# Go项目整体打包

## Docker

在项目创建dockerfile文件，打包到docker镜像，这种方式需要go.mod和go.sum两个文件，列出来需要的包和版本

## Vendor

用项目文件夹和包管理文件夹（一般是vendor）搭配来形成一个完整的项目，包管理文件夹中包含所有运行项目所依赖的环境。

1. go get github.com/tools/godep

使用godep进行包管理。  在pokerserver目录下执行godep save。godep会将所有的依赖包拷贝到pokerserver/vendor文件夹下面，并且去除到.git文件夹

1. 在环境变量中设置 GO15VENDOREXPERIMENT=1 它将开启go的vendor功能

## 自动生成go.mod、go.sum、vendor方法

1. 首先将你的版本更新到最新的Go版本1.11，如何更新版本可以自行百度。
2. 通过go命令行，进入到你当前的工程目录下，在命令行设置临时环境变量set GO111MODULE=on；
3. 执行命令go mod init在当前目录下生成一个go.mod文件，执行这条命令时，当前目录不能存在go.mod文件。如果之前生成过，要先删除；
4. 如果你工程中存在一些不能确定版本的包，那么生成的go.mod文件可能就不完整，因此继续执行下面的命令；
5. 执行go mod tidy命令，它会添加缺失的模块以及移除不需要的模块。执行后会生成go.sum文件(模块下载条目)。添加参数-v，例如go mod tidy -v可以将执行的信息，即删除和添加的包打印到命令行；
6. 如果执行上面命令失败，可能是因为一些包拉不下来（golang.org/x/下面的包），在环境变量中添加GO111MODULE=on和GOPROXY= <https://goproxy.io>两个环境变量，重启终端再试，有必要时重启电脑。（GO111MODULE有三种模式，on为开启module，off为开启gopath，auto为项目有module配置时使用module，没有时默认使用gopath，不修改时默认为auto）
7. 执行命令go mod verify来检查当前模块的依赖是否全部下载下来，是否下载下来被修改过。如果所有的模块都没有被修改过，那么执行这条命令之后，会打印all modules verified。
8. 执行命令go mod vendor生成vendor文件夹，该文件夹下将会放置你go.mod文件描述的依赖包，文件夹下同时还有一个文件modules.txt，它是你整个工程的所有模块。在执行这条命令之前，如果你工程之前有vendor目录，应该先进行删除。同理go mod vendor -v会将添加到vendor中的模块打印出来；

**参考资料：**<https://www.jianshu.com/p/bbed916d16ea>

# **golang.org/x/sys 包下载不下来**

1. **手动下载：git clone** <https://github.com/golang/sys.git>
2. **设置环境变量：**GOPROXY= <https://goproxy.io>
3. **在go.mod中的话替换下载包replace … …**

# **强制转换类型**

## **接口转结构体**

**interface.(\*struct) //将interface断言为某个结构体**

# **文件操作**

<https://studygolang.com/articles/19727?fr=sidebar>

# 代码功能打包

## 生成唯一订单号

strings.Replace(uuid.NewFastUUID().String(), "-", "", -1)

## 查询本机ip

**addrs, err := net.InterfaceAddrs()  
if err == nil {  
 for \_, address := range addrs {  
 if ipnet, ok := address.(\*net.IPNet); ok && !ipnet.IP.IsLoopback() {  
 if ipnet.IP.To4() != nil {  
 xx := ipnet.IP.String()  
 //if strings.HasPrefix(xx,"192.168.0"){  
 // fmt.Println(xx)  
 //}  
 fmt.Println(xx)  
 //break  
 }  
 }  
 //fmt.Println("ip：",address)  
 }  
}**