리눅스 시스템 프로그래밍

day6 프로세스 실습

고 강 태
010-8269-3535
james@thinkbee.kr
https://www.linkedin.com/in/thinkbeekr/
http://www.facebook.com/gangtai.goh

sid 와 pid

sid 와 프로세스

프로세스의 세션 식별 번호를 구하거나, 새로운 세션을 생성한다

```
#include <sys/types.h>
#include <unistd.h>

pid_t getsid(pid_t pid);
 - pid: 프로세스 식별 번호이다.
 - return: 호출 성공 SID, 실패 -1
```

pid=0 현재 프로세스에 대해 알아온다.

세션 (session):

- 일반적으로 시스템과 연결된 하나의 제어 단말기를 포함한 단위
- 식별 번호(id)가 부여되어 있다.
- 세션 > 그룹 > 프로세스
- 세션의 리더 (프로세스): 자신의 PID = PGID = PSID 일 경우

sid 테스트

day5/sid_ex.c

```
#include <sys/types.h> #include <unistd.h>

main(int argc, char *argv[]) {
    pid_t pid;
    int interval;

    if(argc != 3)    exit(1);

    pid = atoi(argv[1]);
    interval = atoi(argv[2]);

    printf("shell process...\n");
    printf("process id:%d, group id:%d, session id:%d\n",
        pid, getpgid(pid), getsid(pid));
    printf("current process.. not daemon...\n");
    printf("process id:%d, group id:%d, session id:%d\n",
        getpid(), getpgrp(), getsid(0));

    sleep(interval);
}
```

sid 테스트

day5/sid_ex.c

```
$ ps
   PID TTY          TIME CMD
  5466 pts/1    00:00:00 bash
 11030 pts/1    00:00:00 ps
$ ./sid_ex 5466 0
shell process...
process id:5466, group id:5466, session id:5466
current process.. not daemon...
process id:11032, group id:11032, session id:5466
```

sid 테스트

day5/sid_ex.c

```
$ ps
   PID TTY          TIME CMD
  5466 pts/1    00:00:00 bash
 11038 pts/1    00:00:00 ps
$ ./sid_ex 5466 600 &
[1] 11040
$ shell process...
process id:5466, group id:5466, session id:5466
current process.. not daemon...
process id:11040, group id:11040, session id:5466

$ ./sid_ex 5466 600 &
[2] 11042
$ shell process...
process id:5466, group id:5466, session id:5466
current process.. not daemon...
process id:11042, group id:11042, session id:5466

$ ps
   PID TTY          TIME CMD
  5466 pts/1    00:00:00 bash
 11040 pts/1    00:00:00 sid_ex
 11042 pts/1    00:00:00 sid_ex
 11043 pts/1    00:00:00 ps
$ exit
```

로그아웃후 확인해 보자

setsid()

day5/sid_ex.c

로그아웃후 다시 로그인해서 sid_ex 프로세스를 길게 출력해 보자

```
~$ ps -ef | grep sid
qkboo     19079   1579  0 09:30 ?        00:00:00 ./sid_ex 2169 600
qkboo     19080   1579  0 09:30 ?        00:00:00 ./sid_ex 2169 600
qkboo     19200  19188  0 09:31 pts/2    00:00:00 grep --color=auto sid

~$ kill -9 19079
qkboo@ubuntu:~$ ps -ef | grep sid
qkboo     19080   1579  0 09:30 ?        00:00:00 ./sid_ex 2169 600
qkboo     19213  19188  0 09:33 pts/2    00:00:00 grep --color=auto sid
~$ kill -9 19080
~$ ps -ef | grep sid
qkboo     19216  19188  0 09:33 pts/2    00:00:00 grep --color=auto sid
```

kill 명령으로 죽인다

setsid()

데몬 프로세스: setsid()로 데몬 프로세스로 만들 수 있다.

```
#include <sys/types.h>
#include <unistd.h>

pid_t setsid(void);
 - 그룹리더가 아니면 새 세션 생성해 세션과 그룹 리더가 된다.
```

데몬 프로세스가 된다

setsid()

데몬 프로세스

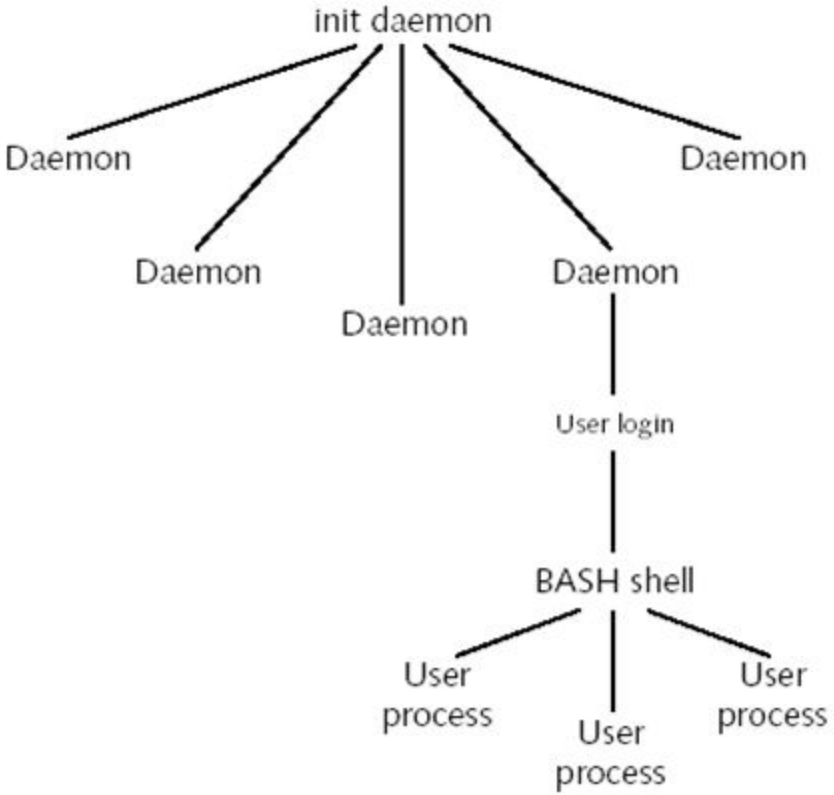

Figure 9-2: Process genealogy

setsid()

day5/sid_set.c

데몬 프로세스: setsid()로 데몬 프로세스로 만들 수 있다.

```c
#include <sys/types.h> #include <unistd.h>

main() {
    pid_t pid;

    if((pid = fork()) > 0) {
        sleep(1);
        exit(1);
    }
    else if(pid == 0) {
        printf("old session id: %d\n", getsid(0));
        printf("new session id: %d\n", setsid());
        sleep(600);
    }
}
```

setsid()

day5/sid_set.c

데몬 프로세스: setsid()로 데몬 프로세스로 만들 수 있다.

```
$ ./sid_set
old session id: 11525
new session id: 11616

$ ps -ef | grep sid
qkboo     11616      1  0 01:13 ?        00:00:00 ./sid_set
qkboo     11622  11525  0 01:13 pts/1    00:00:00 grep --color=auto
sid
```

터미널을 끊고 다시 로그인해 확인한다.

kill PID 로 종료 할 수 있다

실습:

실습: ex_dircheck.c

프로그램에 경로를 주고 경로가 디렉토리인지 아닌지 확인하도록 한다.

다음 중 하나를 선택해 디렉토리를 점검한다.

- DIR 구조체
- fopen()
- stat 구조체

실습: ex_wc.c

리눅스 명령 wc 는 파일의 문장의 줄수, 단어수, 문자수를 출력해 준다. 또한 여러 파일을 전달받아, 각 파일을 자식 프로세스에서 "wc" 명령을 수행해 처리하도록 작성해 보자

실행 예)

```
$ ./ex_wc a.txt b.txt c.txt
10 25 205  a.txt
32 59 290  b.txt
24 45 201  c.txt
```

자식 프로세스에서 wc 명령을 사용한다

실습: ex_fork_sum.c

두 양수를 입력받아 다음 작업을 만들어 보세요.

1. 부모 프로세스
   - x의 y제곱을 계산한다.

2. 자식 프로세스
   - x에서 y까지 합을 계산한다.

3. 두 값의 합을 출력한다.

실습: fork_curtime.c

자식 프로세스에서 10초 간격으로 현재 시각을 파일
"current_time.txt" 에 저장하는 프로그램을 작성하시오.
- 단, 세션이 종료되도 프로그램이 실행되도록 한다.

데몬 프로세스

실습: Sample

실습: ex_dircheck.c

두 수를 입력받아 부모와 자식 프로세스에서 계산한다.

```c
#include <unistd.h> #include <dirent.h> #include <sys/types.h>

int isDir(char *);

int main(int argc, char *argv[]) {
    if(argc != 2)    return;

    if(isDir(argv[1]))
        printf("%s not exist\n", argv[1]);
    else
        printf("%s exist\n", argv[1]);
}

int isDir(char *path)  {
    DIR *dirp;
    dirp = opendir(path);
    closedir(dirp);
    return dirp == NULL ? 1 : 0;
}
```

실습: ex_wc.c

두 수를 입력받아 부모와 자식 프로세스에서 계산한다.

```c
int main(int argc, char *argv[]) {
    pid_t pid;
    char filename[64];
    int n;

    printf("\n");
    for(n = 1; n < argc; n++)     {
        strcpy(filename, argv[n]);
        printf("run \"wc %s\"\n", filename);
        pid = fork();
        if(pid == 0)    // child
        {
            execlp("wc", "wc", filename, (char *)0);
            //printf("wc %s\n", filename);
            exit(1);    // if fail to run execlp
        }
    }
}
```

실습: ex_fork_sum.c (1)

두 수를 입력받아 부모와 자식 프로세스에서 계산한다.

```c
int main() {
    int x, y, i;
    pid_t pid;
    int c_result = 0, p_result = 1;

    printf("input two numbers (1~10) : ");
    scanf("%d %d", &x, &y);

    pid = fork();

    if(pid > 0)
    {
        for(i = 0; i < y; i++)
            p_result *= x;
        printf("parent : %d\n", p_result);
        wait(&c_result);
        c_result = c_result >> 8;
        printf("result is %d\n", p_result + c_result);
    }
```

실습: ex_fork_sum.c (2)

두 양수를 입력받아 다음 작업을 만들어 보세요.

```
int main() {
    ...

    else if(pid == 0)
    {
        for(i = x; i <= y; i++)
            c_result += i;
        printf("child : %d\n", c_result);
        exit(c_result);
    }
}
```

실습: ex_fork_curtime.c (1)

파일에 10초 간격으로 현재 시각을 저장하는 자식 프로세스를 만든다.

```c
#include <unistd.h>
#include <time.h> #include <sys/types.h> #include <fcntl.h>

int main()  {
    time_t result;
    pid_t pid;

    int filedes;
    ssize_t nread;
    char buffer[32];

    pid = fork();

    if(pid > 0) {
        sleep(1);
        exit(1);
    }
```

실습: ex_fork_curtime.c (2)

파일에 10초 간격으로 현재 시각을 저장하는 자식 프로세스를 만든다.

```
    else if(pid == 0) {
        setsid();
        filedes = open("./current_time.txt",
                        O_RDWR | O_CREAT, 0644);
        for(;;)
        {
            result = time(NULL);
            strcpy(buffer, asctime(localtime(&result)));
            printf("%s", buffer);
            write(filedes, buffer, strlen(buffer));
            sleep(2);
        }
    }
}
```