

Raspberry Pi Spy

Raspberry Pi tutorials, scripts, help and downloads

Running A Python Script At Boot Using Cron

Posted on [July 27, 2013](#) by [Matt](#)



There may be times when you want to run a Python script when your Raspberry Pi boots up. There are a number of different techniques to do this but I prefer the method that uses “cron”.



Cron is a job scheduler that allows the system to perform tasks at defined times or intervals. It is a very powerful tool and useful in lots of situations. You can use it to run commands or in this case, a Python script.

Step 1 – Create A Python Script

The first step is creating your Python script. This will be the script that will run at boot time. It is important to remember its name and location. In this example I will assume the script is called “MyScript.py” and it is located in “/home/pi/”.

Double check you’ve got the correct path by typing :

```
cat /home/pi/MyScript.py
```

This should show the contents of your script.

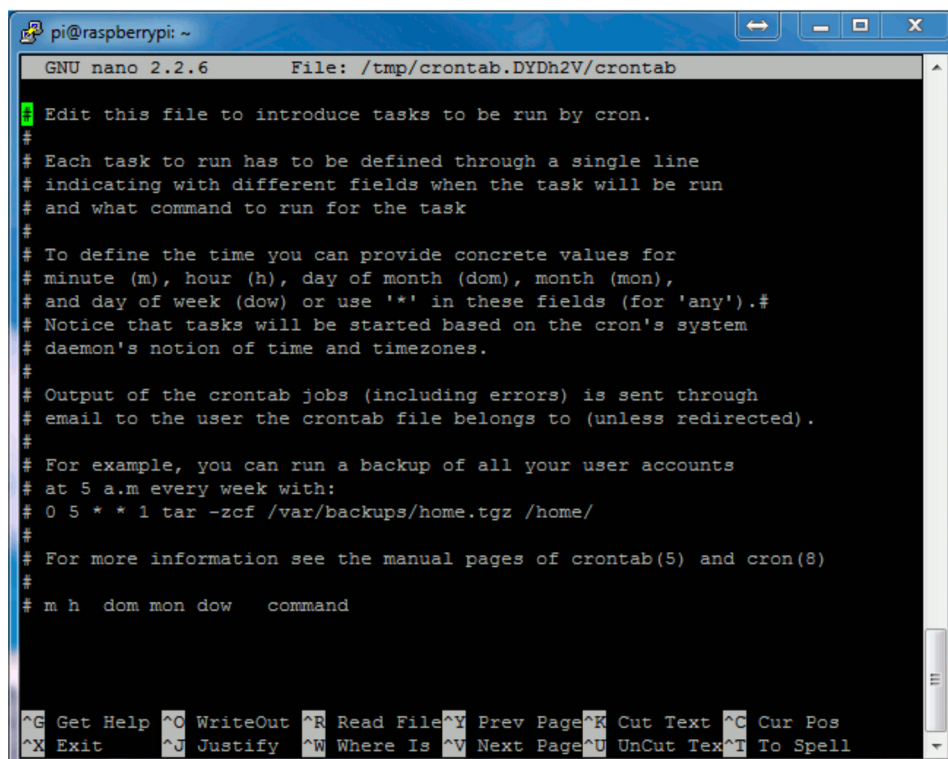
Make sure your script works and does what you expect it to. Once you are running at boot it isn’t so easy to debug so don’t rush!

Step 2 – Add A New Cron Job

To create a new job to Cron we will modify the “crontab”. This is a table that contains the list of jobs that Cron will monitor and run according to it’s details. To edit it we use the command :

```
sudo crontab -e
```

Each user of the system (ie “pi”) can have its own Crontab but in this case we want to add it as an admin so we prefix our “crontab -e” command with “sudo”. You should see something that looks like this :



```
pi@raspberrypi: ~
GNU nano 2.2.6 File: /tmp/crontab.DYDh2V/crontab

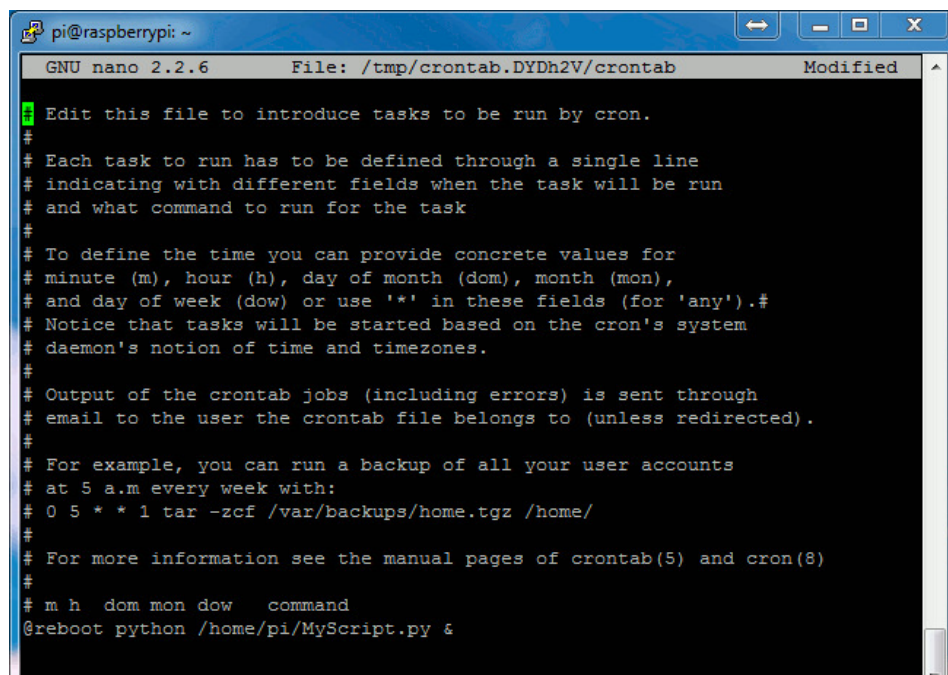
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
^G Get Help ^C WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Tex ^T To Spell
```

Using your cursor keys scroll to the bottom and add the following line :

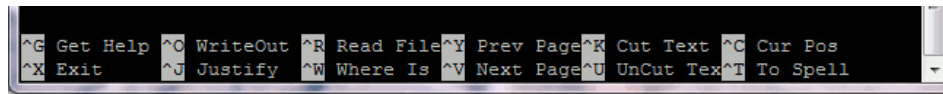
```
@reboot python /home/pi/MyScript.py &
```

This tells Cron that every boot (or reboot or start-up) we want to run Python with the script MyScript.py. The “&” at the end of the line means the command is run in the background and it won’t stop the system booting up as before.

Your screen should look something like this :



```
pi@raspberrypi: ~
GNU nano 2.2.6 File: /tmp/crontab.DYDh2V/crontab Modified
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
@reboot python /home/pi/MyScript.py &
```



To save these changes click “CTRL-X”, then “Y” and finally “Return”. You should now be back at the command prompt.

To start testing you can now reboot using :

```
sudo reboot
```

Once setup your Python script will run whenever you reboot or start-up your Pi. There may be times when you reboot and don’t want the script running. To stop it you can find out its process number and “kill” it. To do this type :

```
ps aux | grep /home/pi/MyScript.py
```

This should give you a line starting with “root” and ending in the path to your script. Immediately after the “root” should be a process number. For example :

```
root 1863 0.0 1.0 24908 4012 ? S1 19:45 0:00 python /home/pi/MyScript.py
```

In this case we can stop the process using :

```
sudo kill 1863
```

Final Thoughts

If you are feeling adventurous you can write your Python script to automatically exit if a certain condition is met so you don’t need to ever “kill” it. Ideas include :

- Test the GPIO pins and quit if a switch is being pressed. Maybe two switches being held down.
- Test if a network connection is available and quit if it is. This may indicate you are testing (a camera for example) and you only want the script to auto-run when there is no network present.
- Check for the existence of a particular file. This would allow you to create a named file to prevent the script from running at next boot.

There are other techniques to run scripts at boot up and you might want to Google “rc.local” or “init.d”. I prefer the Cron method because it is so simple.

For additional information on the powerful features of Cron take a look at the [Wikipedia Cron page](#).

This entry was posted in [Python](#), [Raspbian](#) and tagged [cron](#), [python](#). Bookmark the [permalink](#).

 64 12 14

3

22 Responses to *Running A Python Script At Boot Using Cron*

**manuti** says:

July 29, 2013 at 9:25 am

Perfect for running my IP detection script.

https://github.com/manuti/Check_IP_and_send_email

<http://raspberryparatorpes.net>

[Reply](#)

**J Ashfield** says:

July 29, 2013 at 11:54 am

Thanks for this post, I spent most of this morning looking a easy way to do this. It's sooo easy your way, I tried testing other methods but they were too hard for a newbie like me. I thoroughly enjoy reading your posts. My favorite project is the 20x4 LCD ,I bought one and used it to display the temperature from the digital thermometer (the one from your post) and got it to run the program at startup 😊

[Reply](#)

**JimT** says:

August 1, 2013 at 11:43 pm

I have a python program that plays wav files depending on GPIO pin readings. It works fine when I run it from the command line, but it doesn't appear to work when I start it using Cron. I see it running when I issue the 'ps aux' command, but there is no audio when I start it using Cron. Another odd thing is that 'sudo kill' does not kill that process.

I have to run it using 'sudo' from the command line, but 'ps aux' shows that it runs as root when I put it in Cron, so I don't think that's the problem.

Any ideas on how I can debug this? Thank you!

[Reply](#)

**cavok** says:

August 2, 2013 at 10:02 pm

i also had a similar problem,
exact same issue as described above. depending on which method the script was autostarted it would work or not...
python script playing wav file depending on interrupt on the gpio state.
here's what worked for me:

1. make a file "/bin/autologin.sh"

2. put the following two lines in it:

```
#!/bin/bash
```

```
/bin/login -f root
```

3. set permissions

```
chmod a+x /bin/autologin.sh
```

4. edit "/etc/inittab"

find a line that looks very similar and edit it to:

```
1:2345:respawn:/sbin/getty -n -l /bin/autologin.sh 38400 tty1
```

5. edit “/root/.bashrc”

add the following 3 lines:

```
if [[ $(tty) == '/dev/tty1' ]]; then
```

```
python /home/pi/yourscript.py
```

```
fi
```

and it should reliably start at every boot.

[Reply](#)



JimT says:

August 5, 2013 at 3:57 pm

Thank you! I will try that.

[Reply](#)



JimT says:

August 15, 2013 at 4:58 pm

The program starts at boot, and recognizes the GPIO signals, but when it plays audio files, no sound comes out of the speakers. Any ideas?

[Reply](#)



Catalin says:

September 24, 2013 at 2:47 pm

Make sure that the output is set to your needs:

```
sudo amixer cset numid=3 1
```

```
amixer cset numid=3 n where:
```

Where is the required interface : 0=auto, 1=analog, 2=hdmi.

[Reply](#)



JimT says:

August 14, 2013 at 12:08 am

cavok, That didn't work for me. Where does the output from Python print commands go? It might help to see the output.

[Reply](#)



Shane says:

September 16, 2013 at 8:02 pm

Having a similar problem

Have my RPi receiving inputs from an MPR121 and either triggering an arduino via serial to make patterns with an LED strip, or play sounds using py game mixer.

If I run from the command line (I have just put the file in /home/pi/beetbox_serial.py) run as sudo then all is well.

If I schedule it (cron or /etc/rc.local to try it at startup) then I get serial output (and so lights) but no sounds- just a pop (also no printout from the Python script which I do get (tells me which inputs are triggered) when this works

Is this an Env setting problem? Is this something due to the GPIO? The GPIO's are reading it's the sound output that is the problem (seems common to this thread)

Any help would be appreciated

Shane

[Reply](#)



carlos says:

October 23, 2013 at 3:35 pm

Hi, I did as you explained, my script is just

```
print "hello"
```

```
print "hello"
```

```
print "hello"
```

```
print "hello"
```

I removed the & at the end of the crontab statements, but nothing happens. Do you have any idea why this is happening?

[Reply](#)



Matt says:

October 23, 2013 at 8:22 pm

It is possible it is running but you won't see the output because it happens before you've logged in. If your script was flashing an LED in a loop it would be easier to see. I use this technique to launch scripts that read sensors and take photos so the script keeps running in a loop in the background.

[Reply](#)



zarafiq says:

November 6, 2013 at 10:54 pm

Neat! Thanks!

[Reply](#)



Mohit says:

November 19, 2013 at 6:07 am

Would this also execute scripts that require sudo access ?

For e.g starting up a Flask server on start up ...

[Reply](#)



Matt says:

November 19, 2013 at 7:34 pm

Should work fine. I use it to launch Python scripts that use GPIO and they require sudo.

[Reply](#)



Rrusporoxus says:



November 25, 2013 at 9:19 am

Very nice Tutorial
worked fine.

I needed
`os.chdir("myDirectory")`
in my python script file...

[Reply](#)



Shawn says:

November 25, 2013 at 7:28 pm

Worked perfectly! I've used this a couple of times now. Thank you!

[Reply](#)



Maham says:

December 3, 2013 at 1:51 pm

Don't know why, it is not working for me 😞 . I have followde the above instructions carefully.

1. I made a python script "Hello.py" in /home/pi
2. I checked,if it running or not,it is giving me output.
3. I used crontab -e command and added the line @reboot python /home/pi/Hello.py &
4. Reboot RPi using sudo reboot , but nothing happened!

Please help me out.

[Reply](#)



John Streff says:

December 11, 2013 at 5:06 pm

Thanks for this post.

It works.

Thanks John

[Reply](#)



bill says:

December 18, 2013 at 7:55 pm

I can not edit the file. I actually can't edit any of the files people suggest I edit to make a script run at start.

There is always a permission problem. How can I get around it? I use a fresh installation every time I try.


There must be some way to bypass this whole "permissions" thing altogether as it truly serves no purpose.

I need to run a script at startup.

[Reply](#)



Matt says:



December 20, 2013 at 7:24 pm

What image are you using and where are you getting it from?

[Reply](#)



Alex says:

December 18, 2013 at 8:25 pm

I did this with a python script of mine and now it boots to a black screen after the initial pi boot screen and then there is nothing I can do. Is there anyway to tell it to stop doing this without being able to startup properly? Anything through the cmdline.txt on the sd card? Or anyway to tell it not to run cron before it finishes booting and goes black?

[Reply](#)



John says:

January 21, 2014 at 4:51 pm

Hi! I was using this example with one small Python code I was writing the other day and it works perfectly!

By the way, what are the pros/cons of using this method instead of running your python code as a daemon on start up? Except from the complexity point of view!

[Reply](#)

This site is not associated with the official Raspberrypi.org site or the Raspberry Pi Foundation. Raspberry Pi is a trademark of the Raspberry Pi Foundation.

Copyright © 2013 - All Rights Reserved - Matt Hawkins