이번 포스팅에서는 리눅스 전용 작업 스케줄러인 crontab(크론탭?)에 대해 알아보자. 특정 시간을 주기로 원하는 작업을 계속 실행시켜 주는 개꿀 프로그램이다. 크론탭에 등록된 작업들은 커맨드 창(Shell)이 켜져 있든, 말든 동작하기 때문에 서버 등을 운영할 때 편리한 기능이기도 하다.

리눅스라면 왠만하면 기본적으로 설치되어 있다고 생각하기 때문에 설치 관련 설명은 생략하겠다.

## 현재 크론탭에 등록된 스케줄 확인

```
$ crontab -l
```

현재 등록되어 있는 스케줄을 확인한다. 하나도 등록되지 않았다면, no crontab for <user명>을 출력하며 끝나버리지만 예약된 스케줄이 있다면 아래와 같이 생긴 예약된 스케줄을 확인할 수 있다.

```
ubuntu@ip-172-31-26-227:~$ crontab -l
# Edit this file to introduce tasks to be run by cron.
.....<중략>
# m h  dom mon dow   command
SHELL=/bin/bash
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin
// 크론탭 스케줄 목록
0 */6 * * * python3 /home/ubuntu/main.py
```

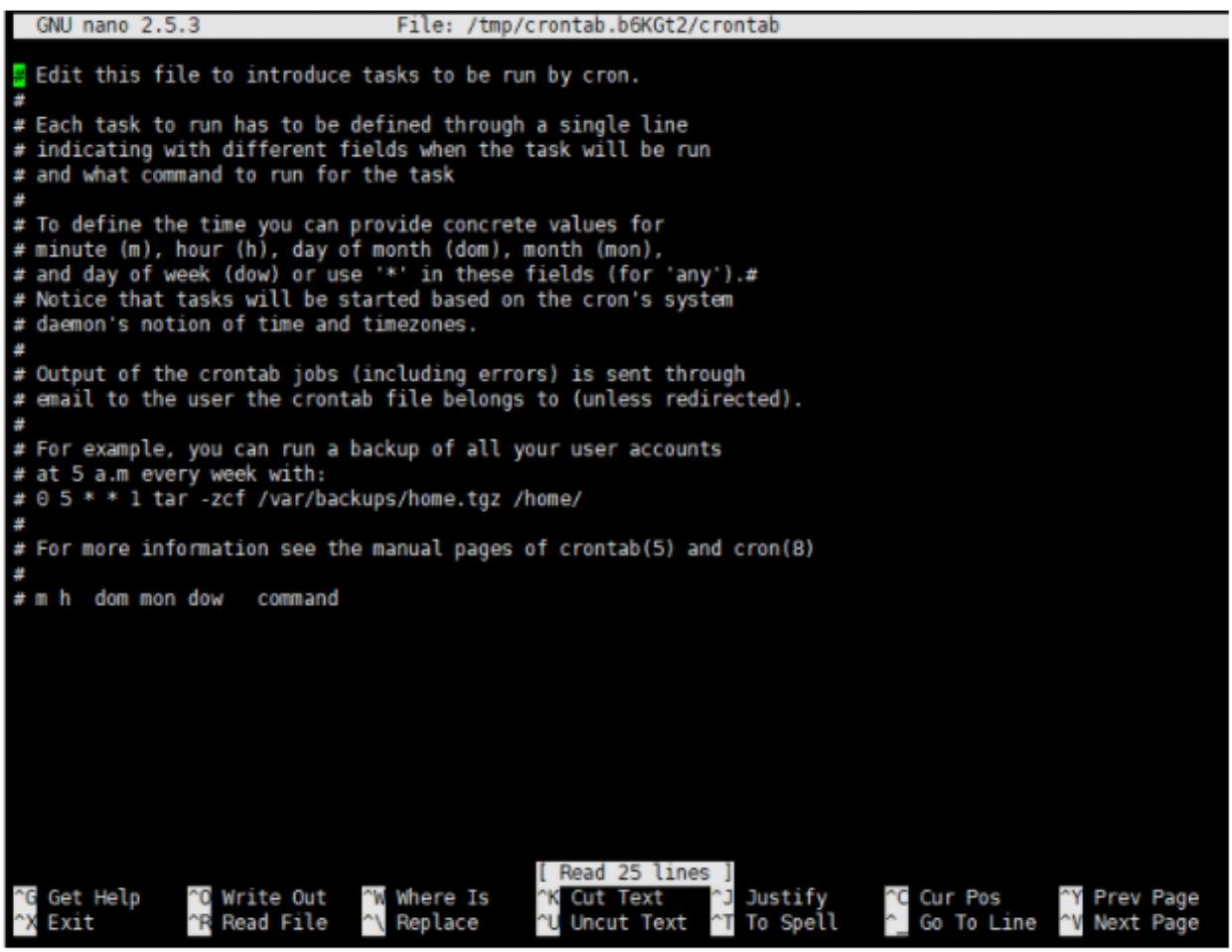
맨 밑에 등록된 한 줄이 바로 등록된 스케줄이다. 위의 스케줄은 자정을 기준으로 6시간마다 main.py를 python3로 실행시켜라는 뜻이다.

# 스케줄 등록하기

스케줄은 다음 명령어를 사용하여 등록할 수 있다. 해당 명령어를 실행하면, -l 옵션으로 보게 되었던 스케줄 목록을 텍스트 에디터(vi, vim, nano) 등으로 열 수 있게 되고, 해당 스케줄을 직접 작성해주면 된다.

```
$ crontab -e
```

아래의 그림은 필자가 nano 에디터로 해당 스케줄 목록을 연 경우이다.



스케줄의 등록 형식은 다음과 같다. 각 *에 해당하는 부분이 분,시,일,월,요일 등을 기재할 수 있으며 그대로 *을 적어놓으면 해당 시간열에 존재하는 모든 숫자를 포함한다는 뜻이 된다.~~하 뭔소리지~~

'시간'이 해당함.

```
* * * * *  수행할 명령어

T T T T T
| | | | |
| | | | |
| | | | └──────── 요일 (0 - 6) (0:일요일, 1:월요일, 2:화요일, …, 6:토요일)
| | | └────────── 월 (1 - 12)
| | └──────────── 일 (1 - 31)
| └────────────── 시 (0 - 23)
└──────────────── 분 (0 - 59)
```

필자는 설명을 매우 못하는 걸 알고 있기 때문에 위키에서 가져온 예시 목록도 함께 아래에 올렸으니 확인하도록 하자.

## 3. 예시 [편집]

```
* * * * * /root/every_1min.sh
```

→ 매 1분마다 /root/every_1min.sh 를 수행 (하루에 1440회[2])

```
15,45 * * * * /root/every_30min.sh
```

→ 매시 15분, 45분에 /root/every_30min.sh 를 수행 (하루에 48회[3])

```
*/10 * * * * /root/every_10min.sh
```

→ 10분마다 /root/every_10min.sh 를 수행 (하루에 144회[4])

```
0 2 * * * /root/backup.sh
```

→ 매일 02:00에/root/backup.sh 를 수행 (하루에 1회)

```
30 */6 * * * /root/every_6hours.sh
```

→ 매 6시간마다 수행(00:30, 06:30, 12:30, 18:30)

```
30 1-23/6 * * * /root/every_6hours.sh
```

→ 1시부터 매 6시간마다 수행(01:30, 07:30, 13:30, 19:30)

```
0 8 * * 1-5 /root/weekday.sh
```

→ 평일(월요일~금요일) 08:00

```
0 8 * * 0,6 /root/weekend.sh
```

→ 주말(일요일, 토요일) 08:00

## 스케줄 모두 삭제하기

모든 스케줄을 삭제하고 싶다면, 해당 명령어를 사용하면 된다. 단 crontab -e를 실행하여 해당 에디터로 스케줄을 직접 삭제해도 무방하기 때문에 편한 걸 사용하자.

```
$ crontab -r
```

## crontab(크론탭) 실행이 되지 않을 때

간혹 이래도 작동이 제대로 되지 않는 경우가 있다. 특히나 크론탭의 경우, 이게 잘 돌아가고 있는건지 만건지 커맨드 창에 나타나지 않기 때문에 꼭 스케줄 등록후, 확인을 해줄 필요가 있다.

제 시간에 실행이 되지 않을 경우, 아래와 같은 것들 따져보자.

### 1. 특정 파일을 가리켜야 할때, 절대경로로 등록하였는가?

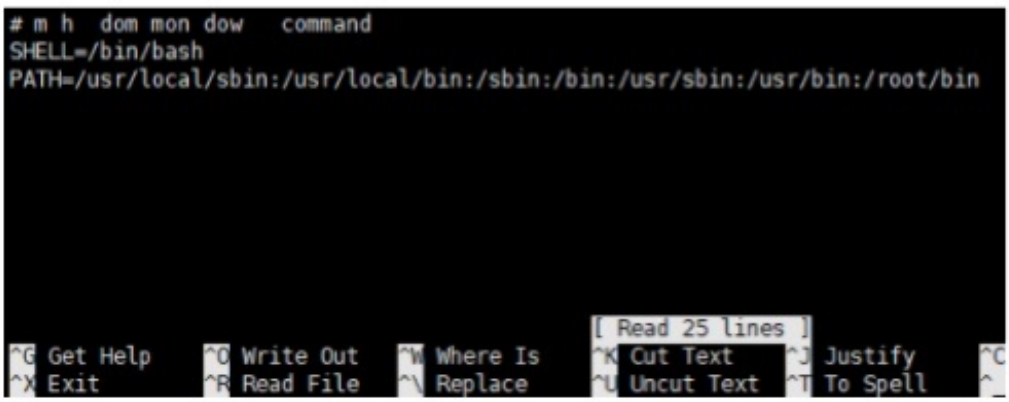
예를 들어 /home/ubuntu/main.py를 실행시킬려고 한다면, 아래의 두가지가 모두 가능하다.

```
$ pwd
/home/ubuntu/
// 둘다 가능
$ python3 main.py
$ python3 /home/ubuntu/main.py
```

2가지가 모두 가능한 이유는 쉘의 현재 경로가 /home/ubuntu/였기 때문이다. crontab의 경우, /(최상위 경로)에서 명령을 실행한다고 가정하고 반드시 절대 경로로 등록해주는 걸 권장한다. 즉, 크론탭에 등록시키기 전에 해당 명령어를 / 경로에서 실행시켜 보고 등록하자.

## 2. 환경 변수 문제

크론탭의 경우, 명령어를 실행할 때, 완전히 별도의 쉘을 띄워서 실행시키기 때문에 관련된 환경변수를 설정할 필요가 있다. 만약 해당 프로그램이 등록된 환경변수를 바탕으로 모종의 작업을 수행해야 할 경우, 아래처럼 crontab- e를 실행시킨 스케줄 목록 최상단에 환경 변수를 미리 등록하도록 하자.



## 3. 사용자 환경과 리눅스의 Timezone은 일치하는가?

AWS 같은 환경에서 crontab을 사용해야 할 때, 거의 이 문제를 겪을 것이다. 말그대로 crontab 자체는 정상적으로 동작하지만, 시간대가 달라 엉뚱한 시간을 기준으로 스케줄을 수행하는 것이다.

아래와 같이 localtime 및 timezone 정보를 서울로 수정해주도록 하자.

### 리눅스의 Timezone 변경

↑ AWS 리눅스.

```
$ date
Wed Jan  2 06:41:49 UTC 2019

$ sudo rm /etc/localtime
$ sudo ln -s /usr/share/zoneinfo/Asia/Seoul /etc/localtime

$ date
Wed Jan  2 17:25:15 KST 2019
```

### crontab의 Timezone 변경

```
$ sudo dpkg-reconfigure tzdata
// Asia 및 Seoul 선택
$ sudo service cron stop
$ sudo service cron start
```

## 4. 로그에서 실행이 안된 원인을 살펴보자

크론탭은 스케줄을 실행했지만 에러가 떠서 터졌을 경우, 로그에 기록된다. 해당 기록은 /var/log/cron에서 확인할 수 있으며 만약 Timezone이 달라서 혹은 스케줄 등록 문법이 틀려서 아예 crontab이 실행되지 않은 경우는 로그가 남지 않을 수도 있음을 유의하자.