# 计算机操作系统 - 链接

* [计算机操作系统 - 链接](#计算机操作系统---链接)
	+ [编译系统](#编译系统)
	+ [静态链接](#静态链接)
	+ [目标文件](#目标文件)
	+ [动态链接](#动态链接)

## 编译系统

以下是一个 hello.c 程序：

#include <stdio.h>

int main()
{
 printf("hello, world\n");
 return 0;
}

在 Unix 系统上，由编译器把源文件转换为目标文件。

gcc -o hello hello.c

这个过程大致如下：

* 预处理阶段：处理以 # 开头的预处理命令；
* 编译阶段：翻译成汇编文件；
* 汇编阶段：将汇编文件翻译成可重定位目标文件；
* 链接阶段：将可重定位目标文件和 printf.o 等单独预编译好的目标文件进行合并，得到最终的可执行目标文件。

## 静态链接

静态链接器以一组可重定位目标文件为输入，生成一个完全链接的可执行目标文件作为输出。链接器主要完成以下两个任务：

* 符号解析：每个符号对应于一个函数、一个全局变量或一个静态变量，符号解析的目的是将每个符号引用与一个符号定义关联起来。
* 重定位：链接器通过把每个符号定义与一个内存位置关联起来，然后修改所有对这些符号的引用，使得它们指向这个内存位置。

## 目标文件

* 可执行目标文件：可以直接在内存中执行；
* 可重定位目标文件：可与其它可重定位目标文件在链接阶段合并，创建一个可执行目标文件；
* 共享目标文件：这是一种特殊的可重定位目标文件，可以在运行时被动态加载进内存并链接；

## 动态链接

静态库有以下两个问题：

* 当静态库更新时那么整个程序都要重新进行链接；
* 对于 printf 这种标准函数库，如果每个程序都要有代码，这会极大浪费资源。

共享库是为了解决静态库的这两个问题而设计的，在 Linux 系统中通常用 .so 后缀来表示，Windows 系统上它们被称为 DLL。它具有以下特点：

* 在给定的文件系统中一个库只有一个文件，所有引用该库的可执行目标文件都共享这个文件，它不会被复制到引用它的可执行文件中；
* 在内存中，一个共享库的 .text 节（已编译程序的机器代码）的一个副本可以被不同的正在运行的进程共享。