**参考：**[**http://www.cnblogs.com/Shirlies/p/4282182.html**](http://www.cnblogs.com/Shirlies/p/4282182.html)

[**多文件目录下makefile文件递归执行编译所有c文件**](http://www.cnblogs.com/Shirlies/p/4282182.html)

首先说说本次嵌套执行makefile文件的目的：只需make根目录下的makefile文件，即可编译所有c文件，包括子目录下的。

意义：自动化编译行为，以后编译自己的c文件时可把这些makefile文件直接复制到相应目录即可方便编译出所有文件。这些makefile文件是通用的，只需根据自己的工程情况改动少许内容即可。下面会说。

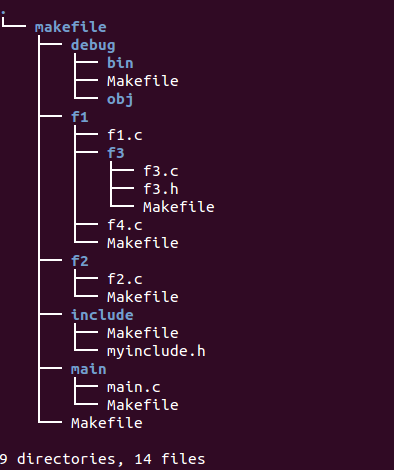
总体思路是：把目标文件放在debug文件夹下的obj目录下，把最终的二进制文件放在debug文件夹下的bin目录下;如何递归编译所有除了debug目录下的makefile文件呢：获得当前目录下的所有子目录，执行子目录下的makefile文件;获取当前目录下的所有c文件，编译c文件并放到指定的目标文件夹下。最后再执行debug目录下的makefile文件生成bin文件。

注意：除了debug文件夹比较特殊外，其他的子目录下都需要有Makefile文件，而且这些Makefile是相同的，除了根目录下的makefile文件有些不同外。即除了bin和obj目录以外的其他目录都需要Makefile文件，即使目录下没有c文件或者其他目录。

过程：首先在根目录下新建一个debug文件夹，debug文件夹下有bin目录和obj目录和一个Makefile文件，结构如下图。（这个debug文件里的makefile文件需要最后执行）（tree工具需要自己安装的，ubuntu下直接输入sudo apt-get install tree即可，但有时可能需要先sudo apt-get update才行）



整个目录结果如下图：



然后在根目录下新建Makefile文件，根目录下也可能会有c文件，故也需处理根目录下的c文件，内容如下：

[复制代码](javascript:void(0);)

#设置编译器

CC=gcc

#debug文件夹里的makefile文件需要最后执行，所以这里需要执行的子目录要排除debug文件夹，这里使用awk排除了debug文件夹，读取剩下的文件夹

SUBDIRS=$(shell ls -l | grep ^d | awk '{if($$9 != "debug") print $$9}')

#无需下一行的注释代码，因为我们已经知道debug里的makefile是最后执行的，所以最后直接去debug目录下执行指定的makefile文件就行，具体下面有注释

#DEBUG=$(shell ls -l | grep ^d | awk '{if($$9 == "debug") print $$9}')

#记住当前工程的根目录路径

ROOT\_DIR=$(shell pwd)

#最终bin文件的名字，可以更改为自己需要的

BIN=myapp

#目标文件所在的目录

OBJS\_DIR=debug/obj

#bin文件所在的目录

BIN\_DIR=debug/bin

#获取当前目录下的c文件集，放在变量CUR\_SOURCE中

CUR\_SOURCE=${wildcard \*.c}

#将对应的c文件名转为o文件后放在下面的CUR\_OBJS变量中

CUR\_OBJS=${patsubst %.c, %.o, $(CUR\_SOURCE)}

#将以下变量导出到子shell中，本次相当于导出到子目录下的makefile中

export CC BIN OBJS\_DIR BIN\_DIR ROOT\_DIR

#注意这里的顺序，需要先执行SUBDIRS最后才能是DEBUG

all:$(SUBDIRS) $(CUR\_OBJS) DEBUG

#递归执行子目录下的makefile文件，这是递归执行的关键

$(SUBDIRS):ECHO

make -C $@

DEBUG:ECHO

#直接去debug目录下执行makefile文件

make -C debug

ECHO:

@echo $(SUBDIRS)

#将c文件编译为o文件，并放在指定放置目标文件的目录中即OBJS\_DIR

$(CUR\_OBJS):%.o:%.c

$(CC) -c $^ -o $(ROOT\_DIR)/$(OBJS\_DIR)/$@

CLEAN:

@rm $(OBJS\_DIR)/\*.o

@rm -rf $(BIN\_DIR)/\*

[复制代码](javascript:void(0);)

上面的注释很详细了，具体的命令如果不清楚，自己可以google一下，譬如：wildcard patsubst awk等

读者可以根据自己的需要更改自己的debug目录和目标文件目录和bin文件目录

其他子目录下的Makefile文件的内容如下：

[复制代码](javascript:void(0);)

1 #子目录的Makefile直接读取其子目录就行

2 SUBDIRS=$(shell ls -l | grep ^d | awk '{print $$9}')

3 #以下同根目录下的makefile的相同代码的解释

4 CUR\_SOURCE=${wildcard \*.c}

5 CUR\_OBJS=${patsubst %.c, %.o, $(CUR\_SOURCE)}

6 all:$(SUBDIRS) $(CUR\_OBJS)

7 $(SUBDIRS):ECHO

8 make -C $@

9 $(CUR\_OBJS):%.o:%.c

10 $(CC) -c $^ -o $(ROOT\_DIR)/$(OBJS\_DIR)/$@

11 ECHO:

12 @echo $(SUBDIRS)

[复制代码](javascript:void(0);)

debug目录下的Makefile文件如下：

1 OBJS=\*.o

2 ODIR=obj

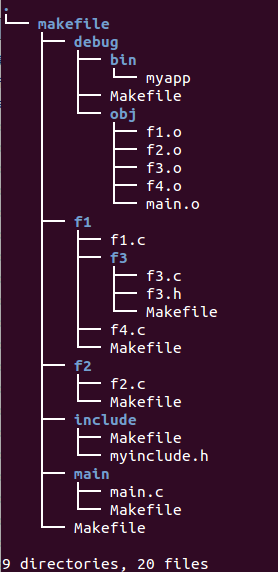
3 $(ROOT\_DIR)/$(BIN\_DIR)/$(BIN):$(ODIR)/$(OBJS)

4 $(CC) -o $@ $^

最后只需在根目录下，我的是我的根目录makefile目录下，执行make命令即可：



结果目录结果为：



然后执行". debug/bin/myapp"即可;最后可以执行make CLEAN清楚掉所有的目标文件和bin文件。

参考资料为：<http://blog.csdn.net/zplove003/article/details/7066595>

关于makefile文件的编写，见一下链接：[跟我一起写makefile](http://blog.csdn.net/haoel/article/details/2886/)和http://wiki.ubuntu.org.cn/index.php?title=%E8%B7%9F%E6%88%91%E4%B8%80%E8%B5%B7%E5%86%99Makefile&variant=zh-cn