<https://blog.csdn.net/wang_hufeng/article/details/53899120>

-Wl选项告诉编译器将后面的参数传递给链接器。

-soname则指定了动态库的soname(简单共享名，Short for shared object name)

-Wl 表示后面的参数也就是-soname,libhello.so.1直接传给连接器ld进行处理。实际上，每一个库都有一个soname，当连接器发现它正 在查找的程序库中有这样一个名称，连接器便会将soname嵌入连结中的二进制文件内，而不是它正在运行的实际文件名，在程序执行期间，程序会查找拥有 soname名字的文件，而不是库的文件名，换句话说，soname是库的区分标志。

这样做的目的主要是允许系统中多个版本的库文件共存，习惯上在命名库文件的时候通常与soname相同

libxxxx.so.major.minor

其中，xxxx是库的名字，major是主版本号，minor 是次版本号

简单的来说，soname指定了库的名字，而不去管生成的是什么名字的库，在做连接是将这个soname指定的名字加入执行文件中，而程序运行是也是去加载soname指定的库文件名。所以如果程序连接了新升级的库，只需要将这个新库拷贝到目录下面后，对其以soname做一个符号链接就能调用。

如果库升级了，但是程序依旧使用旧的链接库，那么只需对这个将旧库名字软链接到新升级的库中去即可。

soname的关键功能是它提供了兼容性的标准：

当要升级系统中的一个库时，并且新库的soname和老库的soname一样，用旧库链接生成的程序使用新库依然能正常运行。这个特性使得在**[Linux](http://lib.csdn.net/base/linux" \t "_blank)**下，升级使得共享库的程序和定位错误变得十分容易。

在Linux中，应用程序通过使用soname，来指定所希望库的版本，库作者可以通过保留或改变soname来声明，哪些版本是兼容的，这使得程序员摆脱了共享库版本冲突问题的困扰。

可以通过readelf -d来查看每个动态库的SONAME

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/wang\_hu\_feng/article/details/53899120