<https://www.cnblogs.com/zhanggaofeng/p/6075725.html>

信号列表

[复制代码](javascript:void(0);)

SIGABRT 进程停止运行 6

SIGALRM 警告钟

SIGFPE 算述运算例外

SIGHUP 系统挂断

SIGILL 非法指令

SIGINT 终端中断 2

SIGKILL 停止进程（此信号不能被忽略或捕获）

SIGPIPE 向没有读的管道写入数据

SIGSEGV 无效内存段访问

SIGQOUT 终端退出 3

SIGTERM 终止

SIGUSR1 用户定义信号1

SIGUSR2 用户定义信号2

SIGCHLD 子进程已经停止或退出

SIGCONT 如果被停止则继续执行

SIGSTOP 停止执行

SIGTSTP 终端停止信号

SIGTOUT 后台进程请求进行写操作

SIGTTIN 后台进程请求进行读操作

[复制代码](javascript:void(0);)

[复制代码](javascript:void(0);)

typedef void (\*sighandler\_t)(int);

sighandler\_t signal(int signum, sighandler\_t handler);

signal函数

作用1：站在应用程序的角度，注册一个信号处理函数

作用2：忽略信号，设置信号默认处理 信号的安装和回复

参数

--signal是一个带signum和handler两个参数的函数，准备捕捉或屏蔽的信号由参数signum给出，接收到指定信号时将要调用的函数有handler给出

--handler这个函数必须有一个int类型的参数（即接收到的信号代码），它本身的类型是void

--handler也可以是下面两个特殊值：① SIG\_IGN 屏蔽该信号 ② SIG\_DFL 恢复默认行为

[复制代码](javascript:void(0);)

[复制代码](javascript:void(0);)

//忽略，屏蔽信号  
#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <errno.h>

#include <sys/types.h>

#include <signal.h>

int main(int arg, char \*args[])

{

pid\_t pid=fork();

if(pid==-1)

{

printf("fork() failed! error message:%s\n",strerror(errno));

return -1;

}

//注册信号，屏蔽SIGCHLD信号，子进程退出，将不会给父进程发送信号，因此也不会出现僵尸进程

signal(SIGCHLD,SIG\_IGN);

if(pid>0)

{

printf("father is runing !\n");

sleep(10);

}

if(pid==0)

{

printf("i am child!\n");

exit(0);

}

printf("game over!\n");

return 0;

}

[复制代码](javascript:void(0);)

[复制代码](javascript:void(0);)

//恢复信号

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <errno.h>

#include <sys/types.h>

#include <signal.h>

void catch\_signal(int sign)

{

switch (sign)

{

case SIGINT:

printf("ctrl + C 被执行了!\n");

//exit(0);

break;

}

}

int main(int arg, char \*args[])

{

//注册终端中断信号

signal(SIGINT, catch\_signal);

char tempc = 0;

while ((tempc = getchar()) != 'a')

{

printf("tempc=%d\n", tempc);

//sleep()

}

//恢复信号

signal(SIGINT, SIG\_DFL);

while (1)

{

pause();

}

printf("game over!\n");

return 0;

}

[复制代码](javascript:void(0);)

[复制代码](javascript:void(0);)

//signal()函数的返回值

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <errno.h>

#include <sys/types.h>

#include <signal.h>

void catch\_signal(int sign)

{

switch (sign)

{

case SIGINT:

printf("ctrl + C 被执行了!\n");

//exit(0);

break;

}

}

int main(int arg, char \*args[])

{

/\*

\* signal()函数的返回值是signal()函数上一次的行为

\* \*/

typedef void (\*sighandler\_t)(int);

//因为第一次注册信号SIGINT,所以上一次的行为就是默认行为

sighandler\_t old=signal(SIGINT, catch\_signal);

if(old==SIG\_ERR)

{

//注册信号失败

perror("signal error");

}

/\*正规写法\*/

if(signal(SIGQUIT,catch\_signal)==SIG\_ERR)

{

//注册新号失败

perror("signal error");

}

char tempc = 0;

while ((tempc = getchar()) != 'a')

{

printf("tempc=%d\n", tempc);

//sleep()

}

//把默认行为重新注册，不就是恢复默认信号了

signal(SIGINT, old);

while (1)

{

pause();

}

printf("game over!\n");

return 0;

}