# Makefile学习—规则

## 1.规则基本构成

目标：目标依赖

命令

## 2.注意事项

命令必须使用tab键开头，一般是shell命令。

一个规则中可以没有目标依赖，仅仅实现某种操作，如：



一个规则中可以没有命令，仅仅描述依赖关系，如：



一个规则中必须有一个目标

## 3.目标

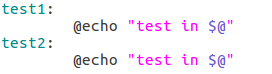
1）默认目标：一个Makefile里可以有多个目标，一般会选择第一个作为默认目标

2）多个目标：一个规则中可以有多个目标，多个目标具有相同的生成命令和依赖文件

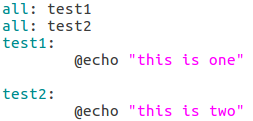




相当于：



3）多个规则目标：多个规则可以是同一个目标，Make在解释时，会将多个规则的依赖文件合并，如：



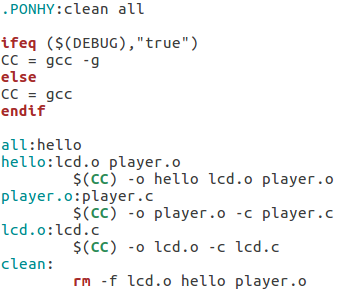


4）伪目标：并不是一个真正的文件名，可以看做是一个标签，无依赖，相比一般文件不会去重新生成、执行。伪目标可以无条件执行。

伪目标的生成，关键字.PONHY，如：



作用：实现某种操作，描述依赖关系，如以下makefile中的clean和all



## 4.目标依赖

### 4.1文件时间戳

根据时间戳来判断目标依赖文件是否更新，所有文件编译过，则对所有文件编译，生成可执行程序，在上次make之后修改过的c文件，会被重新编译，在上次make之后修改过的头文件，依赖此头文件的会被重新编译

### 4.2自动产生依赖

gcc –M或gcc -MM命令生成该文件要依赖的文件

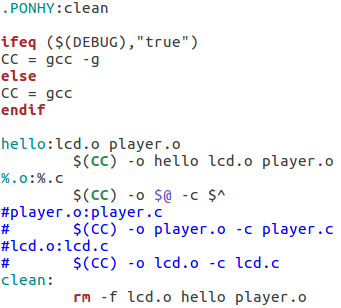


### 4.3模式匹配

目标依赖变量：$^

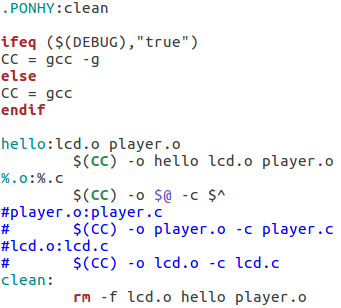
目标变量：$@

使用%.o:%.c来代替lcd.o:lcd.c，player.o:player.c像这样模式的语句

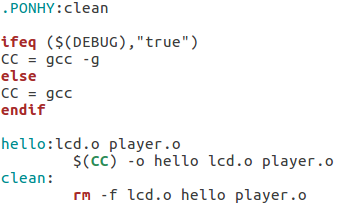


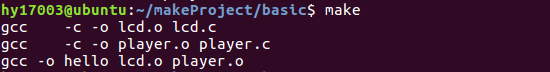
### 4.4隐式规则

上面的例子中下面的代码即使不写也可以：



因为根据makefile的隐式规则自动将.c文件编译成.o文件，因此，makefile可以写成这样：





## 5.生成命令

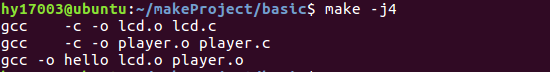
### 5.1 命令的组成

Shell命令组成、tab键开头

### 5.2 命令的执行

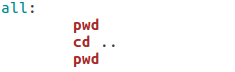
每条命令，make会开一个进程，每条命令执行完，make会检测每个命令的返回码，若命令返回成功，make会继续执行下一个命令，若命令执行出错，make会终止执行当前规则，退出。

### 5.3 并发执行命令

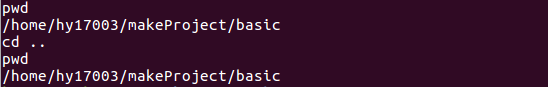


### 5.4 命令同一进程执行

某些场合下，需要几条命令要在同一进程中执行，例如下面的例子



Makefile中，第一条打印当前路径，第二条命令回到父目录，第三条命令再打印当前路径：



从结果可以看出，两条打印路径是相同的，这是因为每条命令都是在独立的进程中执行的，为使各行命令在同一进程中执行，可以这样实现：

