一.make 工具

make 工具可以自动完成编译工作。包括：

如果仅修改了某几个源文件，则只重新编译这几个源文件

如果头文件被修改，则重新编译所有包含该头文件的源文件

make 工具通过Makefile文件来完成并自动维护编译工作，Makefile文件描述了整个工程的编译，链接等规则

二.基本规则

TARGET …:DEPENDENCIES..

COMMAND

…..

TARGET（目标）:程序产生的文件，如可执行文件和目标文件

目标也可以是要执行的动作，如clean，也称为伪目标

DEPENDENCIES(依赖)：用来产生目标的输入文件列表，一个目标通常用来依赖于多个文件

例：.PHONY:clean

main:main.o add.o sub.o

gcc main.o add.o sub.o -o main

main.o:main.c add.h sub.h

gcc -c main.c -o main.o

add.o:add.c add.h

gcc -c add.c -o add.o

sub.o:sub.c sub.h

gcc -c sub.c -o sub.o

clean:

rm -f main main.o add.o sub.o

依赖的意思也就意味着，如果依赖的文件列表被修改就重新编译这一项，即执行下面的命令

对于伪目标，通常会加上.PHONY，避免和同名文件冲突。

三.自动化变量

$@ 规则的目标文件名

$< 规则的第一个依赖文件名

$^ 规则的所有依赖文件列表

则上面的可以修改为：

.PHONY:clean

OBJECTS=main.o add.o sub.o

main:$(OBJECTS)

gcc $^ -o main

main.o:main.c add.h sub.h

gcc -c $< -o $@

add.o:add.c add.h

gcc -c $< -o $@

sub.o:sub.c sub.h

@gcc -c $<-o [$@//不再打印gcc -c sub.c](mailto:$@//不再打印gcc%20-c%20sub.c) -o sub.o

clean:

rm -f main $(OBJECTS)

一般将makefile文件，命名为Makefile，make命令会自动寻找该文件，也可用其他命名

如make clean -f Makefile.1 也能执行Makefile.1中的clean命令

在命令前面加一个@将不再打印命令的输出

四.

.PHONY:clean all

BIN=01test 02test

all:$(BIN)//因为要生成all，必须生成BIN，而BIN未指定，所以make默认将同名的.c文件编译为可执行文件

clean:

rm -f \*.o $(BIN)//删除所有的.o文件和01test,02test

这个也能正确生成01test 02test

.PHONY:clean all

BIN=01test 02test

all:$(BIN)

%.o:%.c//表示把对应的.c文件生成对应的.o文件

gcc -Wall -g -c $< -o $@

01test:01test.o

gcc -Wall -g $^ -o $@

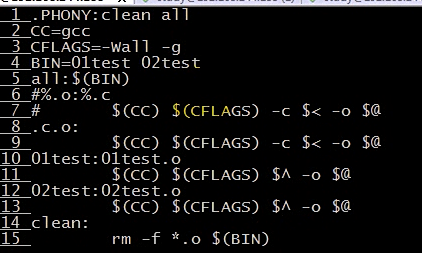
02test:02test.o

gcc -Wall -g $^ -o $@

clean:

rm -f \*.o $(BIN)

上面的%.o:%.c也可以用.c.o:来代替



五.make常用内嵌函数

函数调用

$(function arguments)

$(wildcard PATTERN)

当前目录下匹配模式的文件

如：src=$(wildcard.\*c)//生成所有.c文件保存到src

$(patsubst PATTERN,REPLACEMENT,TEXT)

模式替换函数

如$(patsubst %.c,%.o, %Src)//把.c文件替换成.o文件

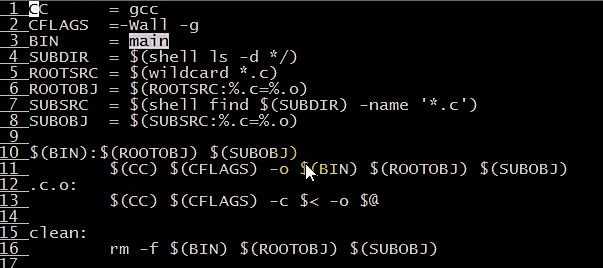
等价于$(src:.c=.o)

shell函数

执行shell命令

例如 $(shell ls -d \*/)

多级目录



SUBDIR：将当前所有的文件夹目录保存在SUBDIR

ROOTSRC：当前文件夹下的所有.c文件

SUBSRC:保存了所有的子文件下的.c文件

SUBOBJ：子文件下的.o文件