

Ask Ubuntu is a question and answer site for Ubuntu users and developers. It only takes a minute to sign up.

[Join this community](#)

Anybody can ask a question

Anybody can answer

The best answers are voted up and rise to the top

Repeat a command every x interval of time in terminal?

Asked 5 years, 7 months ago Active 7 months ago Viewed 276k times



How can I **repeat** a command every interval of time , so that it will allow me to run commands for **checking** or **monitoring** directories ?

146

There is no need for a script, i need just a simple command to be executed in terminal.



command-line

bash

scripts

monitoring



73

edited Dec 27 '16 at 7:59



muru

1

asked Mar 6 '14 at 15:52

user239745

8 Answers

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

223

Open Terminal and type:

```
watch -n x <your command>
```

change **x** to be the time in seconds you want.

For more help using the `watch` command and its options, run `man watch` or visit this [Link](#).

For example : the following will list, every **60s**, on the same Terminal, the contents of the Desktop directory so that you can know if any changes took place:

```
watch -n 60 ls -l ~/Desktop
```

edited Jul 26 '16 at 16:31

Brian Moths



306 2 8

answered Mar 6 '14 at 15:54

nux



24.9k 30 99 119

34 +1 but be careful when using expansions. For example, try the difference between `watch -n 1 'echo $COLUMNS'` and `watch -n 1 echo $COLUMNS` when resizing your terminal - the former is expanded every second, but the latter is expanded only once *before* `watch` starts. – [IOb0](#) Mar 6 '14 at 16:06

It Works like a charm ! Ty nux – [user239745](#) Mar 6 '14 at 16:28

Problem I have with this solution is that you can't run `watch` as a process and just leave it running in the background (for example with `&disown`) – [Cestarian](#) May 13 '15 at 2:06

2 Is there any way to use `watch` with "history enabled" type command? I love using `watch`, but sometimes I'd prefer to see a log of previous executions as well, instead of just the last one. And yes, I know I can use scripting (`while true`) to accomplish this, but using the `watch` utility is so much cleaner! – [rinogo](#) Sep 14 '15 at 18:44

this worked for simpler commands but with pipelined commands chaining this didn't work for me.. following was the command I tried =>`cat api.log | grep 'calling' | wc -l` – [Sudip Bhandari](#) Sep 29 '16 at 7:43

You can also use this command in terminal, apart from nux's answer :

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

Example

```
while true; do ls; sleep 2; done
```

This command will print output of `ls` at an interval of 2 sec.

Use `Ctrl + C` to stop the process.

There is few drawbacks of `watch`

- It can not use any aliased commands.
- If the output of any command is quite long, scrolling does not work properly.
- There is some trouble to set maximum time interval beyond certain value.
- `watch` will interpret ANSI color sequences passing escape characters using `-c` or `--color` option. For example output of `pygmentize` will work but it will fail for `ls --color=auto`.

In the above circumstances this may appear as a better option.

edited Apr 13 '17 at 12:37

 Community ♦
1

answered Mar 6 '14 at 16:00

 **souravc**
30.5k 15 83 114

`watch` exists for that, this is a bit useless I would say – Bruno Pereira Mar 6 '14 at 16:36

14 I am not claiming this answer is to be used at first place. `watch` is good in most cases. That is why I mentioned "apart from nux's answer" at the beginning. But there are few problems with `watch` for example One **can not use any aliased commands with** `watch`. Take for example `ll` which is aliased to `ls -laF` but can not be used with `watch`. Also in case if the output of any command is quite long you will be in trouble in scrolling using `watch`. In these few special cases this answer may appear a better option. – souravc Mar 6 '14 at 17:25

@souravc My version of `watch` at least allows the `-c` or `--color` options for colorized output. – Lily Chung Mar 7 '14 at 1:57

8 `while sleep x` is better - it's easier to kill. – d33tah Mar 8 '14 at 15:43

3 This is a nice alternative and, unlike `watch`, it keeps the command history. – adelriosantiago Feb 14 '17 at 18:31

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.



example, `watch` is a command to "snoop on another tty line".

2. `while true; do command; sleep SECONDS; done` also has a caveat - your command might be harder to kill using **CTR+C**. You might want to prefer `while sleep SECONDS; do command; done` - it's not only shorter, but also easier to interrupt. The caveat is that it will first sleep, then run your command, so you'll need to wait some `SECONDS` before the first occurrence of the command will happen.

edited Apr 6 '15 at 19:09

answered Mar 8 '14 at 15:51



d33tah

450 3 10

- 2 Hopefully it's acceptable as an answer instead of a comment - I wanted to show another solution here and commenting would actually give me less attention. – d33tah Mar 8 '14 at 15:52

Why exactly does it matter where you put `sleep` in the `while` loop? I couldn't find any difference, `Ctrl+C` broke the loop *instantly* no matter what. – dessert Nov 17 '17 at 8:03

@dessert: depends on what you're trying to break out from I guess. Normally, `ctrl+c` would just kill your `command` and `sleep` and only break if you kill `true`. – d33tah Dec 11 '17 at 17:08



11



Sounds like the ideal task for the `cron` daemon which allows for running periodic commands. Run the `crontab -e` command to start editing your user's cron configuration. Its format is documented in [crontab\(5\)](#). Basically you have five time-related, space-separated fields followed by a command:

The time and date fields are:

field	allowed values
-----	-----
minute	0-59
hour	0-23
day of month	1-31
month	1-12 (or names, see below)
day of week	0-7 (0 or 7 is Sunday , or use names)

For example, if you would like to run a Python script on every Tuesday, 11 AM:

```
0 11 * * 1 python ~/yourscript.py
```

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

```
# Creates a temporary directory for ~/.distcc at boot
@reboot ln -sfm "$(mktemp -d "/tmp/distcc.XXXXXXXX")" "$HOME/.distcc"
```

answered Mar 7 '14 at 18:54



Lekensteyn

132k 51 275 368

-
- 4 The question asks how to run something periodically in the terminal. `cron` runs behind the scenes rather than in the terminal – [northern-bradley](#) Dec 15 '14 at 19:22
-



If you are monitoring the file system, then `inotifywait` is brilliant and certainly adds less load on your system.

5

Example :



In **1st** terminal type this command :

```
$ inotifywait .
```

Then in **2nd** terminal, any command that affects the current directory,

```
$ touch newfile
```

Then in original terminal `inotifywait` will wake up and report the event

```
./ CREATE newfile2
```

Or in a loop

```
$ while true ; do inotifywait . ; done
Setting up watches.
Watches established.
./ OPEN newfile2
```

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

```

Watches established.
./ DELETE newfile
Setting up watches.
Watches established.
./ CREATE,ISDIR newdir
Setting up watches.
Watches established.

```

edited May 18 '14 at 2:13

**nux****24.9k**

30

99

119

answered Mar 6 '14 at 16:35

**X Tian****215**

1

6

the user told you , no script , and maybe he dont want to monitor anything – **nux** Mar 6 '14 at 16:36

- 3 I didn't tell him to write a script, I suggested that if they are looping inorder to watch for particular filesystem event, then inotifywait is useful, and uses less resources than repeating a command. I often run several commands on a command line eg `grep something InLogFile|less` is that a script ? – **X Tian** Mar 6 '14 at 16:43

its a good answer , try to edit it to look more simple . – **nux** Mar 6 '14 at 16:45

- 3 What could be simpler than `.` . I can't leave out the command. – **X Tian** Mar 6 '14 at 16:48

Thanks @XTian, a great command. I also now saw in the man page that you can add `-m` to continually monitor without a loop. – **yoniLavi** Jul 21 '14 at 16:06

You can create your own `repeat` command doing the following steps; [credits here](#):

4

First, open your `.bash_aliases` file:

```
$ xdg-open ~/.bash-aliases
```

Second, paste these lines at the bottom of the file and save:

```
repeat() {
n=$1
shift
while [ $# -gt 0 ] ; do

```

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

```
}
```

Third, either close and open again your terminal, or type:

```
$ source ~/.bash_aliases
```

Et voilà ! You can now use it like this:

```
$ repeat 5 echo Hello World !!!
```

or


```
$ repeat 5 ./myscript.sh
```

answered Aug 8 '14 at 19:38



Bluffer

41 1

there is a small typo in the line `xdg-open ~/.bash-aliases`. it should be: `xdg-open ~/.bash_aliases` (ie: underscore) – **ssinfod** Feb 4 '18 at 1:21 



you can use `crontab`. run the command `crontab -e` and open it with your preferred text editor, then add this line

4

```
*/10 * * * * /path-to-your-command
```



This will run your command every 10 minutes

```
* */4 * * * /path-to-your-command
```

This will run your command every 4 hours

Another possible solution

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).

X number of times to repeat.

Y time to wait to repeat.

Example :

```
$ echo; for i in $(seq 5); do $cmd "This is echo number: $i"; sleep 1; done
```

edited Jun 3 '15 at 10:19

answered Mar 7 '14 at 5:57



Maythux

55.7k

35

188

234

Why is this an improvement? You just added an extra, needless, step by saving the command as a variable. The only things this does is i) make it longer to type ii) forces you to use only simple commands, no pipes or redirects etc. – [terdon](#) ♦ Jul 3 '14 at 11:52

Remove the extra variable – [Maythux](#) May 14 '15 at 11:24



3

Another concern with the "watch" approach proposed above is that it does display the result only when the process is done. "date;sleep 58;date" will display the 2 dates only after 59 seconds... If you start something running for 4 minutes, that display slowly multiple pages of content, you will not really see it.



On the other hand, the concern with the "while" approach is that it doesn't take the task duration into consideration.

```
while true; do script_that_take_between_10s_to_50s.sh; sleep 50; done
```

With this, the script will run sometime every minutes, sometime might take 1m40. So even if a cron will be able to run it every minutes, here, it will not.

So to see the output on the shell as it's generated and wait for the exact request time, you need to look at the time before, and after, and loop with the while.

Something like:

By using our site, you acknowledge that you have read and understand our [Cookie Policy](#), [Privacy Policy](#), and our [Terms of Service](#).


```

sleep `echo $(( ( RANDOM % 30 ) + 1 ))`
echo Before waiting `date`
after=`date +%s`
DELAY=`echo "60-($after-$before)" | bc`
sleep $DELAY
echo Done waiting `date`
done

```

This will output this:

As you can see, the command runs every minutes:

```

Date starting Mon Dec 14 15:49:34 EST 2015
Before waiting Mon Dec 14 15:49:52 EST 2015
Done waiting Mon Dec 14 15:50:34 EST 2015

```

```

Date starting Mon Dec 14 15:50:34 EST 2015
Before waiting Mon Dec 14 15:50:39 EST 2015
Done waiting Mon Dec 14 15:51:34 EST 2015

```

So just replace the "sleep `echo \$(((RANDOM % 30) + 1))`" command with what ever you want and that will be run, on the terminal/shell, exactly every minute. If you want another schedule, just change the "60" seconds with what ever you need.

Shorter version without the debug lines:

```

while ( true ); do
  before=`date +%s`
  sleep `echo $(( ( RANDOM % 30 ) + 1 ))` # Place you command here
  after=`date +%s`
  DELAY=`echo "60-($after-$before)" | bc`
  sleep $DELAY
done

```

answered Dec 14 '15 at 20:56



jmspaggi

31 2

Replying to myself... If your command takes more than the delay you configure, \$DELAY will get a negative value and the sleep command will fail so the script will restart right away. Need to be aware of that. – **jmspaggi** Dec 14 '15 at 21:06

By using our site, you acknowledge that you have read and understand our Cookie Policy, Privacy Policy, and our Terms of Service.