<https://blog.csdn.net/shenhuxi_yu/article/details/53054020>

**1：尽在Makefile文件的目标项冒号后的另起一行的代码才是shell代码。**

eg：xx=xx1

第一种方式：

xx = xx1         // 这里时makefile代码

第二种方式：  
yy：xx = xx2   // 这是是makefile代码，makefile允许变量赋值时，'='号两边留空格  ？？？？？？可以吗

第三种方式：  
yy：  
    xx=xx3 // 只有这里是shell代码 ，shell不允许‘=’号两边有空格哦。

yy:

    xx= xx3 // 只有这里是shell代码 ，shell不允许‘=’号两边有空格哦。

注意此时xx的值是" xx3"，多了一个空格  
**有一个例外：  
xx=$(shell 这里的代码也是shell代码)  
  
2：Makefile中的shell，每一行是一个进程，不同行之间变量值不能传递。所以，Makefile中的shell不管多长也要写在一行。**

**如果某规则有n个shell命令行构成，而相互之间没有用';'和'\'连接起来的话，就是相互之间没有关联的shell命令，相互之间也不能变量共享。**eg：

SUBDIR=src example

all:

    @for subdir in $(SUBDIR); /      // 这里往下是一行shell

    do/

        echo "building " $$subdir; /

    done

3：Makefile中的变量以$开头，使用$(VAR)或${VAR}来引用变量的定义。 所以，为了避免和shell的变量冲突，shell的变量以$$开头

注意：Makefile中在对一些简单变量的引用，我们也可以不使用“（）”和“{}”来标记变量名，而直接使用“$x”的格式来实现，此种用法仅限于变量名为单字符的情况。另外自动化变量也使用这种格式。对于一般多字符变量的引用必须使用括号了标记，否则make将把变量名的首字母作为作为变量而不是整个字符串（“$PATH”在Makefile中实际上是“$(P)ATH”）。这一点和shell中变量的引用方式不同。shell中变量的引用可以是“${xx}”或者“$xx”格式。但在Makefile中多字符变量名的引用只能是“$(xx)”或者“${xx}”格式。

eg1：从当前目录路径中提取出 /application 或 /base\_class 之前的部分  
PROJECT\_ROOT\_DIR = $(shell pwd | awk -F'/application|/base\_class' '{print $$1}')  
eg2：上例中$$subdir就是shell中的变量， 而$(SUBDIR)是Makefile的中的变量

4.makefile追踪调试命令$(warning "xxxxxx $(PWD)")或者$(error "xxxxxx $(PWD)")，这两个命令放在任意位置都会执行

=============================================================================================

1、在Makefile中只能在target中调用Shell脚本，其他地方是不能输出的。比如如下代码就是没有任何输出：

VAR="Hello"

echo "$(VAR)"

all:

.....

以上代码任何时候都不会输出，没有在target内，如果上述代码改为如下：

VAR="Hello"

all:

echo "$(VAR)"

.....

以上代码，在make all的时候将会执行echo命令。

最后打印结果是:

echo ""Hello""

"Hello"

2、在Makefile中执行shell命令，一行创建一个进程来执行。这也是为什么很多Makefile中有很多行的末尾都是“;  \”，以此来保证代码是一行而不是多行，这样Makefile可以在一个进程中执行，例如：

SUBDIR=src example

all:

@for subdir in $(SUBDIR); \

do\

echo "building "; \

done

上述可以看出for循环中每行都是以”; \”结尾的。

3、Makefile中所有以$打头的单词都会被解释成Makefile中的变量。如果你需要调用shell中的变量（或者正则表达式中锚定句位$），都需要加两个$符号（$$）。实例如下：

PATH="/data/"

all:

echo ${PATH}/\*Makefile中变量，即"/data"\*/

echo $$PATH /\*shell中的变量\*/

例子中的第一个${PATH}引用的是Makefile中的变量，而不是shell中的PATH环境变量，后者引用的事Shell中的PATH环境变量。

     以上三点的是Makefile调用shell应该注意的地方，写Makefile一定要注意。