用 Makefile 管理 API 项目

本节核心内容

- 介绍 Makefile
- 介绍如何使用 Makefile

> 本小节源码下载路径：[demo11 (https://github.com/lexkong/apiserver_demos/tree/mas](https://github.com/lexkong/apiserver_demos/tree/mas)
>
> 可先下载源码到本地，结合源码理解后续内容，边学边练。
>
> 本小节的代码是基于 [demo10 (https://github.com/lexkong/apiserver_demos/tree/mas](https://github.com/lexkong/apiserver_demos/tree/mas) 来开发的。

为什么需要 Makefile

Go 语言的 go 命令自带源码管理功能，比如通过 `go build` 可以实现对源码的编译，但是 Go 自带的源码管理功能在实际项目中还是满足不了需求，有时候执行 `go build` 时，会附带很多编译参数，直接执行 `go build` 命令也会很麻烦。这时候一般是通过更专业的 Makefile 来管理源码，通过 Makefile 可以实现诸如：编译、安装、清理等功能，其实需要的管理功能都可以通过 Makefile 来添加，Makefile 生来就是做这些的。

Makefile 简介

一个工程中的源文件不计其数，其按类型、功能、模块分别放在若干个目录中，Makefile 定义了一系列的规则来指定，哪些文件需要先编译，哪些文件需要后编译，哪些文件需要重新编译，甚至于进行更复杂的功能操作，因为 Makefile 就像一个 Shell 脚本一样，其中也可以执行操作系统的命令（摘自百度百科）。

makefile 的规则

Makefile 基本格式如下:

```
target ... : prerequisites ...
    command
    ...
```

其中:

- target - 编译文件要生成的目标
- prerequisites - 编译文件需要的依赖
- command - 依赖生成目标所需要执行的命令（任意的 shell 命令），Makefile 中的命令必须以 [tab] 开头

比如我们平时使用的 `gcc a.c b.c -o test` 这里的 test 就是我们要生成的目标， a.c、b.c就是我们生成目标需要的依赖，而 `gcc a.c b.c -o test` 则是命令。将这行命令用 Makefile 的方式来写就是:

```
test: a.c b.c
    gcc a.c b.c -o test
```

API Server 添加 Makefile

在 apiserver 根目录下新建文件 Makefile，内容为:

```
all: gotool
    @go build -v .
clean:
    rm -f apiserver
    find . -name "[._]*.s[a-w][a-z]" | xargs -i
rm -f {}
gotool:
    gofmt -w .
    go tool vet . |& grep -v vendor;true
ca:
    openssl req -new -nodes -x509 -out
conf/server.crt -keyout conf/server.key -days
3650 -subj "/C=DE/ST=NRW/L=Earth/O=Random
Company/OU=IT/CN=127.0.0.1/emailAddress=xxxxx@qq.
com"

help:
    @echo "make - compile the source code"
    @echo "make clean - remove binary file and
vim swp files"
    @echo "make gotool - run go tool 'fmt' and
'vet'"
    @echo "make ca - generate ca files"

.PHONY: clean gotool ca help
```

上面的 Makefile 文件中，.PHONY 是个伪目标，形式上是一个目标，但是不需要依赖，伪目标一般只是为了执行目标下面的命令（比如 clean 就是伪目标）。@ 放在行首，表示不打印此行。默认在编译的过程中，会把此行的展开效果字符串打印出来。

上面的 Makefile 实现了如下功能：

- make：执行 go build -v . 生成 Go 二进制文件
- make gotool：执行 gofmt -w . 和 go tool vet .（格式化代码和源码静态检查）
- make clean：做一些清理工作：删除二进制文件、删除 vim swp 文件
- make ca：生成证书
- make help：打印 help 信息

编译

在前面各小节中编译二进制均是通过 go build -v . 的方式，添加 Makefile 后可以通过如下方式来编译：

```
$ make
```

小结

本小节简单介绍了 Makefile，并介绍了 apiserver 所使用的 Makefile 文件，通过该小节，展示了如何通过 Makefile 来管理和编译 API 源码。

本小册不是专门介绍 Makefile 的，想要了解更多 Makefile 知识，请参考 [Makefile使用总结](https://www.cnblogs.com/wang_yb/p/3990952.html)。