Search Docker Hub Explore Pricing

Explore › Official Images › odoo ›

odoo

DOCKER OFFICIAL IMAGE • 10M+ • ☆1.2K

Odoo (formerly known as OpenERP) is a suite of open-source business apps.

```
docker pull odoo
```

Overview Tags

Quick reference

- **Maintained by**:
  Odoo
- **Where to get help**:
  the Docker Community Slack, Server Fault, Unix & Linux, or Stack Overflow

Supported tags and respective `Dockerfile` links

- `16.0`, `16`, `latest`
- `15.0`, `15`
- `14.0`, `14`

Quick reference (cont.)

- **Where to file issues**:
  https://github.com/odoo/docker/issues

- **Supported architectures**: (more info)
  `amd64`

- **Published image artifact details**:
  repo-info repo's `repos/odoo/` directory (history)
  (image metadata, transfer size, etc)

- **Image updates**:
  official-images repo's `library/odoo` label
  official-images repo's `library/odoo` file (history)

- **Source of this description**:
  docs repo's `odoo/` directory (history)

What is Odoo?

Odoo, formerly known as OpenERP, is a suite of open-source business apps written in Python and released under the LGPL license. This suite of applications covers all business needs, from Website/Ecommerce down to manufacturing, inventory and accounting, all seamlessly integrated. It is the first time ever a software editor managed to reach such a functional coverage. Odoo is the most installed business software in the world. Odoo is used by 2.000.000 users worldwide ranging from very small companies (1 user) to very large ones (300 000 users).

> www.odoo.com

How to use this image

This image requires a running PostgreSQL server.

Start a PostgreSQL server

```
$ docker run -d -e POSTGRES_USER=odoo -e POSTGRES_PASSWORD=odoo -e POSTGRES_D
```

Start an Odoo instance

```
$ docker run -p 8069:8069 --name odoo --link db:db -t odoo
```

The alias of the container running Postgres must be db for Odoo to be able to connect to the Postgres server.

Stop and restart an Odoo instance

```
$ docker stop odoo
$ docker start -a odoo
```

Use named volumes to preserve data

When the Odoo container is created like described above, the odoo filestore is created inside the container. If the container is removed, the filestore is lost. The preferred way to prevent that is by using a Docker named volume.

```
$ docker run -v odoo-data:/var/lib/odoo -d -p 8069:8069 --name odoo --link db
```

With the above command, the volume named `odoo-data` will persist even if the container is removed and can be re-used by issuing the same command.

The path `/var/lib/odoo` used as the mount point of the volume must match the odoo `data_dir` in the config file or as CLI parameters.

Note that the same principle applies to the Postgresql container and a named volume can be used to preserve the database when the container is removed. So the database container could be started like this (before the odoo container):

```
$ docker run -d -v odoo-db:/var/lib/postgresql/data -e POSTGRES_USER=odoo -e
```

Stop and restart a PostgreSQL server

When a PostgreSQL server is restarted, the Odoo instances linked to that server must be restarted as well because the server address has changed and the link is thus broken.

Restarting a PostgreSQL server does not affect the created databases.

Run Odoo with a custom configuration

The default configuration file for the server (located at `/etc/odoo/odoo.conf`) can be overriden at startup using volumes. Suppose you have a custom configuration at `/path/to/config/odoo.conf` , then

```
$ docker run -v /path/to/config:/etc/odoo -p 8069:8069 --name odoo --link db:
```

Please use this configuration template to write your custom configuration as we already set some arguments for running Odoo inside a Docker container.

You can also directly specify Odoo arguments inline. Those arguments must be given after the keyword `--` in the command-line, as follows

```
$ docker run -p 8069:8069 --name odoo --link db:db -t odoo -- --db-filter=odo
```

Mount custom addons

You can mount your own Odoo addons within the Odoo container, at `/mnt/extra-addons`

```
$ docker run -v /path/to/addons:/mnt/extra-addons -p 8069:8069 --name odoo --
```

Note: Altough there is no official Odoo Enterprise Docker image, the Enterprise modules can be mounted by using the above mentionned method.

Run multiple Odoo instances

```
$ docker run -p 8070:8069 --name odoo2 --link db:db -t odoo
$ docker run -p 8071:8069 --name odoo3 --link db:db -t odoo
```

Note: For plain use of mails and reports functionalities, when the host and container ports differ (e.g. 8070 and 8069), one has to set, in Odoo, `Settings->Parameters->System Parameters` (requires technical features), web.base.url to the container port (e.g. 127.0.0.1:8069).

Environment Variables

Tweak these environment variables to easily connect to a postgres server:

- `HOST` : The address of the postgres server. If you used a postgres container, set to the name of the container. Defaults to `db` .
- `PORT` : The port the postgres server is listening to. Defaults to `5432` .
- `USER` : The postgres role with which Odoo will connect. If you used a postgres container, set to the same value as `POSTGRES_USER` . Defaults to `odoo` .
- `PASSWORD` : The password of the postgres role with which Odoo will connect. If you used a postgres container, set to the same value as `POSTGRES_PASSWORD` . Defaults to `odoo` .

Docker Compose examples

The simplest `docker-compose.yml` file would be:

```
version: '3.1'
services:
  web:
    image: odoo:16.0
    depends_on:
      - db
    ports:
      - "8069:8069"
  db:
    image: postgres:15
    environment:
      - POSTGRES_DB=postgres
      - POSTGRES_PASSWORD=odoo
      - POSTGRES_USER=odoo
```

If the default postgres credentials does not suit you, tweak the environment variables:

```
version: '3.1'
services:
  web:
    image: odoo:16.0
    depends_on:
      - mydb
    ports:
      - "8069:8069"
    environment:
    - HOST=mydb
    - USER=odoo
    - PASSWORD=myodoo
  mydb:
    image: postgres:15
    environment:
      - POSTGRES_DB=postgres
      - POSTGRES_PASSWORD=myodoo
      - POSTGRES_USER=odoo
```

Here's a last example showing you how to

- mount custom addons located in `./addons`
- use a custom configuration file located in `.config/odoo.conf`
- use named volumes for the Odoo and postgres data dir

- use a `secrets` file named `odoo_pg_pass` that contains the postgreql password shared by both services

```
version: '3.1'
services:
  web:
    image: odoo:16.0
    depends_on:
      - db
    ports:
      - "8069:8069"
    volumes:
      - odoo-web-data:/var/lib/odoo
      - ./config:/etc/odoo
      - ./addons:/mnt/extra-addons
    environment:
      - PASSWORD_FILE=/run/secrets/postgresql_password
    secrets:
      - postgresql_password
  db:
    image: postgres:15
    environment:
      - POSTGRES_DB=postgres
      - POSTGRES_PASSWORD_FILE=/run/secrets/postgresql_password
      - POSTGRES_USER=odoo
      - PGDATA=/var/lib/postgresql/data/pgdata
    volumes:
      - odoo-db-data:/var/lib/postgresql/data/pgdata
    secrets:
      - postgresql_password
volumes:
  odoo-web-data:
  odoo-db-data:

secrets:
  postgresql_password:
    file: odoo_pg_pass
```

To start your Odoo instance, go in the directory of the `docker-compose.yml` file you created from the previous examples and type:

```
docker-compose up -d
```

How to upgrade this image

Odoo images are updated on a regular basis to make them use recent releases (a new release of each version of Odoo is built every night). Please be aware that what follows is about upgrading from an old release to the latest one provided of the same major version, as upgrading from a major version to another is a much more complex process requiring elaborated migration scripts (see Odoo Upgrade page or this community project which aims to write those scripts).

Suppose you created a database from an Odoo instance named old-odoo, and you want to access this database from a new Odoo instance named new-odoo, e.g. because you've just downloaded a newer Odoo image.

By default, Odoo 16.0 uses a filestore (located at `/var/lib/odoo/filestore/`) for attachments. You should restore this filestore in your new Odoo instance by running

```
$ docker run --volumes-from old-odoo -p 8070:8069 --name new-odoo --link db:d
```

License

View license information for the software contained in this image.

As with all Docker images, these likely also contain other software which may be under other licenses (such as Bash, etc from the base distribution, along with any direct or indirect dependencies of the primary software being contained).

Some additional license information which was able to be auto-detected might be found in the `repo-info` repository's `odoo/` directory.

As for any pre-built image usage, it is the image user's responsibility to ensure that any use of this image complies with any relevant licenses for all software contained within.

Recent Tags

latest 16.0 16 15.0 15 14.0 14 13.0 13 12.0

About Official Images

Docker Official Images are a curated set of Docker open source and drop-in solution repositories.

Why Official Images?

These images have clear documentation, promote best practices, and are designed for the most common use cases.

Why

Overview

What is a Container

Products

Product Overview

Product Offerings

Docker Desktop

Docker Hub

Features

Container Runtime

Developer Tools

Docker App

Kubernetes

Developers

Getting Started

Play with Docker

Community

Open Source

Documentation

Company

About Us

Resources

Blog

Customers

Partners

Newsroom

Events and Webinars

Careers

Contact Us

| Terms of Service | Subscription Service Agreement | Privacy | Legal

Cookies Settings