# 3.7 守护进程

//一：普通进程运行观察

//ps -eo pid,ppid,sid,tty,pgrp,comm,stat,cmd | grep -E 'bash|PID|nginx'

//a)进程有对应的终端，如果终端退出，那么对应的进程也就消失了；它的父进程是一个bash

//b)终端被占住了，你输入各种命令这个终端都没有反应；

//二：守护进程基本概念

//守护进程 一种长期运行的进程：这种进程在后台运行，并且不跟任何的控制终端关联；

//基本特点：

//a)生存期长[不是必须，但一般应该这样做]，一般是操作系统启动的时候他就启动，操作系统关闭的时候他才关闭；

//b)守护进程跟终端无关联，也就是说他们没有控制终端，所以你控制终端退出，也不会导致守护进程退出；

//c)守护进程是在后台运行,不会占着终端，终端可以执行其他命令

//linux操作系统本身是有很多的守护进程在默默的运行，维持着系统的日常活动。大概30-50个；

//a)ppid = 0：内核进程，跟随系统启动而启动，声明周期贯穿整个系统；

//b)cmd列名字带[]这种，叫内核守护进程；

//c)老祖init：也是系统守护进程,它负责启动各运行层次特定的系统服务；所以很多进程的PPID是init。而且这个init也负责收养孤儿进程；

//d)cmd列中名字不带[]的普通守护进程（用户级守护进程）

//共同点总结：

//a)大多数守护进程都是以超级 用户特权运行的；

//b)守护进程没有控制终端，TT这列显示?

//内核守护进程以无控制终端方式启动

//普通守护进程可能是守护进程调用了setsid的结果（无控制端）；

//三：守护进程编写规则

//(1)调用umask(0);

//umask是个函数，用来限制（屏蔽）一些文件权限的。

//(2)fork()一个子进程(脱离终端)出来,然后父进程退出( 把终端空出来，不让终端卡住)；固定套路

//fork()的目的是想成功调用setsid()来建立新会话，目的是

//子进程有单独的sid；而且子进程也成为了一个新进程组的组长进程；同时，子进程不关联任何终端了；

//--------------讲解一些概念

//（3.1）文件描述符：正数，用来标识一个文件。

//当你打开一个存在的文件或者创建一个新文件，操作系统都会返回这个文件描述符（其实就是代表这个文件的），

//后续对这个文件的操作的一些函数，都会用到这个文件描述符作为参数；

//linux中三个特殊的文件描述符，数字分别为0,1,2

//0:标准输入【键盘】，对应的符号常量叫STDIN\_FILENO

//1:标准输出【屏幕】，对应的符号常量叫STDOUT\_FILENO

//2:标准错误【屏幕】，对应的符号常量叫STDERR\_FILENO

//类Unix操作系统，默认从STDIN\_FILENO读数据，向STDOUT\_FILENO来写数据，向STDERR\_FILENO来写错误；

//类Unix操作系统有个说法：一切皆文件,所以它把标准输入，标准输出，标准错误 都看成文件。

//与其说 把 标准输入，标准输出，标准错误 都看成文件 到不如说

//象看待文件一样看待 标准输入，标准输出，标准错误

//象操作文件一样操作 标准输入，标准输出，标准错误

//同时，你程序一旦运行起来，这三个文件描述符0,1,2会被自动打开(自动指向对应的设备)；

//文件描述符虽然是数字，但是，如果我们把文件描述符直接理解成指针（指针里边保存的是地址——地址说白了也是个数字）；

//write(STDOUT\_FILENO,"aaaabbb",6);

//（3.2）输入输出重定向

//输出重定向：我标准输出文件描述符，不指向屏幕了，假如我指向（重定向）一个文件

//重定向，在命令行中用 >即可；

//输入重定向 <

//（3.3）空设备（黑洞）

// /dev/null ：是一个特殊的设备文件，它丢弃一切写入其中的数据（象黑洞一样）；

//----

//守护进程虽然可以通过终端启动，但是和终端不挂钩。

//守护进程是在后台运行，它不应该从键盘上接收任何东西，也不应该把输出结果打印到屏幕或者终端上来

//所以，一般按照江湖规矩，我们要把守护进程的 标准输入，标准输出，重定向到 空设备（黑洞）；

//从而确保守护进程不从键盘接收任何东西，也不把输出结果打印到屏幕；

//int fd;

//fd = open("/dev/null",O\_RDWR) ;//打开空设备

//dup2(fd,STDIN\_FILENO); //复制文件描述符 ，像个指针赋值,把第一个参数指向的内容赋给了第二个参数；

//dup2(fd,STDOUT\_FILENO);

//if(fd > STDERR\_FILENO)

// close(fd); //等价于fd = null;

//（3.4）实现范例

//守护进程可以用命令启动，如果想开机启动，则需要借助 系统初始化脚本来启动。

//四：守护进程不会收到的信号：内核发给你，另外的进程发给你的；

//（4.1）SIGHUP信号

//守护进程不会收到来自内核的 SIGHUP 信号； 潜台词就是 如果守护进程收到了 SIGHUP信号，那么肯定是另外的进程发给你的；

//很多守护进程把这个信号作为通知信号，表示配置文件已经发生改动，守护进程应该重新读入其配置文件；

//4.2）SIGINT、SIGWINCH信号

//守护进程不会收到来自内核的 SIGINT（ctrl+C),SIGWINCH(终端窗口大小改变) 信号；

//五：守护进程和后台进程的区别

//(1)守护进程和终端不挂钩；后台进程能往终端上输出东西(和终端挂钩)；

//(2)守护进程关闭终端时不受影响，守护进程会随着终端的退出而退出；

//(3)......其他的，大家自己总结；