目 录

1 显示命令

makefile 的基本格式如下所示：

```
targets: prerequisites
    command
```

每条命令必须以 tab 键开头，make 会按顺序一条一条地执行命令。

当 make 执行命令时，会在命令执行前将命令行输出到屏幕上。对于命令的显示，有如下三种显示方式：

1. 将“@”字符加在命令行前，那么这条命令就不会被 make 显示出来。
2. 将 make 带参数“-n”或“–just-print”，可以只显示命令，但不执行命令。
3. 将 make 带参数“-s”或“–slient”用于全面禁止命令的显示。

2 命令执行

当一个目标需要被更新时，make 会一条一条地执行其后的命令。

需要注意的是，如果要让上一条命令的结果应用在下一条命令，需要使用分号分隔这两条命令，而不是把这两条命令写在两行上。例子如下：

```
exec:
    cd /home/pengsida
    pwd

exec:
    cd /home/pengsida; pwd
```

以上两个例子的输出结果不同。

3 命令出错

每执行一条指令，make 都会检测每条命令的返回码。如果命令的返回码非零，则说明命令出错，make 将终止所有命令的执行。

有时候，命令出错不代表就是错误的。比如使用 mkdir 创建一个目录。如果目录本身存在，那么 mkdir 将报错。可是我们本来就想要一个目录，不管新旧与否，所以我们就不希望 mkdir 出错而终止 make 的运行。

有两种方法可以忽略命令的出错，如下所示：

1. 在命令行前面加一个减号“-”，如下所示：

```
clean:
    -rm -f *.o
```

2. 给 make 带上“-i”或者“–ignore-errors”参数，这样就会忽略所有命令的错误。

4 定义命令包

makefile 中可以使用“define”和“endef”将命令序列定义为一个变量，例子如下:

```
define run-yacc
    yacc $(firstword $^)
    mv y.tab.c $@
endef
```

上述例子中，“run-yacc”是命令包的名字，使用的方法如下:

```
foo.c: foo.y
    $(run-yacc)
```