

# REFACTORING TO MODULES:

All you need to know  
in less than an hour

LET'S GO BACK IN  
TIME



POLL TIME!



Pre 1.0 (2012)



1.2 (2013)



1.5 (2015)



1.8 (2017)



1.11 (2018)

# BARUCH SADOGURSKY

CHIEF STICKER OFFICER

(ALSO, HEAD OF DEVELOPER RELATIONS)



JBARUCH@JFROG.COM



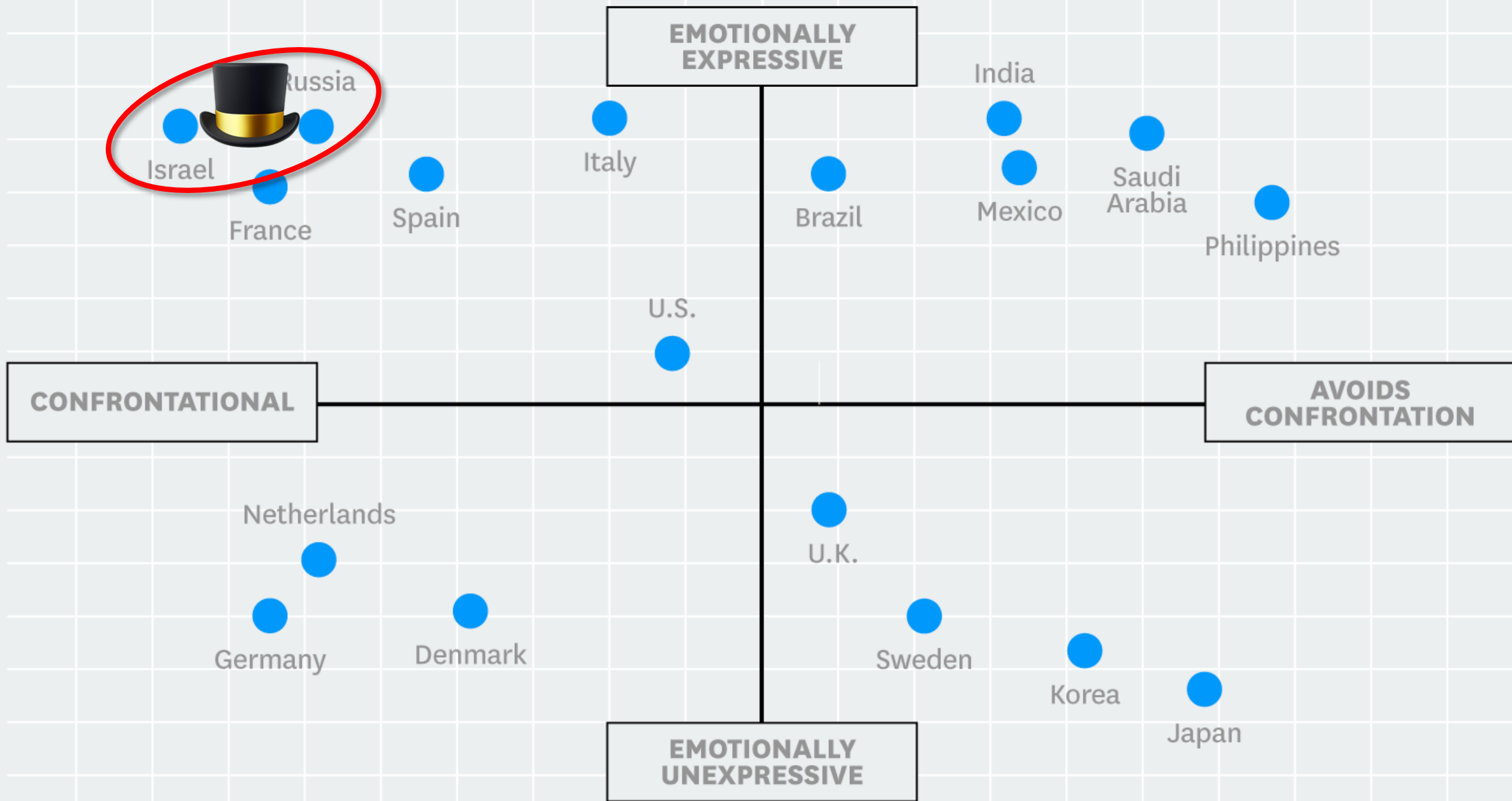
@JBARUCH



+1(408)890-9281







# SHOWNOTES



<http://jfrog.com/shownotes>



Slides



Video



Links



Comments, Ratings



Raffle

# WHY WE HAVE A PROBLEM?

## History

Design began in late 2007.

Key players:

- Robert Griesemer, Rob Pike, Ken Thompson
- Later: Ian Lance Taylor, Russ Cox

# WHY WE HAVE A PROBLEM?

A personal history of dependencies at Google

Plan 9 demo: a story

Early Google: one Makefile

2003: Makefile generated from per-directory BUILD files

- explicit dependencies
- 40% smaller binaries

Dependencies still not checkable!



# SIMPLE SOLUTION!


- 🐸 Dependencies are sources
- 🐸 Remote import is a VCS path
- 🐸 Dump everything together into one source tree (GOPATH)
- 🐸 Compile
- 🐸 Profit




@jbaruch #golang <http://jfrog.com/shownotes>

**BUT... HOW DO I...**

 Know which dependencies do I use?

 Know which dependencies did you use?

 Know which dependencies should I use?

 Know is it our code that I am editing right now?

 WTF is going on?!

# YEAH...

“ To date, we’ve resorted to an email semaphore whenever someone fixes a bug a package, imploring everyone else to run `go get -u`. You can probably imagine how successful this is, and how much time is being spent chasing bugs that were already fixed.

Dave Cheney



# DUPLICATE YOUR DEPENDENCIES

“ Check your dependencies to your own VCS.

Brad Fitzpatrick

# BUILD YOUR OWN DEPENDENCY MANAGER

“ It's not the role of the tooling provided by the language to dictate how you manage your code in the production sense.

Andrew Gerrand

# WE EXPECT YOU TO ALREADY HAVE A HOMEGROWN DEPENDENCY MANAGER

“ If you need to build any tooling around what Go uses (Git, Mercurial, Bazaar), you already understand those tools, so it should be straightforward to integrate with whatever system you have.

Andrew Gerrand

# DON'T TRUST WHAT WE'VE BUILT

“ go-get is nice for playing around, but if you do something serious, like deploying to production, your deploy script now involves fetching some random dude's stuff on GitHub.

Brad Fitzpatrick



**NEXT THING YOU KNOW...**

**THERE ARE 19 DEPENDENCY MANAGERS**



# QUIZ TIME!



godeps.json



dependencies.tsv



govendor, govend,  
goven, gv



trash, garbage, rubbish



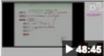
Weapons manufacturer

GOPATH + VENDORING = 🌟💖🌟

[Google Stores Billions of Lines of Code in a Single Repository ...](#)  
<https://www.youtube.com/watch?v=W71BTkUbdqE> ▼  
Sep 15, 2015 - Uploaded by @Scale  
This talk will outline the scale of Google's codebase, describe Google's custom-built monolithic source ...



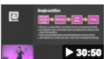
[Bazel How to build at Google scale? - YouTube](#)  
<https://www.youtube.com/watch?v=OUwu0myF8M>  
Mar 10, 2018 - Uploaded by FOSDEM  
Bazel is Google's own build tool, open-sourced in Fe... ... Uber Technology Day: Monorepo to Multirepo and ...




[Billions of lines of code in a single repository, SRSLY? by Guillaume ...](#)  
<https://www.youtube.com/watch?v=yM0GQw1zgrA> ▼  
Nov 10, 2016 - Uploaded by Devxxx  
Google stores all its source code in one single monolithic repository! Imagine 25000 software developers ...




[Very interesting talk about how Google handles a monorepo at ...](#)  
<https://dev.to/matteojoliveau/comment/7a34>  
Dec 2, 2018 - Uploaded by @Scale  
What are the pros and cons of the "monorepo" approach? ...




[Git at Google: Making Big Projects \(and Everyone Else\) Happy, Dave ...](#)  
<https://www.youtube.com/watch?v=cY34mr71ky8> ▼  
May 12, 2015 - Uploaded by GitHub  
Google likes to push the boundaries of what's possible with Git. With big projects like Android and Chrome, we ...




[The Curious Case of Monorepos, Johannes Stein - React London ...](#)  
<https://www.youtube.com/watch?v=DFCtAupKbEk>  
Sep 4, 2017 - Uploaded by Red Badger  
Monorepos have often stigmatized as bad practice and as such something that ... Still, big companies like ...



[Uber Technology Day: Monorepo to Multirepo and Back Again ...](#)  
<https://www.youtube.com/watch?v=IV8-1S28ycM>  
Apr 7, 2017 - Uploaded by Uber Engineering  
Aimee Lucido, a mobile engineer on Uber's Driver Signups team, spoke about Uber's transition from a ...



[The Challenge of Monorepos: Strategies from git-core and Open ...](#)  
<https://www.youtube.com/watch?v=FSYBask5ao> ▼  
May 12, 2015 - Uploaded by GitHub  
The problem of Monorepos, and related large files and locking issues, ... Why Google Stores Billions of Lines ...



[Monorepos in the Wild - Markus Oberlehner @ WeAreDevelopers ...](#)  
<https://www.youtube.com/watch?v=kwhOl4mmqnM>  
May 30, 2017 - Uploaded by WeAreDevelopers  
Visit the largest developers congress in Europe: WeAreDevelopers World Congress, 16 - 18 May 2018 in ...



[BazelCon 2018 Day 1: Virtual Mono-Repo & Bazel - YouTube](#)  
<https://www.youtube.com/watch?v=2gNITegwQD4>  
Nov 2, 2018 - Uploaded by Google Open Source  
Reproducible HEAD dependency in a many-repo world Ittai Zeidman (Wix.com)



# GOPATH, THE PROUD SON OF THE MONOREPO



# TWO HUGE PROBLEMS WITH GOPATH



- It only allows a single version of any given package to exist at once (per GOPATH)



- We cannot programmatically differentiate between code the user is working on and code they merely depend on

# VENDORING – THE WORST KIND OF FORKING

“ Copy all of the files at some version from one version control repository and paste them into a different version control repository

# WHAT'S WRONG WITH IT (WELL, WHAT'S NOT)

- 🐸 History, branch, and tag information is lost
- 🐸 Pulling updates is impossible
- 🐸 It invites modification, divergence, and bad fork
- 🐸 It wastes space
- 🐸 Good luck finding which version of the code you forked

BBC

**WAIT A MINUTE...**

**WE USE SUBMODULES!**

**STILL WRONG!**



You still have no idea what version are you using



You have to connect each dependency as a submodule manually



Switching branches and forks LOL



Working on modules with other teams ROFL







## THE GO DEP EXPERIMENT





# So you want to write a package manager

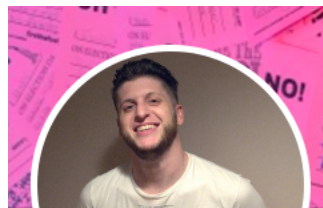
You woke up this morning, rolled out of bed, and thought, “Y’know what? I don’t have enough misery and suffering in my life. I know what to do—I’ll write a language package manager!”

...

## Package management is awful, you should quit right now

Package management is a nasty domain. Really nasty. On the surface, it *seems* like a purely technical problem, amenable to purely technical solutions. And so, quite reasonably, people approach it that way. Over time, these folks move inexorably towards the conclusion that:

1. software is terrible
2. people are terrible
3. there are too many different scenarios
4. nothing will really work for sure
5. it’s provable that nothing will really work for sure
6. our lives are meaningless perturbations in a swirling vortex of chaos and entropy



**sam boyer**

@sdboyer

systems | people

📍 Detroit metro

📅 Joined December 2008

# PROPER DEPENDENCY MANAGEMENT?



Working in project  
directories



Local cache for  
dependencies



Version declarations



Conflict resolution

**CONFLICT ON THE  
CONFLICT  
RESOLUTION**



SAT/SMT  
vs  
MVS/SIV



**ENTER GO  
MODULES**



# ENTER GO MODULES

# BACKWARDS COMPATIBILITY AND MIGRATION



`go mod init`



`go.mod` file is created



The rest is the same:  
imports in code just work

**THAT'S SOME  
SERIOUS MAGIC...**





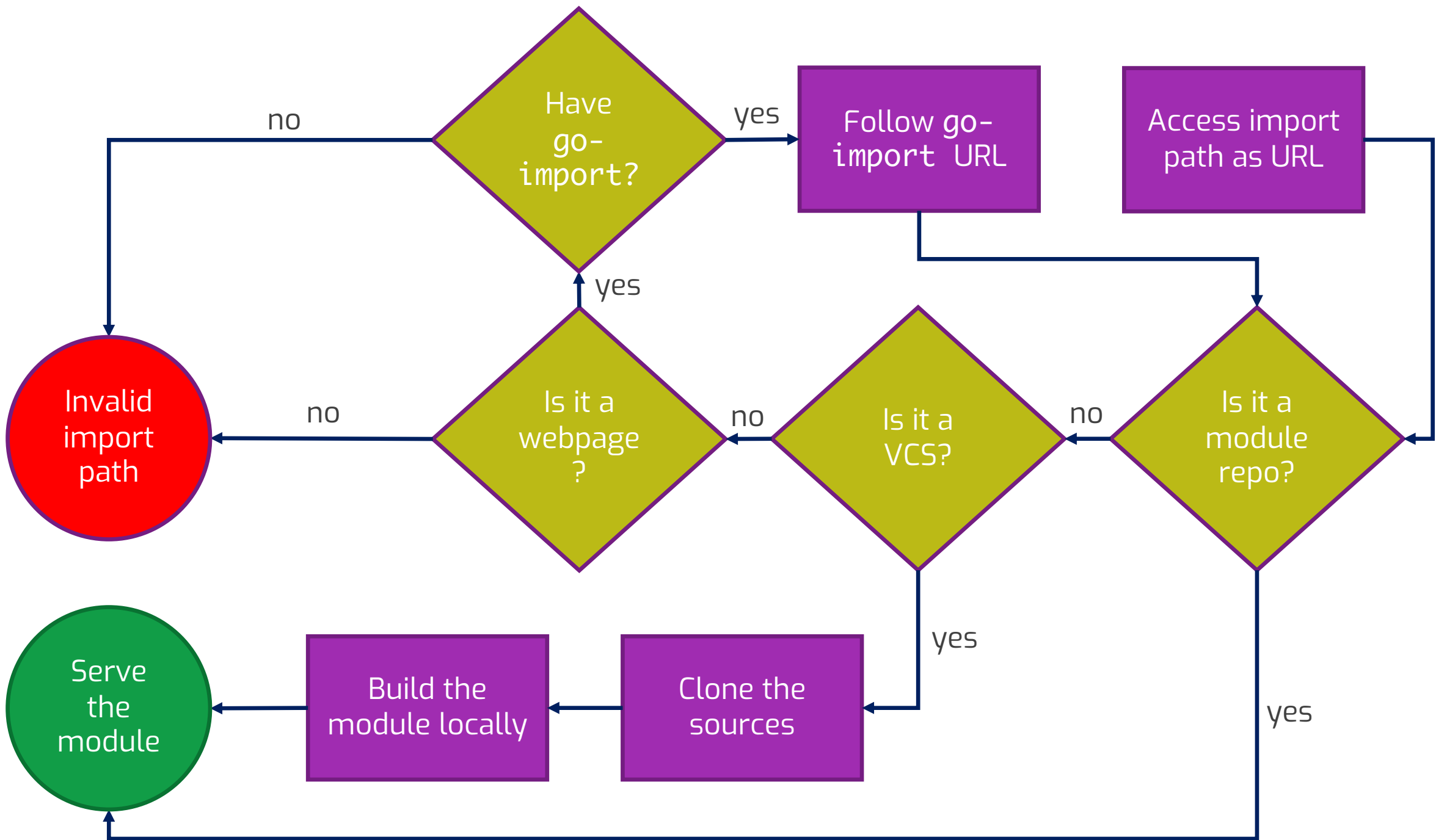
# GO MODULES CONVERT EVERYTHING (ALMOST?)

```
var Converters = map[string]func(string, []byte) (*modfile.File, error){
    "GLOCKFILE":          ParseGLOCKFILE,
    "Godeps/Godeps.json": ParseGodepsJSON,
    "Gopkg.lock":          ParseGopkgLock,
    "dependencies.tsv":    ParseDependenciesTSV,
    "glide.lock":          ParseGlideLock,
    "vendor.conf":         ParseVendorConf,
    "vendor.yml":          ParseVendorYML,
    "vendor/manifest":     ParseVendorManifest,
    "vendor/vendor.json": ParseVendorJSON,
}
```





**WHAT HAPPENS TO GO.MOD WHEN  
YOU ADD IMPORT  
(AND RUN GO GET/GO BUILD)**



# SELECTING A VERSION?

Latest  
compatible  
version tag

**EASY.**



@jbaruch    #golang    <http://jfrog.com/shownotes>

# COMPATIBLE?!



Let's assume SemVer works (LOL)



The latest version of v1.x.x is compatible with v1.0.0 and up



Premise: import path string should always be backwards compatible



# WHAT ABOUT VERSION 2?!



] Incompatible code can't  
use the same import path



Add /v2/ to the module  
path



Use /v2/ in the import  
path

```
import "github.com/my/module/v2/mypkg"
```

**WHAT IF IT DOESN'T  
HAVE ANY SEMVER  
TAGS?!**



Pseudo version

v0.0.0-yyyyymmddhhmmss-abcdefabcdef



# WHAT IF (WHEN) I WANT TO BAN A VERSION?!



You can specify “version X or later”: `>= x.y.z`



You can use `exclude` or `replace` for better control

FROM VENDORING  
TO HIERARCHY OF  
MODULE  
REPOSITORIES



# GO MODULES DEFINE AN HIERARCHY OF CACHES

Public Modules Repository

GoCenter

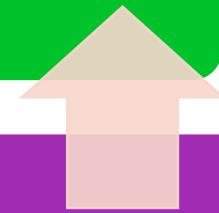
Organizational Modules Repository

The Athens Project

JFrog Artifactory

Local cache on the developer's machine

`$GOPATH/pkg/mod`



# LOCAL CACHE ON THE DEVELOPER'S MACHINE



After the mods are resolved (or built) they are cached in `$GOPATH/pkg/mod`



Provides immediate access



Not shared



Not reliable (can be wiped at any moment)

# ORGANIZATIONAL MODULES REPOSITORY



JFrog Artifactory or  
Project Athens



Provides faster (Intranet)  
access



Provides reproducible  
builds as it caches the  
dependencies used once  
for build reproduction



Requires team  
infrastructure and  
maintenance (SaaS offers  
exist)



# PUBLIC MODULES REPOSITORIES



## GoCenter



Google announced a vision for a federation of public repositories



Provides fast access



Provides reproducible builds as it caches the popular and requested dependencies from version control



Highly available, requires no maintenance, free

# TWITTER ADS AND Q&A



[jfrog.com/shownotes](http://jfrog.com/shownotes)



[@jbaruch](https://twitter.com/jbaruch)



[#GoLang](https://twitter.com/#!/golang)



[gocenter.io](http://gocenter.io)