**How To Install and Manage Supervisor | DigitalOcean**

1. How To Install and Manage Supervisor | DigitalOcean
2. Introduction
3. Prerequisites
4. Step 1 - Installation
5. Step 2 - Adding a Program
6. Step 3 - Managing Programs
7. Conclusion

### [Introduction](chrome-extension://odfonlkabodgbolnmmkdijkaeggofoop/pages/content/app.html?url=https%3A%2F%2Fwww.digitalocean.com%2Fcommunity%2Ftutorials%2Fhow-to-install-and-manage-supervisor-on-ubuntu-and-debian-vps#introduction)

In many VPS environments, it is often the case that you will have a number of small programs that you want to run persistently, whether these are small shell scripts, Node.js apps, or any large-sized packages.

Usually, external packages are supplied with a unit file that allows them to be managed by an init system such as [systemd](https://www.digitalocean.com/community/tutorials/how-to-use-systemctl-to-manage-systemd-services-and-units), or packaged as [docker](https://www.digitalocean.com/community/tutorial_collections/how-to-install-and-use-docker) images which can be managed by a container engine. However, for software that isn’t well-packaged, or for users who would prefer not to interact with a low-level init system on their server, it is helpful to have a lightweight alternative.

[Supervisor](http://supervisord.org/) is a process manager which provides a singular interface for managing and monitoring a number of long-running programs. In this tutorial, you will install Supervisor on a Linux server and learn how to manage Supervisor configurations for multiple applications.

## [Prerequisites](chrome-extension://odfonlkabodgbolnmmkdijkaeggofoop/pages/content/app.html?url=https%3A%2F%2Fwww.digitalocean.com%2Fcommunity%2Ftutorials%2Fhow-to-install-and-manage-supervisor-on-ubuntu-and-debian-vps#prerequisites)

To complete this guide, you will need:

* An Linux server and a non-root user with sudo privileges. You can learn more about how to set up a user with these privileges in our [Initial Server Setup with Ubuntu 20.04](https://www.digitalocean.com/community/tutorials/initial-server-setup-with-ubuntu-20-04) guide.

## [Step 1 - Installation](chrome-extension://odfonlkabodgbolnmmkdijkaeggofoop/pages/content/app.html?url=https%3A%2F%2Fwww.digitalocean.com%2Fcommunity%2Ftutorials%2Fhow-to-install-and-manage-supervisor-on-ubuntu-and-debian-vps#step-1-installation)

Begin by updating your package sources and installing Supervisor:

sudo apt update && sudo apt install supervisor

The supervisor service runs automatically after installation. You can check its status:

sudo systemctl status supervisor

You should receive the following output:

Output● supervisor.service - Supervisor process control system for UNIX

Loaded: loaded (/lib/systemd/system/supervisor.service; enabled; vendor preset: enabled)

Active: active (running) since Wed 2021-11-17 22:56:48 UTC; 5min ago

Now that we have Supervisor installed, we can look at adding our first programs.

## [Step 2 - Adding a Program](chrome-extension://odfonlkabodgbolnmmkdijkaeggofoop/pages/content/app.html?url=https%3A%2F%2Fwww.digitalocean.com%2Fcommunity%2Ftutorials%2Fhow-to-install-and-manage-supervisor-on-ubuntu-and-debian-vps#step-2-adding-a-program)

A best practice for working with Supervisor is to write a configuration file for every program it will handle.

All programs run under Supervisor must be run in a [non-daemonising mode](http://supervisord.org/subprocess.html#nondaemonizing-of-subprocesses) (sometimes also called ‘foreground mode’). If, by default, your program automatically returns to the shell after running, then you may need to consult the program’s manual to find the option to enable this mode, or Supervisor will not be able to properly determine the status of the program.

In order to demonstrate Supervisor’s functionality, we’ll create a shell script that does nothing other than produce some predictable output once a second, but will run continuously in the background until it is manually stopped. Using nano or your favorite text editor, open a file called idle.sh in your home directory:

Add the following contents:

~/idle.sh

#!/bin/bash

while true

do

# Echo current date to stdout

echo `date`

# Echo 'error!' to stderr

echo 'error!' >&2

sleep 1

done

Save and close the file. If you are using nano, press Ctrl+X, then when prompted, Y and Enter.

Next, make your script executable:

The per-program configuration files for Supervisor programs are located in the /etc/supervisor/conf.d directory, typically running one program per file and ending in .conf. We’ll create a configuration file for this script, as`/etc/supervisor/conf.d/idle.conf:

sudo nano /etc/supervisor/conf.d/idle.conf

Add these contents:

/etc/supervisor/conf.d/idle.conf

command=/home/ubuntu/idle.sh

autostart=true

autorestart=true

stderr\_logfile=/var/log/idle.err.log

stdout\_logfile=/var/log/idle.out.log

We’ll review this line by line:

command=/home/ubuntu/idle.sh

The configuration begins by defining a program with the name idle and the full path to the program:

autostart=true

autorestart=true

The next two lines define the automatic behavior of the script under certain conditions.

The autostart option tells Supervisor that this program should be started when the system boots. Setting this to false will require a manual start following any system shutdown.

autorestart defines how Supervisor should manage the program in the event that it exits:

* false tells Supervisor not to ever restart the program after it exits.
* true tells Supervisor to always restart the program after it exits.
* unexpected tells Supervisor to only restart the program if it exits with an unexpected error code (by default anything other than codes 0 or 2). To learn more about error codes, look into the errno command.

stderr\_logfile=/var/log/idle.err.log

stdout\_logfile=/var/log/idle.out.log

The final two lines define the locations of the two main log files for the program. As suggested by the option names, stdout and stderr will be directed to the stdout\_logfile and stderr\_logfile locations respectively. The specified directories must already exist, as Supervisor will not attempt to create any missing directories.

The configuration we have created here is a minimal template for a Supervisor program. [The Supervisor documentation](http://supervisord.org/configuration.html#program-x-section-settings) lists many more optional configuration options that are available to tune how programs are run.

Once our configuration file is created and saved, we can inform Supervisor of our new program through the supervisorctl command. First we tell Supervisor to look for any new or changed program configurations in the /etc/supervisor/conf.d directory with:

sudo supervisorctl reread

Outputidle: available

Followed by telling it to enact any changes with:

sudo supervisorctl update

Outputidle: added process group

Any time you make a change to any program configuration file, running the two previous commands will bring the changes into effect.

At this point our program should now be running. We can check its output by looking at the output log file:

sudo tail /var/log/idle.out.log

OutputSat Nov 20 22:21:22 UTC 2021

Sat Nov 20 22:21:23 UTC 2021

Sat Nov 20 22:21:24 UTC 2021

Sat Nov 20 22:21:25 UTC 2021

Sat Nov 20 22:21:26 UTC 2021

Sat Nov 20 22:21:27 UTC 2021

Sat Nov 20 22:21:28 UTC 2021

Sat Nov 20 22:21:29 UTC 2021

Sat Nov 20 22:21:30 UTC 2021

Sat Nov 20 22:21:31 UTC 2021

Next, we’ll cover some other usage of Supervisor.

## [Step 3 - Managing Programs](chrome-extension://odfonlkabodgbolnmmkdijkaeggofoop/pages/content/app.html?url=https%3A%2F%2Fwww.digitalocean.com%2Fcommunity%2Ftutorials%2Fhow-to-install-and-manage-supervisor-on-ubuntu-and-debian-vps#step-3-managing-programs)

Beyond running programs, you will want to stop, restart, or see their status. The supervisorctl program, which we used in Step 2, also has an interactive mode which we can use to control our programs.

To enter the interactive mode, run supervisorctl with no arguments:

Outputidle RUNNING pid 12614, uptime 1:49:37

supervisor>

supervisorctl will initially print the status and uptime of all configured programs, followed by its command prompt. Entering help will reveal all of its available commands:

Outputdefault commands (type help <topic>):

=====================================

add clear fg open quit remove restart start stop update

avail exit maintail pid reload reread shutdown status tail version

You can start or stop a program with the associated commands followed by the program name:

Outputidle: stopped

Outputidle: started

Using the tail command, you can view the most recent entries in the stdout and stderr logs for your program:

OutputSun Nov 21 00:36:10 UTC 2021

Sun Nov 21 00:36:11 UTC 2021

Sun Nov 21 00:36:12 UTC 2021

Sun Nov 21 00:36:13 UTC 2021

Sun Nov 21 00:36:14 UTC 2021

Sun Nov 21 00:36:15 UTC 2021

Sun Nov 21 00:36:17 UTC 2021

supervisor> tail idle stderr

Outputerror!

error!

error!

error!

error!

error!

error!

Using status you can view again the current execution state of each program after making any changes:

Outputidle STOPPED Nov 21 01:07 AM

Finally, you can exit supervisorctl with Ctrl+C or by entering quit into the prompt:

## [Conclusion](chrome-extension://odfonlkabodgbolnmmkdijkaeggofoop/pages/content/app.html?url=https%3A%2F%2Fwww.digitalocean.com%2Fcommunity%2Ftutorials%2Fhow-to-install-and-manage-supervisor-on-ubuntu-and-debian-vps#conclusion)

In this tutorial, you learned how to install and manage Supervisor. As mentioned, Supervisor is very lightweight by modern standards, but it continues to be well-maintained, and it can be a useful tool for smaller deployments. It is also a low-maintenance and self-contained way of generating logs as a component part of a larger deployment.

If you’re running multiple small apps that need to be web accessible, you may also want to read about [configuring Nginx as a reverse proxy](https://www.digitalocean.com/community/tutorials/how-to-configure-nginx-as-a-web-server-and-reverse-proxy-for-apache-on-one-ubuntu-20-04-server), which is another fundamental component of small, reusable deployments.