<http://www.cnblogs.com/liqiu/archive/2013/03/15/2961518.html>

**静态模式**

最近看Makefile，正常依赖没什么问题，但是遇到一些特殊的符号就有点不明白了，例如：$(objects): %.o: %.c、$<之类的。最近看了篇文章写的不错，贴出来供大家分享。

静态模式可以更加容易地定义多目标的规则，可以让我们的规则变得更加的有弹性和灵活。我们还是先来看一下语法：

<targets ...>: <target-pattern>: <prereq-patterns ...>  
　　<commands>  
...

如果我们的<target-parrtern>定义成“%.o”，意思是我们的<target>集合中都是以“.o”结尾的，而如果我们的<prereq-parrterns>定义成“%.c”，意思是对<target-parrtern>所形成的目标集进行二次定义，其计算方法是，取<target-parrtern>模式中的“%”（也就是去掉了[.o]这个结尾），并为其加上[.c]这个结尾，形成的新集合。所以，我们的“目标模式”或是“依赖模式”中都应该有“%”这个字符。  
  
没看明白没有关系，看下面的例子：

objects = foo.o bar.o

all: $(objects)

$(objects): %.o: %.c

$(CC) -c $(CFLAGS) $< -o $@

上面的例子中，指明了我们的目标从$objects中获取，“%.o”表明要所有以“.o”结尾的目标，也就是“foo.o bar.o”，也就是变量$object集合的模式，而依赖模式“%.c”则取模式“%.o”的“%”，也就是“foo bar”，并为其加下“.c”的后缀，于是，我们的依赖目标就是“foo.c bar.c”。而命令中的“$<”和“$@”则是自动化变量，“$<”表示所有的依赖目标集（也就是“foo.c bar.c”），“$@”表示目标集（也就是“foo.o bar.o”）。于是，上面的规则展开后等价于下面的规则：

foo.o : foo.c

$(CC) -c $(CFLAGS) foo.c -o foo.o

bar.o : bar.c

$(CC) -c $(CFLAGS) bar.c -o bar.o

试想，如果我们的“%.o”有几百个，那种我们只要用这种很简单的“静态模式规则”就可以写完一堆规则，实在是太有效率了。“静态模式规则”的用法很灵活，如果用得好，那会一个很强大的功能。