[golang开发:环境篇(七) Go mod 使用](https://www.cnblogs.com/feixiangmanon/p/12709221.html)

<https://www.cnblogs.com/feixiangmanon/p/12709221.htm>

**Glide用的好好的,为什么要使用Modules**

在 Go 1.11 中 ，官方加入package management tool，称为Go Modules。Go mod 没有出现之前，用的最多的包管理器就是 govendor、glide等,这些工具都未达到Go对包管理的预期，比如每次下载，比如墙外下载，比如对GOPATH的依赖，比如多版本的控制等等。然后Go Modules就出现了，它不依赖于GOPATH，只跟项目有关，可以指定代理，可以很容易的通过版本进行控制，摆脱GOPATH的依赖，也为Go以后的自由发展奠定了基础。

**使用Go mod 命令管理包**

| **命令** | **描述** |
| --- | --- |
| go init | 在当前目录项目下初始化mod |
| go tidy | 拉取依赖的模块，移除不用的模块 |
| go vendor | 将依赖复制到vendor下 |
| go edit | 编辑go.mod |
| go verify | 验证依赖是否正确 |

其实工作基本上都使用init和tidy就够了。

**设置环境变量**

GO111MODULE  
有三个值，off、on、auto，off 和 on 即关闭和开启，auto 则会根据当前目录下是否有 go.mod 文件来判断是否使用 modules 功能。无论使用哪种模式，module 功能默认不在 GOPATH 目录下查找依赖文件。  
GOPROXY  
设置代理服务，[https://goproxy.io](https://goproxy.io/" \t "_blank)。也可以自己搭代理服务，然后把 GOPROXY 设置为代理服务器的地址。  
vim ~/.bash\_profile  
加入配置的两行  
export GO111MODULE=on  
export GOPROXY=https://goproxy.io

source ~/.bash\_profile

**举个栗子**

创建项目 myproject  
main.go

package main

import (

"github.com/satori/go.uuid"

"fmt"

)

func main() {

uid := uuid.NewV4()

fmt.Println(uid)

}

执行Go mod命令， init 和 tidy

go mod init

go: creating new go.mod: module myproject

go mod tidy

go: finding golang.org/x/tools latest

go: downloading golang.org/x/tools v0.0.0-20200415034506-5d8e1897c761

go: extracting golang.org/x/tools v0.0.0-20200415034506-5d8e1897c761

go: finding gopkg.in/check.v1 latest

go: downloading gopkg.in/check.v1 v1.0.0-20200227125254-8fa46927fb4f

go: extracting gopkg.in/check.v1 v1.0.0-20200227125254-8fa46927fb4f

go: finding github.com/niemeyer/pretty latest

go: downloading github.com/niemeyer/pretty v0.0.0-20200227124842-a10e7caefd8e

go: extracting github.com/niemeyer/pretty v0.0.0-20200227124842-a10e7caefd8e

go: downloading github.com/kr/text v0.1.0

go: extracting github.com/kr/text v0.1.0

编译执行结果

go build main.go

./main

6723138d-ab2c-4de6-b996-732362985548

可以看下Go mod生成的最主要的文件 go.mod

cat go.mod

module myproject

go 1.13

require (

github.com/niemeyer/pretty v0.0.0-20200227124842-a10e7caefd8e // indirect

github.com/satori/go.uuid v1.2.0

gopkg.in/check.v1 v1.0.0-20200227125254-8fa46927fb4f // indirect

)

每个包后面都跟了一个版本。如果想切换分支的话，后面的版本可以任意切换到需要的分支上，比如

require (

github.com/niemeyer/pretty master

github.com/satori/go.uuid v1.2.0

gopkg.in/check.v1 v1.0.0-20200227125254-8fa46927fb4f // indirect

)

也可以使用本地代码替换远程代码分支。就可以使用下面的  
/data/www/go/src/go.uuid 代替远程分支 github.com/satori/go.uuid。  
在go.mod最后一行加上下面的代码

replace github.com/satori/go.uuid => /data/www/go/src/go.uuid

Go mod的使用是不是特别简单。