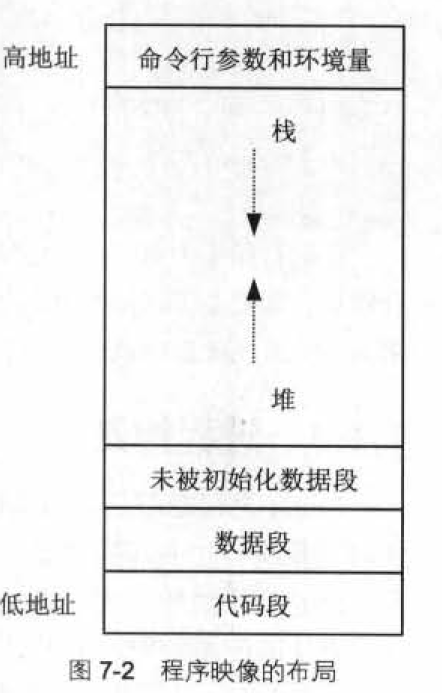
# Linux C++后台开发知识总结

### 系统编程的基本概念

1. 系统调用：为了从操作系统获得资源或服务，从**用户空间**向**内核**发起的函数调用。为什么需要系统调用？这是因为操作系统不允许用户空间的应用程序直接操作内核的数据和或执行代码，内核为了让用户空间的应用程序间接操作其数据和执行代码，提供了系统调用机制。系统调用机制在不同的操作系统上实现是不同的，比如，可以通过发送中断通知内核，需要进行系统调用。编程时，编译器和C函数库会自动处理关于系统调用的部分。
2. glibc: C库是linux系统编程的核心，linux的C函数库由GNUlibc提供，包含了标准C库，系统调用封装、线程支持和基本应用工具。
3. C编译器：gcc辅助实现了标准C和系统ABI，因此linux系统编程和编译器连接的也十分紧密，一般都是gcc。
4. 文件和文件系统：Linux遵循一切皆文件的理念，文件是Linux中最重要的抽象概念。一个“文件”必须被打开，才能被读取和写入。文件可以以只读、只写、读写、追加等模式打开，并通过文件描述符来引用，进行读写操作，操作完成后关闭文件。
5. 普通文件：普通文件的本质是一系列线性的字节流。
6. 文件元数据：元数据即自我描述的数据，文件的元数据包括文件的一系列属性，如文件权限、大小、时间，以及文件数据的位置等等，**唯独没有文件名**，Linux通过iNode来存储每个文件的元数据，iNode为一系列整数编号。
7. 目录和链接：访问一个文件是通过文件名来访问的。目录就是用来保存文件名以及文件名到iNode链接的文件，其在磁盘上的物理形式可以是个哈希表或其他表格，文件名到iNode的映射，称为链接。目录本身也是个文件，因此目录本身也有个iNode与之对应，目录的链接除了包含文件名到其iNode的映射外，还包含到其它目录iNode的映射，因而可以形成层次结构。而通过一系列目录层次寻找到对应文件的iNode，则成为路径解析，从根目录开始，则成为绝对路径，从当前目录开始，成为相对路径。内核不允许像操作普通文件那样操作目录，只允许通过系统调用来添加和删除链接。
8. 特殊文件：块设备文件，字符设备文件，命名管道，Unix域套接字。块设备文件按照字节数组的方式进行访问，设备驱动将字节映射到可寻址的设备上，可以读写字节数组中任一位置的字节，常见的块设备文件有硬盘，软盘等。字符设备文件按照线性字节队列的方式进行访问，设备驱动将字节写入缓存，应用程序则从缓存中读取走相应字节，常见的字符设备文件有键盘。命名管道FIFOs是通过文件描述符来进行进程间通信的机制，普通管道是指将一个程序的输出通过管道作为另一个程序的输入，这个管道是通过系统调用在内存中创建。命名管道域普通管道类似，不过命名管道是通过文件来进行访问。套接字文件是进程间通信的一种高级形式，允许不同的进程进行通信。
9. 文件系统：以合法层次结构组织的目录和文件的集合，称为文件系统，包括普通的存储于物理磁盘上的物理文件系统，存储于内存的虚拟文件系统，以及网络文件系统。块设备文件的最小访问单位是扇区，一般为512B，这是由硬盘结构确定的，必须为2整数倍。文件系统的最小逻辑单元是块，块是文件系统的抽象，而不是物理的单位，块的大小一般是2的指数倍乘以扇区大小，比如512B，1024B，4KB等。对物理磁盘的访问有两种方式，通过设备驱动和虚拟文件系统，第一种方式绕过了文件系统直接操作磁盘数据，一般通过第二种方式进行访问。VFS存在于内存中，通过对各种文件系统进行整合，提供统一的编程接口给上层应用程序使用。
10. 进程：进程是目标代码的一次执行过程，即执行中的目标代码。进程是操作系统资源调度的基本单位。一个C程序经过预编译，编译，汇编几个步骤生产目标文件，再经过链接，将目标代码生成为可执行文件(linux下为ELF)，一个可执行文件包含代码段，数据段和bss段。其中代码段包含可执行的代码，数据段包含已初始化的数据，bss段包含未初始化的数据。内核将可执行代码，加载到内存，为程序分配内存空间(堆栈空间)，为程序分配进程标识符PID及其他资源，保存进程PID和状态信息，将其放入运行队列等待运行。
11. 进程的内存映像：内核在内存中如何存放一个程序。与可执行代码不同的是，进程除了代码段、数据段、bss段还有堆栈段和环境变量、命令行参数。
12. 线程：