一般来说，最简单的就是直接在命令行下输入make命令，make命令会找当前目录的makefile来执行，一切都是自动的。

## **一、make的退出码**

make命令执行后有三个退出码：

0 —— 表示成功执行。

1 —— 如果make运行时出现任何错误，其返回1。

2 —— 如果你使用了make的“-q”选项，并且make使得一些目标不需要更新，那么返回2。

Make的相关参数我们会在后续章节中讲述。

## ****二、指定Makefile****

GNU make找寻默认的Makefile的规则是在当前目录下依次找三个文件——“GNUmakefile”、“makefile”和“Makefile”。

我们也可以给make命令指定一个特殊名字的Makefile。要达到这个功能，我们要使用make的“-f”或是“--file”参数（“-- makefile”参数也行）。例如  
make –f hchen.mk

## ****三、指定目标****

一般来说，make的最终目标是makefile中的第一个目标，而其它目标一般是由这个目标连带出来的。

要使用其他目标，需在make命令后直接跟目标的名字就可以完成（如前面提到的“make clean”形式）

.PHONY: all  
all: prog1 prog2 prog3 prog4  
  
从这个例子中，我们可以看到，这个makefile中有四个需要编译的程序——“prog1”， “prog2”， “prog3”和 “prog4”，我们可以使用“make all”命令来编译所有的目标（如果把all置成第一个目标，那么只需执行“make”）

在Unix世界中，软件发布时，特别是GNU这种开源软件的发布时，其 makefile都包含了编译、安装、打包等功能。我们可以参照这种规则来书写我们的makefile中的目标。

“all” 这个伪目标是所有目标的目标，其功能一般是编译所有的目标。

“clean” 这个伪目标功能是删除所有被make创建的文件。

“install” 这个伪目标功能是安装已编译好的程序，其实就是把目标执行文件拷贝到指定的目标中去。

“print” 这个伪目标的功能是例出改变过的源文件。

“tar”  这个伪目标功能是把源程序打包备份。也就是一个tar文件。

“dist” 这个伪目标功能是创建一个压缩文件，一般是把tar文件压成Z文件。或是gz文件。

“TAGS” 这个伪目标功能是更新所有的目标，以备完整地重编译使用。

“check”和“test” 这两个伪目标一般用来测试makefile的流程。

当然一个项目的makefile中也不一定要书写这样的目标