

计算机学习的第 0 课

```
1 a = $(b)
1 b = hello
2
3 c := world
4 d := $(c)
5 c := test
6
7 all: a c
8
9 a:
10    @echo "a=$(a), b=$(b)"
11
12 c:
13    @echo "c=$(c), d=$(d)"
```

= 可以用来创建递归变量

:= 可以用来创建简单变量

```
jqlg@X1:~/project/var$ make  
a=hello, b=hello  
c=test, d=world  
jqlg@X1:~/project/var$ █
```

```
1 a = $(b)
1 b = hello
2
3 c := world
4 d := $(c)
5 c := test
6
7 e := hello
8 e += world
9
10 f ?= foo
11 f ?= bar
12
13 all: a c e f
14
15 a:
16     @echo "a=$(a), b=$(b)"
17
18 c:
19     @echo "c=$(c), d=$(d)"
20
21 e:
22     @echo "e=$(e)"
23
24 f:
25     @echo "f=$(f)"
26
27
28
```

```
jqlg@X1:~/project/var$ make  
a=hello, b=hello  
c=test, d=world  
e=hello world  
f=foo  
jqlg@X1:~/project/var$ █
```

```
1 objects := hello.o world.o
1 sources := $(objects:.o=.c)
2
3 all: replace
4
5 replace:
6     @echo "objects=$(objects), sources=$(sources)"
```

```
jqlg@X1:~/project/var$ make  
objects=hello.o world.o, sources=hello.c world.c  
jqlg@X1:~/project/var$ █
```

```
1 objects := hello.o world.o
2 sources := $(objects:.o=.c)
3 name := $(shell whoami)
4
5 all: replace sh
6
7 replace:
8     @echo "objects=$(objects), sources=$(sources)"
9
10 sh:
11     @echo "name=$(name)"
12
```

```
jqlg@X1:~/project/var$ whoami  
jqlg  
jqlg@X1:~/project/var$ make  
objects=hello.o world.o, sources=hello.c world.c  
name=jqlg  
jqlg@X1:~/project/var$ █
```

```
1 OBJS := hello.o world.o
2
3 run: $(OBJS)
4     $(CC) -o run $(OBJS)
5
6 $(OBJS): %.o : %.c
7     @echo "target=$@, dependent=$<"
8     cc -o $@ -c $<
9
10 .PHONY: clean
11 clean :
12     -rm $(OBJS) run
```

%.o : %.c - 静态模式规则

模式规则中 % 大体上与Linux Shell 中的 * 等效

```
1 OBJS := hello.o world.o
2
3 run: $(OBJS)
4     $(CC) -o run $(OBJS)
5
6 $(OBJS): %.o : %.c
7     @echo "target=$@, dependent=$<"
8     cc -o $@ -c $<
9
10 .PHONY: clean
11 clean :
12     -rm $(OBJS) run
```

\$@ - 表示目标的文件名

\$< - 表示第一个必要条件的文件名

```
jqlg@X1:~/project/hello$ make  
target=hello.o, dependent=hello.c  
cc -o hello.o -c hello.c  
target=world.o, dependent=world.c  
cc -o world.o -c world.c  
cc -o run hello.o world.o  
jqlg@X1:~/project/hello$ █
```

```
1 objects := hello.o world.o
2
3 CFLAGS := -O2
4 world.o: CFLAGS += -g
5
6 run: $(objects)
7     $(CC) -o $@ $^
8
9 $(objects): %.o:%.c
10    $(CC) -c $(CFLAGS) $< -o $@
11
12 .PHONY: clean
13 clean:
14     -rm $(objects) run
```

\$^ - 表示所有必要条件的文件名，
并且以空格隔开这些文件名

```
jqlg@X1:~/project/hello$ make  
cc -c -O2 hello.c -o hello.o  
cc -c -O2 -g world.c -o world.o  
cc -o run hello.o world.o  
jqlg@X1:~/project/hello$ █
```

```
1 all: cmd1 cmd2  
2  
3 cmd1:  
4     @cd /tmp  
5     @pwd  
6  
7 cmd2:  
8     @cd /tmp ; pwd
```

两条命令之间存在依赖关系时，
需要写在同一行，用分号隔开

```
jqlg@X1:~/project/cmd$ make  
/home/jqlg/project/cmd  
/tmp  
jqlg@X1:~/project/cmd$ █
```

```
1 all: gcc-test cc-test
2
3 gcc-test:
4 ifeq ($(CC), gcc)
5     @echo "gcc"
6 else
7     @echo "Others"
8 endif
9
10 cc-test:
11 ifeq ($(CC), cc)
12     @echo "cc"
13 else
14     @echo "Others"
15 endif
```

```
jqlg@X1:~/project/if$ make  
Others  
cc  
jqlg@X1:~/project/if$ █
```

```
1 foo =
1 bar = $(foo)
2
3 def:
4 ifdef foo
5     @echo "foo is defined"
6 else
7     @echo "foo is not defined"
8 endif
9
10 ifndef bar
11     @echo "bar is defined"
12 else
13     @echo "bar is not defined"
14 endif
```

```
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~
```

```
jqlg@X1:~/project/if$ make  
foo is not defined  
bar is defined  
jqlg@X1:~/project/if$ █
```

```
1 sources1 := he.cllo.c wo.crlld.c
2 objects1 := $(subst .c,.o,$(sources1)) 注意: 参数之间不能有空格
3 all: str_sub1 str_sub2
4
5 str_sub1:
6     @echo "sources1=$(sources1), objects1=$(objects1)"
7
8
9 sources2 := he.cllo.c wo.crlld.c
10 objects2 := $(patsubst %.c,%.o, $(sources2))
11
12 str_sub2:
13     @echo "sources1=$(sources2), objects1=$(objects2)"

~
```

```
jqlg@X1:~/project/string$ make  
sources1=he.cllo.c wo.crld.c, objects1=he.ollo.o wo.orld.o  
sources1=he.cllo.c wo.crld.c, objects1=he.cllo.o wo.crld.o  
jqlg@X1:~/project/string$ █
```

```
1 file_path := /home/jqlg/project/hello/hello.c hello/world.c
2
3 dir:
4     @echo "dir = $(dir $(file_path))"
```

根据路径提取文件的目录

```
jqlg@X1:~/project/file$ make  
dir = /home/jqlg/project/hello/ hello/  
jqlg@X1:~/project/file$ █
```

```
1 file1 := hello.c
2 file2 := world.o
3 file3 := test.txt
4
5 suffix:
6     @echo "suffix = $(suffix $(file1))"
7     @echo "suffix = $(suffix $(file2))"
8     @echo "suffix = $(suffix $(file3))"
```

```
jqlg@X1:~/project/file$ make  
suffix = .c  
suffix = .o  
suffix = .txt  
jqlg@X1:~/project/file$ █
```