



Search our articles



DreamHost Knowledge Base > Advanced Topics / Customization > Cron Jobs / Crontab



Creating a custom Cron Job

Overview

This article shows how to manually create a custom cron job using your [Shell user](#). It involves logging into the server via [SSH](#) to run several commands.

These instructions can also be used to edit an existing cron job you created in the panel, however for simplicity, it's recommended that if you created it in the panel, you continue to edit it from the panel. This is important since adding a cron job in the panel will overwrite your custom crontab file as mentioned below.



In the following examples, `username` would be your [Shell user](#) and **example.com** your website.

Basic details

The crontab files are where the lists of jobs and other instructions to the cron daemon are kept. Each user at DreamHost has their own individual crontab file that can be accessed by running the following command under your [Shell user](#):

```
[server]$ crontab -e
```

Crontab files are simple text files that have a particular format. Each line of a crontab file follows a particular format as a series of fields, separated by spaces and/or tabs. Each field can have a single value or a series of values. A single cron job should take up exactly one line, but this can be a long line (more than 80 characters).

Things to look out for when editing/creating your crontab

Each line has five time/date fields, followed by a command, followed by a newline character ("\n"). A common problem is not including a newline, so hit 'Enter/Return' a time or three at the end of your command.

Another common problem is automatic word-wrap breaking up a long line into multiple lines, so make sure your text editor doesn't do this.

Blank lines and leading spaces and tabs are ignored. Lines whose first non-space character is a hash-sign (#) are ignored as they are considered comments. Note that comments are not allowed on the same line as cron commands, since they are interpreted as being part of the command. Similarly, comments are not allowed on the same line as environment variable settings (like [MAILTO](#)).

What if I already created a cron job in the panel under my Shell user?

There are two ways to create custom cron jobs.

- Editing the existing crontab on the server
- Using a custom crontab file

If you've edited the existing crontab

If you have already created a cron job in your panel, you can view it by running `crontab -e` under your Shell user. If you edit the file to add another cron job below the existing panel one, the panel cron job will continue to function normally in addition to your new edited code.

Any adjustments in the panel will not affect your custom code.

If you switched the server's crontab with your custom crontab

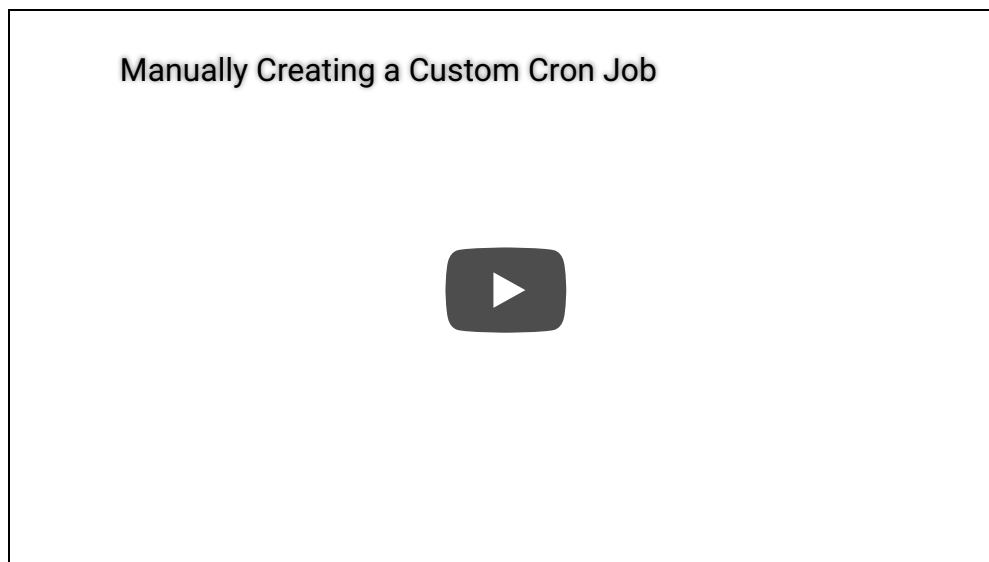
You can also use a custom crontab that you created. If you do this, the server's crontab is overwritten. You can replace the server's crontab by running the following:

```
[server]$ crontab /home/username/mycrontab
```

However, if you then add or edit a cron job in the panel, your custom crontab will be overwritten. So you need to either use the server's crontab, or your custom crontab.

Manually creating a custom cron job

The following section is also available as an instructional video:



The instructions below explain how to add a custom cron job under your Shell user. These instructions assume you have NOT added a cron job in the panel yet, so the crontab file is blank.

1. Log into your server via [SSH](#) using the Shell user you wish to create the cron job under.
2. Once logged in, run the following command to open your crontab file.

```
[server]$ crontab -e
It looks like you don't have a MAILTO line in your crontab file
For performance reasons we ask that you specify an address where
cronjob output will be delivered.  If you do not wish to receive
cronjob output, simply press enter and cronjob output will not be
mailed to you.

For more information regarding this, please visit:
https://help.dreamhost.com/hc/en-us/articles/215088608

Where would you like cronjob output delivered? (leave blank to disable)
: admin@example.com

cronjob output will be emailed to admin@example.com
confirm? (y/N): y
```

After you add your email (or leave it blank), you must choose which text editor you'd like to use.

```
Select an editor.  To change later, run 'select-editor'.
 1. /usr/bin/joe
 2. /usr/bin/jstar
```

```
3. /usr/bin/jpico
4. /usr/bin/jmacs
5. /usr/bin/jed
6. /bin/nano          <---- easiest
7. /usr/bin/vim.basic
8. /usr/bin/rjoe
9. /usr/bin/mcedit
10. /usr/bin/vim.tiny
11. /bin/elvis-tiny
12. /usr/bin/emacs25
13. /bin/ed
```

Choose 1-13 [3]: 6

3. You are then asked to choose an editor to view this file. #6 uses the program `nano` which is the easiest option. View the [Creating and editing a file via SSH](#) article for instructions on how to use `nano`.



The above options display if you're logged into a Shared server. If you are logged into a VPS (private server) running Debian, you will see the following list instead.

Select an editor. To change later, run 'select-editor'.

```
1. /bin/ed
2. /bin/nano          <---- easiest
3. /usr/bin/emacs24
4. /usr/bin/vim.basic
5. /usr/bin/vim.tiny
```

4. A blank crontab file opens. Add the code for your cron job. This example runs a file named **mail.php** under the username of 'username'. This should be the same SSH username you're currently logged in under. This example runs the cron job at 8:13 pm.

```
# Custom cron job
MAILTO="user@example.com"
13 20 * * * php /home/username/mail.php
```



There are two parts to the command. The first part must specify the path to the program you'd like to use to run the cron job. For example let's say you have a PHP file named **script.php** in your domains directory:

- **/home/username/example.com/script.php**

To run this command you'd enter the path to your chosen version of PHP followed by a space, followed by the path to the file:

- **/usr/local/php74/bin/php /home/username/example.com/script.php**

You could also use the default version by using 'php' instead of the full path.

5. Save the file. You should see the following response:

```
crontab: installing new crontab
```

That's it. The cron job should now run every day at 8:13pm.

Crontab commands



Please note that if you choose to replace the server's crontab, all cron jobs created in the panel for this specific username will no longer function since they would have been overwritten on the server.

Additionally, if you update any cron jobs under this user in the panel, it will overwrite your custom crontab. The crontab will be replaced in its original form as created in the panel.

Replace your existing crontab with your custom crontab file

```
[server]$ crontab /home/username/filename
```

Edit your server's crontab

```
[server]$ crontab -e
```

View your crontab

```
[server]$ crontab -l
```

Remove your crontab

```
[server]$ crontab -r
```

Explanation of the Date/Time fields

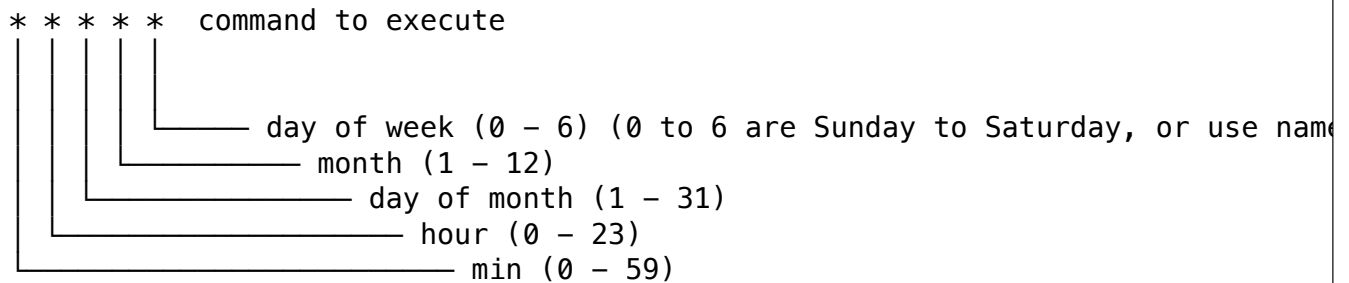
The first five fields of the line are the date and time field which specify how frequently and when to execute a command. When adding the cron job in the DreamHost panel, the Date/Time is added for you automatically based on your 'When to run' setting.

Field no.	Description	Permitted values
1	minute	0-59

2	hour	0-23
3	day of the month	1-31
4	month	1-12
5	day of the week	0-7


 For day of the week, both 0 and 7 are considered Sunday. The time is based on that of the server running cron.

Another (graphical) way of looking at these fields.



There are several ways of specifying multiple values in these fields:

- The comma (',') operator specifies a list of values.
 - 1,3,4,7,8
- The dash ('-') operator specifies a range of values.
 - 1-6
 - This is equivalent to "1,2,3,4,5,6".
- The asterisk ('*') operator (frequently known as a wildcard) specifies all possible values for a field. For example, an asterisk in the hour (second) field would be equivalent to 'every hour'.

 If you set the cron job to run on a specific day of the week, the **month** and **day of month** fields will still appear as an asterisk (*). The day of the week value will override these values, so it will only run once per week on your selected day.

- The slash ('/') operator can be used in conjunction with an asterisk to skip a given number of values.
- Example:
- /3
 - This means to skip to every third value. So "*" / 3 in the hour field is equivalent to "0,3,6,9,12,15,18,21"; "*" specifies 'every hour' but the "/3" means that only the first, fourth, seventh, etc. values given by "*" are used.

You can also use one of these special strings in place of the time/date fields.

Entry	Description	Equivalent to	Example
@yearly (or @annually)	Run once a year at midnight on January 1	0 0 1 1 *	@yearly php /home/username/mail.php
@monthly	Run once a month at midnight on the first day of the month	0 0 1 * *	@monthly php /home/username/mail.php
@weekly	Run once a week at midnight on Sunday morning	0 0 * * 0	@weekly php /home/username/mail.php
@daily (or @midnight)	Run once a day at midnight	0 0 * * *	@daily php /home/username/mail.php
@hourly	Run once an hour at the beginning of the hour	0 * * * *	@hourly php /home/username/mail.php
@reboot	Run at startup (of the cron daemon)	@reboot	@reboot php /home/username/mail.php

Review the following Wikipedia article for further information:

- [Cron – Nonstandard predefined scheduling definitions](#)

Output

The output of the cron job is determined by what is sent to the terminal as a result of the commands/script that are executed. By default, all output is emailed to the location specified in the [MAILTO](#) variable (see the [MAILTO variable requirement](#) article for more information). As noted above, if your cron job command doesn't create any output on the command line then no email is sent.

You can provide special instructions for the standard out (STDOUT) and standard error (STDERR) output by using the `>` operator. When you use `>` without a number before it, it defaults to `1>`. This is the standard (non-error) output.

When you use `2>` you are specifying what to do with the error output. So, for example, `>my_file.txt` would redirect standard output to a file called **my_file.txt**, and `2>my_errors.txt` would redirect the errors to a file called **my_errors.txt**.

Permissions

By default, files created on DreamHost's servers have a permissions level of 644. If you choose to execute a script via cron, you may need to set the permissions for the file to 744 using `chmod` in order to allow it to execute properly.

Examples of custom cron scripts

The following examples show what you could add to a new file in order to create a cron job.

Example 1: This runs a command at 4:10 PM PDT/PST, and emails you the regular and error output to the destination specified by MAILTO.

```
10 16 * * * perl /home/username/bin/yourscript.pl
```

Example 2: This runs a command at 2:00 AM PDT/PST on Saturday, and the only output is errors.

```
0 2 * * 6 sh /home/username/weekly/weekly-pruning.sh > /dev/null
```

Example 3: This runs at midnight on New Years Day (January 1st), and there is no output.

```
0 0 1 1 0 python /home/username/happy.new.years.py >/dev/null 2>&1
```

2>&1 is a special redirect that sends the standard error (“2>”) output to the same place as the standard out (“>” or “1>”) output.

Example 4: This runs a PHP script called cron.php at the top of every hour.

```
0 * * * * php /home/username/cron.php
```

Example 5: This runs a local script (i.e. hosted at DreamHost) every 15 minutes.

```
*/15 * * * * /usr/local/php74/bin/php /home/username/myscript.php
```

Example 6: This runs an external script (i.e. hosted elsewhere) every 30 minutes using `curl`.

```
*/30 * * * * /usr/bin/curl -s https://example.com/send.php &> /dev/null
```

&>/dev/null is an abbreviation for 1> /dev/null 2> &1. It redirects both file descriptor 2 (STDERR) and descriptor 1 (STDOUT) to /dev/null.

View unix.stackexchange.com/a/70971 for more information.

Example 7: This runs a local script (i.e. hosted at DreamHost) every 10 minutes.

```
*/10 * * * * /usr/local/php74/bin/php /home/username/myscript.php
```


Example 8: This uses `wget` to download a file to a directory named `/cronfolder`.

```
*/10 * * * */usr/bin/wget -P /home/username/cronfolder/ https://example.com/in
```

Dedicated Server editing

If you are logged in as a [Dedicated Server admin user](#), you can edit the crontab file directly. It is stored here:

```
/var/spool/cron/crontabs/username
```

You'll need to use `sudo` on your Dedicated Server (or start an interactive session as the root user with `sudo -i`) to access that file.



If you require sudo/admin access, you must upgrade to a Dedicated Server.

Example (opening the file with the 'vi' text editor):

```
[server]$ sudo vi /var/spool/cron/crontabs/username
```

See also

- [SSH overview](#)
- [Crontab overview](#)
- [Execution environment of a cron job](#)
- [How do I create a cron job?](#)
- [Command line PHP overview](#)
- [Cron job troubleshooting](#)

Did this article answer your questions?

Yes

No

Article last updated September 14, 2021 10:18 PST.



RELATED ARTICLES

[How do I create a cron job?](#)

[Crontab overview](#)

[Execution environment of a cron job](#)

[Creating a user with Shell \(SSH\) access](#)

[SSH overview](#)

Still not finding what you're looking for?

[Contact Support](#)

DISCUSSION FORUM

Talk with other users

STATUS UPDATES

Go subscribe

DREAMHOST ACADEMY

Learn Wordpress

BLOG & NEWS

Explore articles



Copyright © 2021 DreamHost, LLC.