7dc47ce8027de8!
[INFO]  --> Found desired version locally github.com/baz/dep-of-bar 9fa8f10901c17b49ed52a824cf9226006580
[INFO]  Setting references.
[INFO]  --> Setting version for github.com/fizz/buzz to 77ed1c8a01217656d2080ad51981f6e99adaa177.
[INFO]  --> Setting version for github.com/foo/bar to d15fa2e2a63dd52104bc96d8ea7dc47ce8027de8.
[INFO]  --> Setting version for github.com/baz/dep-of-bar to 9fa8f10901c17b49ed52a824cf9226006580a06d.
[INFO]  Exporting resolved dependencies...
[INFO]  --> Exporting github.com/fizz/buzz
[INFO]  --> Exporting github.com/foo/bar
[INFO]  --> Exporting github.com/baz/dep-of-bar
[INFO]  Replacing existing vendor dependencies
```

glide get: Add a dependency

- Like go `get`, but adds to glide.yaml
- Specify multiple packages at once
- Will ask you questions about SemVer if detected
- Runs an update on all dependencies
- Updates glide.lock

```
glide get github.com/foo/bar github.com/fizz/buzz
```

Misconceptions

Using `master` does what you expect

- Install only looks at the SHA in the lock file
- Update will pull in the latest, but may not be desirable
- Just pin, deliberately upgrade

Updates are predictable

- Only predictable if you pin everything
- Things will change
- Read the output of update, it will walk you through its decisions

Troubleshooting

[WARN] Lock file may be out of date. Hash check of YAML failed. You may need to run 'update'

CAUSE

- You made changes to your YAML, didn't run update, and tried to install
- When not local (or after pulling someone's code), means lock wasn't committed

SOLUTION

```
glide update
```

[WARN] Version not set for package github.com/example/pkg

CAUSE

- `version` is not specified in glide.yaml for a particular package.

SOLUTION

- Libraries should provide a SemVer ranges for dependencies
- Non-library packages should pin to an explicit version

That package should not be installing

- Is it in your glide.yaml?
- Is it imported in your code?
- Is it in your tests?
- Is it a transient dependency?

Failed to update/download/"set version" on github.com/foo/bar

- Can you `go get` the package and the desired ref?
- Is the specified version correct?
- Is the specified repo correct? (private repos in CI can be complicated)
- Run with the debug flag to see more details

When in doubt?

- Is your glide up-to-date?
- `glide -q <cmd>` (see the errors)
- `glide --debug <cmd>` (see more info)
- `glide cache-clear` (delete $HOME/.glide)
- delete your `./vendor` and re-run `glide install/update`
- 😭 (jk, come talk to us)

Working with Local Dependencies

- Work out of vendor (but no VCS)
- Delete vendored package and use the one in $GOPATH (careful to rerun glide when done)

Links

Glide Docs

glide.readthedocs.io/en/latest (http://glide.readthedocs.io/en/latest)

Package Management Official Proposal

docs.google.com/document/d/18tNd8r5DV0yluCR7tPvkMTsWD_lYcRO7NhpNSDymRr8

(https://docs.google.com/document/d/18tNd8r5DV0yluCR7tPvkMTsWD_lYcRO7NhpNSDymRr8)

Thank you

Chris Roche
Software Engineer, Lyft