http://blog.csdn.net/john_crash/article/details/49889969

Makefile.am将指明工程需要哪些源文件，建造的是什么，如何安装它们。

具体语法如下：

option_where_PRIMARY = targets …

targets是要建造的目标

PRIMARY可以是下面的一个：

可能值	解释
PROGRAMS	目标是可执行程序
LIBRARIES	目标是静态库
LTLIBRARIES	目标是动态库
HEADERS	目标是头文件
SCRIPTS	目标是脚本
DATA	目标是数据

where 表示目标被安装那里，可以是下面的值:

可能项	解释
bin	$(bindir)
lib	$(libdir)
custom	自定义目录
noinst	不安装
check	由’make check’建造。

在**where**前面还可以有一个可选项**option**

dist_ 分发目标(默认)。

nodist_ 不分发。

举例：Makefile.am

```
bin_PROGRAMS = foo run-me
foo_SOURCES = foo.c foo.h print.c print.h
run_me_SOURCES = run.c run.h print.c
```

首先第一句表示产生两个程序foo,run-me，并且将它们安装到bin中。

foo_SOURCES 表示foo需要的源文件。

run_me_SOURCES 表示run-me需要的源文件。

注意：不能转换的符号用’_’代替。

头文件不参加编译，列出来用于分发。automake将自动计算列表对象并编译链接它们。

第二个例子：Makefile.am

```
lib_LIBRARIES = libfoo.a libbar.a
libfoo_a_SOURCES = foo.c privfoo.h
libbar_a_SOURCES = bar.c privbar.h
include_HEADERS = foo.h bar.h
```

这将产生两个静态库文件libfoo.a,libbar.a。

libfoo_a_SOURCES 表明编译libfoo.a需要的源文件。

libbar_a_SOURCES 表明编译libbar.a需要的源文件。

include_HEADERS 表明需要安装的头文件。

也许你在几个目录里面编译，这些目录里面都放置Makefile.am文件。它们必须在configure.ac文件中声明。例如：configure.ac、

```
AC_CONFIG_FILES([Makefile lib/Makefile src/Makefile
```

src/dira/Makefile src/dirb/Makefile])

‘make’ 运行在根目录中。
使用SUBDIRS指定一个递归。
Makefile.am

```
SUBDIRS=lib src
```

src/Makefile.am

```
SUBDIRS = dira dirb
```

记住使用VPATH和$(srcdir)编译，源文件不需要在当前目录。