interweb

Generated on 22/10/2026 08:28

KNOWLEDGE BASE · PUBLIC DOCUMENT

Cron jobs: what they are for and how to create one

Automation & n8n

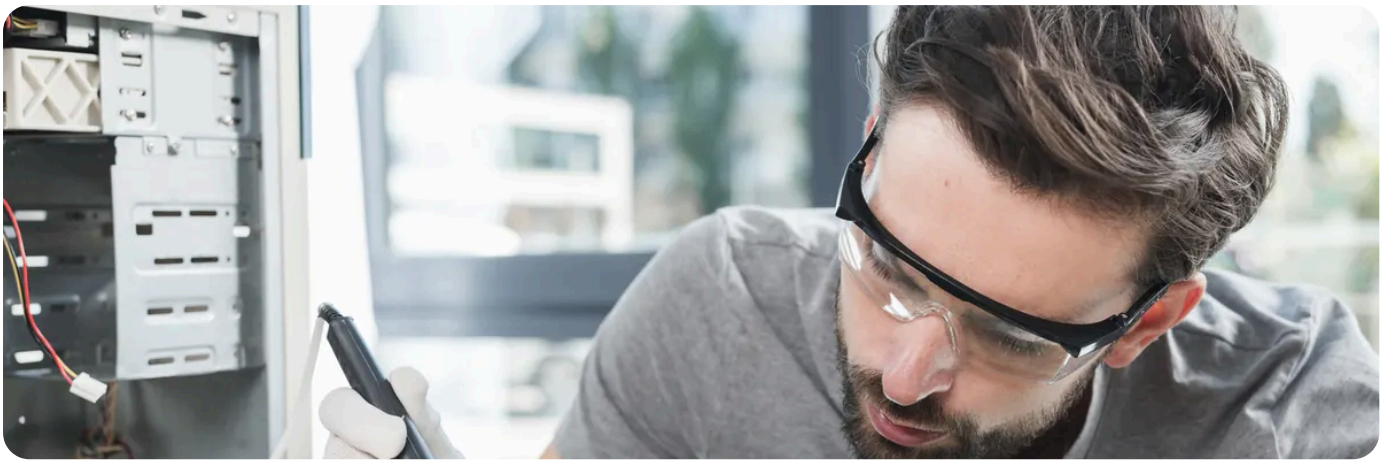
CODE	VERSION	COUNTRY · LANGUAGE
RLX-MZ-KB-458	v2026.10.18	Mozambique · English

Interweb Mozambique
RLX Holdings, S.A. · Group RLX Holdings
Tax ID (NUIT) 401 191 607 · Legal entity no. (NUEL) 105 045 974 · Economic activity code (CAE) 62022

https://www.meu.interweb.co.mz/knowledgebase/458

Cron jobs: what they are for and how to create one

Automation & n8n Created on **September 26, 2026** Updated on **October 18, 2026** 6 min read



A scheduled task, or **cron job**, is an instruction the server runs at set times, whether you are there or not, whether the site has visitors or not. It is one of the features most people have and fewest people know they have, and it explains half of every «this was supposed to happen by itself and it did not».

What it is actually for

Who needs it	For what
WordPress	Scheduled posts, update checks, emptying the trash, and the e-mails your plugins send. All of it goes through its internal scheduler.
Online shops	Syncing stock, closing abandoned carts, issuing invoices, sending the «your order has shipped» message.
Your own scripts	Importing a file a supplier drops every night, deleting temporary files, building a report at 7 in the morning.
Your own backups	A script that exports the database and pushes it off the server. The backup nobody remembers to take by hand is the one a scheduled task never forgets.

The WordPress case, which is worth its own section

The WordPress scheduler is not a cron job: it is a file that only runs **when somebody visits the site**. On a busy site you never notice. On a company site with a handful of visits a day, tasks sit waiting and then all fire at once, on top of the next visitor, who is left staring at a blank page.

The fix is to take the job away from it: turn the internal scheduler off and create a real timed task. Add this to `wp-config.php`, above the line that says to stop editing:

```
define( 'DISABLE_WP_CRON', true );
```

then create the task in the panel, calling `wp-cron.php` on a schedule. Every five minutes is plenty for almost any site.

Turning it off without creating the task is worse than not touching it.
Add the line to `wp-config.php` and forget the rest and scheduled posts stop going out, shop e-mails stop being sent, and nothing throws an error. Do both in the same sitting, or neither.

How to create one

1. In cPanel, under the **Advanced** section, open the scheduled tasks (cron). If you cannot see the icon, type *cron* into the search box at the top of the panel.

2. **Pick the frequency.** There is a list of common choices (hourly, once a day, once a week) that fills the fields in for you. You only need the five fields when you want a time that is not on the list.

3. **Write the command.** This is the line the server will run, with **full paths**. Cron does not know which folder you were in and does not know the shortcuts your terminal knows.

4. **Save, and put your own address** in the notification field at the top of the page. For the first few days you want to receive whatever the task prints, so you know it runs.

Field	What it is
Minute	0 to 59. `0` means on the hour.
Hour	0 to 23, in server time, which may not be yours. Check before scheduling anything at midnight.
Day of month	1 to 31. `*` means every one.
Month	1 to 12.
Day of week	0 to 6, starting on Sunday.

Read together: `*/5 * * * *` is every five minutes, `30 4 * * *` is 4:30 every day, and `0 6 * * 1` is 6 in the morning every Monday.

Two ways to call the same thing.
A PHP script can be run straight by the interpreter, with the full path to PHP and the full path to the file, or requested as if it were a web page, with `wget` or `curl`. The first is faster and never touches the web; the second is what plugins usually ask for. If your application hands you a ready-made line, use theirs.

Why every minute is a bad idea

It looks harmless and it is not. Every run starts a process, and that process counts against your account limits exactly like a visitor does. Sixty starts an hour, 1440 a day, every day, competing with real visitors for the same resources.

And then they stack.
If the task takes longer than a minute to finish, the next one starts before the last one is done. Then the third. An hour later you have dozens of copies of the same script running at once, often touching the same data. The symptom is a site that is slow for no visible reason, and the culprit is the task that «does nothing». Every five minutes covers practically everything people schedule every minute.

How to tell whether it ran

1. **By e-mail.** By default, anything the task prints is sent to the address you gave. If the mail arrives, it ran. If nothing arrives and the task was supposed to print something, it did not.

2. **By a log file.** Append `>> /home/username/cron.log 2>&1` to the command, with your own path. You now have the output and the errors saved, with history, without filling your mailbox.

3. **By the task itself.** Have the script write the date and the result into a table or a file. It is the only proof that it not only started but reached the end.

4. **By the error log.** In the Metrics section of cPanel, PHP errors show up whether the script was run by cron or by a visitor.

Once you trust it
, silence it. Adding `>/dev/null 2>&1` to the end of the command throws the output away and stops the daily mail. Do that after you have watched it run, never before: a silent task that never worked looks exactly like one that does.

The three usual mistakes.
Relative paths (cron does not start in your site folder); the wrong interpreter (the task runs on one PHP version and the site on another, so extensions go missing); and server time, which may not be your time. On that last one, see «What time the server is on, and why PHP and MySQL can disagree».

NEED A HAND?

Got a script and cannot work out the cron line? Send us the script and the schedule you want.

Open a support ticket →

Where to see it in your account

Every run of a scheduled task is a process that counts towards your account. We do not publish a fixed number for incoming processes, CPU, RAM or I/O, so the way to know is to look:

- After creating a task, open Resource Usage in cPanel. It shows CPU, memory, incoming processes and I/O, and when the account reached its limit.
- To test a command before scheduling it, run it once by hand in the cPanel Terminal, which every shared plan includes.
- If the account keeps reaching the limit even with a sensible schedule, you can move up a plan without losing anything, or talk to us.

SEE ALSO

- What is included in each plan →
- How far our help goes: the Support Policy →

interweb

Interweb Mozambique

RLX Holdings, S.A. · Group RLX Holdings

TAX ID (NUIT)	401 191 607
LEGAL ENTITY NO. (NUEL)	105 045 974
ECONOMIC ACTIVITY CODE (CAE)	62022
REGISTERED OFFICE	Bairro 7 de Setembro Vilankulo 1304 · CEP 0411-02, Inhambane, Mozambique
CONTACT	info@interweb.co.mz · +258 84 635 1111 · +258 86 635 1111 · www.interweb.co.mz
OPENING HOURS	Monday to Friday - 08 AM to 04 PM Saturday & Sunday - Closed

Online version of this document

RLX-MZ-KB-458 · v2026.10.18 · Generated on 22/10/2026 08:28