0103信号

1. 信号
2. 什么是信号
3. 为什么要有信号
4. 信号如何使用

信号和信号量没有关系！

产生前
如何产生

信号发送中

信号处理
&&
处理完成

什么是Linux信号？

本质是一种**通知**机制，用户 or OS通过发送一定的信号，通知**进程**，某些事件已经发生，你可以再**后续进行处理！**

结合进程，信号的一些结论：

1. 进程要处理信号，必须具备信号“识别”信号的能力（看到+处理）
2. 凭什么进程能够“识别”信号呢？谁“教育”进程的呢？OS，本质就是写OS处理进程部分的程序员
3. 信号产生是随机的，信号还没来的时候，进程可能正在忙自己的事情，所以信号的后续处理，可能不是立即处理的
4. 信号会临时的记录下对应的信号，方便后续进行处理
5. 在什么时候处理呢？合适的时候
6. 一般而言，信号的产生相对于进程而言是异步的！

信号如何产生

```
(base) [yufc@VM-12-12-centos:~/Files/SYSU]$ kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL       5) SIGTRAP
 6) SIGABRT      7) SIGBUS       8) SIGFPE       9) SIGKILL     10) SIGUSR1
11) SIGSEGV     12) SIGUSR2     13) SIGPIPE     14) SIGALRM     15) SIGTERM
16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO       30) SIGPWR
31) SIGSYS      34) SIGRTMIN    35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3
38) SIGRTMIN+4  39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
58) SIGRTMAX-6  59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
63) SIGRTMAX-1  64) SIGRTMAX
(base) [yufc@VM-12-12-centos:~/Files/SYSU]$
```

[1,31]这些信号
叫做普通信号

[34,64]叫做实时信号

Ctrl + C 的本质就是向目标进程发送2号信号

信号处理的常见方式

1. 默认（进程自带的，程序员写好的逻辑）
2. 忽略（也是信号处理的一种方式）
3. 自定义动作（捕捉信号）

SIGHUP	1	Term	Hangup detected on controlling terminal or death of controlling process
SIGINT	2	Term	Interrupt from keyboard
SIGQUIT	3	Core	Quit from keyboard
SIGILL	4	Core	Illegal Instruction
SIGABRT	6	Core	Abort signal from **abort**(3)
SIGFPE	8	Core	Floating point exception
SIGKILL	9	Term	Kill signal
SIGSEGV	11	Core	Invalid memory reference
SIGPIPE	13	Term	Broken pipe: write to pipe with no readers
SIGALRM	14	Term	Timer signal from **alarm**(2)
SIGTERM	15	Term	Termination signal
SIGUSR1	30,10,16	Term	User-defined signal 1
SIGUSR2	31,12,17	Term	User-defined signal 2
SIGCHLD	20,17,18	Ign	Child stopped or terminated
SIGCONT	19,18,25	Cont	Continue if stopped
SIGSTOP	17,19,23	Stop	Stop process
SIGTSTP	18,20,24	Stop	Stop typed at terminal
SIGTTIN	21,21,26	Stop	Terminal input for background process
SIGTTOU	22,22,27	Stop	Terminal output for background process

man 7 signal
可以查看信号的描述

如何理解组合键变成信号呢？

如何理解信号被进程保存呢？

如何理解信号发送的本质？

如何理解组合键变成信号呢？

如何理解信号被进程保存呢？

如何理解信号发送的本质？

OS解释组合键 --- 查找进程列表 --- 前台运行的进程 --- OS写入对应的信号到进程内部的位图结构当中

键盘的工作方式是通过：中断方式进行的
当然也能够识别组合键

如何理解信号被进程保存呢？

1. 什么信号？
2. 是否产生？

进程 必须具有保存信号的相关数据结构**（位图）**
这个位图在哪呢？**PCB内部！**

理解信号发送的本质：

信号位图是在task_struct里面保存的，谁能改pcb，只有OS有这个资格！因为pcb是内核数据结构
本质：OS向目标进程写**信号**！
OS直接修改pcb中指定位图结构，完成“发送”信号的过程！