"Terminology for C programming"
Paul Johnson Feb. 8, 2007

GNU An organization that creates and advocates for "free software"

gcc The GNU C Compiler, AKA GNU Compiler Collection. A "compiler" is a program that makes programs.

> Example: "gcc -o whatever MyFile.c" will create an executable "./whatever" from a file MyFile.c

Sometimes gcc commands become very complicated because it allows may options. See the output from "gcc –help"

why the ./ The current working directory is not, by default, in the "PATH". The dot-slash symbol means "look for an executable in the current working directory"

binary A nontext file. You can't edit these with a "flat text" editor like Emacs, and generally doing so would destroy them.

object file (*.o) When gcc runs, it converts each file that contains C code into an object file, a binary thing in "machine code. After that, the "linking" process ties those "o" files together into an executable.

make A program that reads a "Makefile" and tells gcc how to compile a program, i.e., what options are required.

library A pre-existing "thing" containing routines and variables that programs can use.

so file (*.so) A "shared object" file. In Linux/Unix, library components are generally built into "so" files. The compiler can "link in" these objects with your program. An "so" file is generally accessed at "run time". In other words, the contents of the "so" file are not gobbled into your program when you compile it. In MS Windows, the equivalent is a DLL file (dynamic link library).

a file (*.a) A library that cannot be loaded at runtime. The compiler has to translate the contents of this library into your program "statically". A statically linked program is bigger, but may run faster. Originally, all C programs were "statically linked" in this way, and the drain on system resources prompted the push for the use of shared object libraries.

header file (*.h) A file that lists the variables and functions available from a library. Almost always, the available header files are found in the filesystem under /usr/include. A program must declare that it intends to use elements from a library by including a reference to the header file.