链滴

从 gopath 到 go mod 的一次尝试

作者：xhaoxiong

原文链接：https://ld246.com/article/1551701357087

来源网站：链滴

windows下的尝试:

gomod初尝试

1. 下载官方包1.11(及其以上版本将会自动支持 gomod) 默认GO111MODULE=auto(auto是指如在gopath下不启用mod)

go mod help 查看帮助

go mod init(项目模块名称)初始化模块，会在项目根目录下生成

go.mod 文件。

go mod tidy根据go.mod文件来处理依赖关系。

go mod vendor将依赖包复制到项目下的 vendor 目录。建议一些使用了被墙包的话可以这么处理，方便用户快速使用命令go build -mod=vendor编译

go list -m all显示依赖关系。go list -m -json all显示详细依赖关系。

go mod download <path@version>下载依赖。参数(path@version)是非必写的，path是包的路，version是包的版本。

2. 在 gopath外新建一个项目，单独开一个cmd 设置set GO111MODULE=on (习惯性的和git初始一样)go mod init然后报错了。 正解如下： go mod xxx（module名称可与文件名不同）

3. 在项目目录下执行 go mod tidy 下载完成后项目路径下会生成go.mod和go.sum

go.mod文件必须要提交到git仓库，但go.sum文件可以不用提交到git仓库(git忽略文件.gitignore中置一下)。

4. go模块版本控制的下载文件及信息会存储到GOPATH的pkg/mod文件夹里。

5. 在国内访问 golang.org/x的各个包都需要翻墙，我们可以在go.mod中使用replace替换成githu上对应的库。(强烈建议翻墙，我使用的lantern专业版+proxifier)非常稳定

以下是在公司项目中碰到的几点坑

1. 在引用mongodb的包时候报错

go mod labix.org/v2/mgo@v0.0.0-20140701140051-000000000287: bzr branch --use-existin -dir

解决办法 谷歌论坛

在go.mod中

```
replace (
labix.org/v2/mgo => github.com/go-mgo/mgo v0.0.0-20160801194620-b6121c6199b7
launchpad.net/gocheck => github.com/go-check/check v0.0.0-20180628173108-788fd784017
)
```

原文链接：从 gopath 到 go mod 的一次尝试

2. 引入本地包的方法(在go.mod中)

```
require (
    test v0.0.0
)

replace (
 test => ../test
)
```

注意:1.引入的包必须也是gomod的(有.mod文件)
    2.replace时必须使用相对路径比如../ ./
    3.require 的包后必须带版本号,replace中可带可不带

3. go.mod文件必须传入git服务器上

linux下的尝试

几乎都是翻墙的问题

```
replace (
    cloud.google.com/go => github.com/googleapis/google-cloud-go v0.34.0
    gopkg.in/tomb.v1 => github.com/go-tomb/tomb v1.0.0-20141024135613-dd632973f1e7
    go.opencensus.io => github.com/census-instrumentation/opencensus-go v0.19.0
    go.uber.org/atomic => github.com/uber-go/atomic v1.3.2
    go.uber.org/multierr => github.com/uber-go/multierr v1.1.0
    go.uber.org/zap => github.com/uber-go/zap v1.9.1

    golang.org/x/crypto => github.com/golang/crypto v0.0.0-20181001203147-e3636079e1a4
    golang.org/x/lint => github.com/golang/lint v0.0.0-20181026193005-c67002cb31c3
    golang.org/x/net => github.com/golang/net v0.0.0-20180826012351-8a410e7b638d
    golang.org/x/oauth2 => github.com/golang/oauth2 v0.0.0-20180821212333-d2e6202438
e
    golang.org/x/sync => github.com/golang/sync v0.0.0-20181108010431-42b317875d0f
    golang.org/x/sys => github.com/golang/sys v0.0.0-20181116152217-5ac8a444bdc5
    golang.org/x/text => github.com/golang/text v0.3.0
    golang.org/x/time => github.com/golang/time v0.0.0-20180412165947-fbb02b2291d2
    golang.org/x/tools => github.com/golang/tools v0.0.0-20181219222714-6e267b5cc78e
    google.golang.org/api => github.com/googleapis/google-api-go-client v0.0.0-201812200
0619-583d854617af
    google.golang.org/appengine => github.com/golang/appengine v1.3.0
    google.golang.org/genproto => github.com/google/go-genproto v0.0.0-20181219182458
5a97ab628bfb
    google.golang.org/grpc => github.com/grpc/grpc-go v1.17.0
    gopkg.in/alecthomas/kingpin.v2 => github.com/alecthomas/kingpin v2.2.6+incompatible
    gopkg.in/mgo.v2 => github.com/go-mgo/mgo v0.0.0-20180705113604-9856a29383ce
    gopkg.in/vmihailenco/msgpack.v2 => github.com/vmihailenco/msgpack v2.9.1+incompati
le
    gopkg.in/yaml.v2 => github.com/go-yaml/yaml v0.0.0-20181115110504-51d6538a90f8
    labix.org/v2/mgo => github.com/go-mgo/mgo v0.0.0-20160801194620-b6121c6199b7
    launchpad.net/gocheck => github.com/go-check/check v0.0.0-20180628173108-788fd784
127
)
```

主要包括:golang.org google.golang.org gopkg.in go.uber.org cloud.google.com 在下载包时会timeout 导致编译失败，以上是对应的github的库

参考资料

ieevee.com

鸟窝

segmentfault

----------分割线----------

Go Module 工程化实践（一）：础概念篇

Go Module 工程化实践（二）：go get 取包原理篇

根据以上资料总结

1. gopath 与go mod的区别

环境变量GOPATH不再用于解析imports包路径，即原有的GOPATH/src/下的包，通过import是找到了。
Go Module功能开启后，下载的包将存放与$GOPATH/pkg/mod路径
$GOPATH/bin路径的功能依旧保持

2. go get 流程的变化

老的go get取包过程类似：git clone + go install，开启Go Module功能后go get就只有 git clone 者 download过程了。
新老实现还有一个不同是，两者存包的位置不同。前者，存放在$GOPATH/src目录下；后者，存放在GOPATH/pkg/mod目录下。
老的go get取完主包后，会对其repo下的submodule进行循环拉取。新的go get不再支持submodul子模块拉取。

3. 依赖包的变化

三方远程包:
检查远程仓库最新的tag版本，有就取得该版本
远程仓库没有tag版本时，直接获取master分支的HEAD版本
如果在go.mod文件中指定了具体版本，go get直接获取该指定版本
go.mod中除了可以指定具体版本号以外，还支持分支名

本地包:
通过replace()进行替换

4. 私有仓库权限和私有vcs非标准路径取包问题

权限问题
windows10下:

控制面板>用户账户>凭据管理手动添加普通凭据即可
linux下:
增加 $HOME/.gitconfig 配置:
[url "ssh://git@github.com/MYORGANIZATION/"]
insteadOf = https://github.com/MYORGANIZA...

增加 $HOME/.netrc:
machine github.com login YOU password APIKEY
将其中的 APIKEY 换成自己的登录KEY。

增加 $HOME/.netrc:

machine github.com login YOU password APIKEY

将其中的 APIKEY 换成自己的登录KEY。

非标准路径问题(https://private.vcs.com:20000)
搭建一个中间服务：https://private.vcs.com 能够通过go get的包路径匹配查询正确的仓库地址。```