

Schedule jobs with crontab on Mac OS X

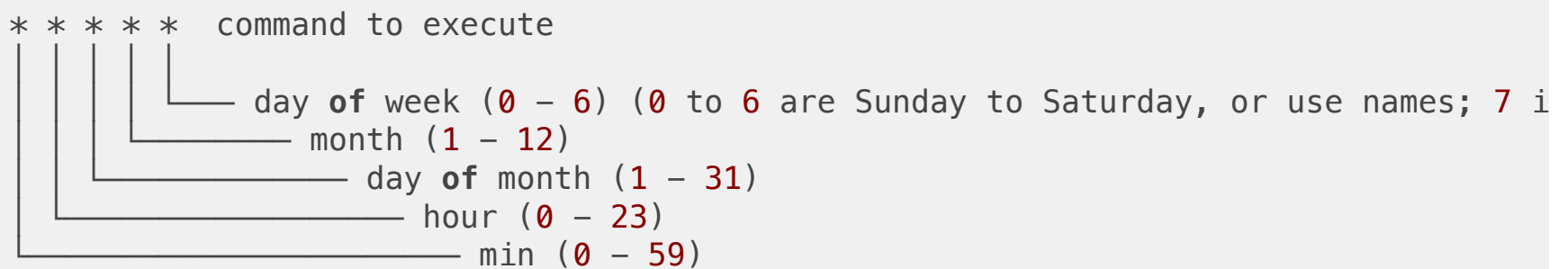
Article by Ole Michelsen posted on February 22, 2015

Running scripts on your computer is great. Running them automatically is even greater. If you are on a Mac (or Linux), you can use our good friend `crontab`, which is a scheduling tool that will run jobs (scripts) at regular intervals.

crontab

You add a job to crontab by editing the job list. A job is specified in the following format (first line):

```
* * * * * command to execute
```



The diagram illustrates the fields of a crontab entry. Five asterisks are aligned vertically on the left. Leader lines connect them to the following descriptions: the first asterisk points to 'day of week (0 - 6) (0 to 6 are Sunday to Saturday, or use names; 7 is Sunday)', the second to 'month (1 - 12)', the third to 'day of month (1 - 31)', the fourth to 'hour (0 - 23)', and the fifth to 'min (0 - 59)'. The text 'command to execute' is positioned to the right of the asterisks.

Example

Let's make an example job. I personally use cronjobs to make regular backups of my MySQL databases, so I will show how to set this up to run once a day.

My script is called `backup.sh`, and will dump my MySQL database to a zip file. We'll set it to run as a cronjob by editing the job list (with the `nano` editor):

```
env EDITOR=nano crontab -e
```

Now enter the following and press `CTRL + O` and `CTRL + X` to save and exit.

```
0 12 * * * cd ~/my/backup/folder && ./backup.sh
```

This will execute the command every day at 12:00, which changes to the folder of the script and runs it. The double ampersand `&&` allows you to specify multiple commands

that will run after each other.

Notice that if your computer is shut down or sleeping at the time, the script will not run until the next day at the specified time.

To see a list of your active crontab jobs, use the following command:

```
crontab -l
```

Environment

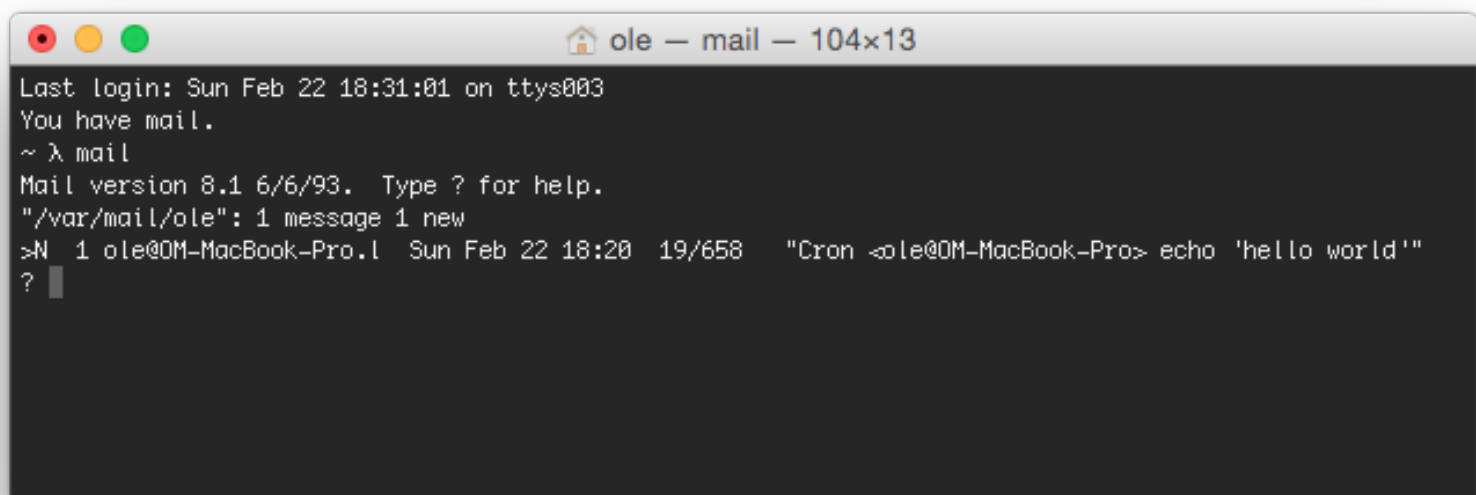
You might get a weird "command not found" error when running your crontab jobs for the first time, even though the script works fine when you run it yourself. That's because crontab executes commands without any of the normal environment variables set up.

To fix this, you can add the PATH in the beginning of your script, so it knows how to find programs like `mysqldump` :

```
#!/bin/sh
PATH=/usr/local/bin:/usr/local/sbin:~/bin:/usr/bin:/bin:/usr/sbin:/sbin
...
```

Disable mail alerts

So after you've set up your scripts and everything is honky dory, you might notice that the Terminal suddenly have started sending you e-mails.



If being pen pals with your Terminal isn't your thing, you can disable this behaviour by inserting this line at the very top of your crontab file:

```
MAILTO=""
```

Presto. Now your jobs will run silent.

Tips

Here's a few small snippets which might come in handy:

Description	Script
Execute on workdays 1AM	<pre>0 1 * * 1-5 /bin/execute/this/script.sh</pre>
Execute every 10 minutes	<pre>*/10 * * * * /bin/execute/this/script.sh</pre>
Log output to file	<pre>*/10 * * * * /bin/execute/this/script.sh >> /var/log/script_output.log 2>&1</pre>