<https://blog.csdn.net/yanlaifan/article/details/52192977>

gcc作为编译工具，用在Linux操作系统中，可以编译C、C++、Object-C、JAVA等语言。编译过程中可以带编译选项，选择编译过程。

一、GCC编译流程

1）预处理 Pre-Processing  
2）编译 Compiling  
3）汇编 Assembling  
4）链接 Linking

二、GCC编译选项  
  
  
1、gcc总体选项列表  
  
  
1） -c ：指编译，不链接，生成目标文件“.o”。  
2） -S ：只编译，不汇编，生成汇编代码“.S”。  
3） -E ：只进行预编译/预处理，不做其他处理。  
4） -o file：把输出文件输出到file里。  
5） -g ：在可执行程序中包含标准调试信息。  
6） -v ：打印出编译器内部编译各过程的命令行信息和编译器的版本。  
7） -I dir ：在头文件的搜索路径列表中添加dir目录  
8） -L dir ：在库文件的搜索路径列表中添加dir目录  
9） -static ：连接静态库（静态库也可以用动态库链接方式链接）  
10） -llibrary ：连接名为library的库文件（显示指定需要链接的动态库文件）  
  
  
2、gcc告警和出错选项  
1） -ansi ：支持符合ANSI标准的C程序  
2） -pedantic ：允许发出ANSI C标准所列出的全部警告信息  
3） -pedantic-error ：允许发出ANSI C标准所列出的全部错误信息  
4） -w ：关闭所有警告  
5） -Wall ：允许发出gcc提供的所有有用的报警信息  
6） -werror ：把所有的告警信息转化为错误信息，并在告警发生时终止编译过程  
  
  
3、gcc优化选项  
gcc可以对代码进行优化，它通过编译选项“-On”来控制优化代码的生成，其中n是一个代表优化级别的整数。对于不同版本的gcc，  
n的取值范围不一致，比较典型的范围为0变化到2或者3。  
虽然优化选项可以加速代码的运行速度，但对于调试而言将是一个很大的挑战。因为代码在经过优化之后，原先在源程序中声明和使用  
的变量很可能不再使用，控制流也可能会突然跳转到意外的地方，循环语句也可能因为循环展开而变得到处都有。  
  
  
三、GCC生成动态库和静态库  
  
  
1）动态库生成  
1.单个源文件/目标直接生成动态库  
a.   
gcc -fPIC -shared xxx.c -o libxxx.so  
b.  
gcc -fPIC -shared xxx.o -o libxxx.so  
  
  
2.多个源文件/目标生成动态库  
a.  
gcc -fPIC -shared xxx1.c xxx2.c xxx3.c -o libxxx.so   
b.  
gcc -fPIC -shared xxx1.o xxx2.o xxx3.o -o libxxx.so   
  
  
  
2）静态库生成  
1.单个源文件/目标直接生成静态库  
a.  
ar -rc libxxx.a xxx.o（正确方法）  
b. ar -rc libxxx.a xxx.c  （**静态库可以生成；当运行连接了该静态库的可执行程序会报错**：could not read symbols:Archive has no index;run ranlib

to add one）  
  
  
2.多个源文件/目标生成静态库  
a.  
ar -rc libxxx.a xxx1.o xxx2.o xxx3.o （正确方法）  
b.  
ar -rc libxxx.a xxx1.c xxx2.c xxx3.c （**静态库可以生成；当运行连接了该静态库的可执行程序会报错**：could not read symbols:Archive has no index;run ranlib to add one）  
  
  
四、多个源文件生成一个可执行文件  
gcc xxx1.c xxx2.c xxx3.c xxx4.c main.c -o main

实例见博客 gcc编译工具生成动态库和静态库之二----实例

版权声明：本文为博主原创文章，未经博主允许不得转载。 https://blog.csdn.net/yanlaifan/article/details/52192977