Instalação Grocy Docker

Link: https://docs.linuxserver.io/images/docker-grocy/

Customizações especiais não oficiais:

1) Feito mapeamento do arquivo de tradução do português brasileiro (strings.po) dentro do "docker-compose.yml" com a seguinte entrada:
./volumes/custom/strings.po:/app/www/localization/pt_BR/strings.po:rw
No /opt/docker/grocy : **vim volumes/custom/strings.po**

2) Configurações de layout, de parâmetros especiais e para supressão de módulos do Grocy:
vim volumes/config/data/config.php

linuxserver/grocy

Grocy is an ERP system for your kitchen! Cut down on food waste, and manage your chores with this brilliant utility.

Keep track of your purchases, how much food you are wasting, what chores need doing and what batteries need charging with this proudly Open Source tool

For more information on grocy visit their website and check it out: https://grocy.info

grocy

Supported Architectures¶

We utilise the docker manifest for multi-platform awareness. More information is available from docker here and our announcement here.

Simply pulling `lscr.io/linuxserver/grocy:latest` should retrieve the correct image for your arch, but you can also pull specific arch images via tags.

The architectures supported by this image are:

Architecture	Available	Tag
x86-64	✔	amd64-<version tag>
arm64	✔	arm64v8-<version tag>
armhf	✔	

Application Setup

Grocy is simple to get running. Configure the container with instructions below, start it, and you can then access it by visiting http://your.ip:9283 - once the page loads, you can log in with the default username and password of admin / admin

Upgrading

Following a container upgrade ensure that you visit the root (`/`) route (click on the logo in the left upper edge) in order to run any necessary database migrations. See https://github.com/grocy/grocy#how-to-update for more details.

Usage

To help you get started creating a container from this image you can either use docker-compose or the docker cli.

Info

Unless a parameter is flaged as 'optional', it is *mandatory* and a value must be provided.

docker-compose (recommended, click here for more info)

```
---
services:
  grocy:
    image: lscr.io/linuxserver/grocy:latest
    container_name: grocy
    environment:
      - PUID=1000
```

```
      - PGID=1000
      - TZ=Etc/UTC
    volumes:
      - /path/to/grocy/config:/config
    ports:
      - 9283:80
    restart: unless-stopped
```

docker cli (click here for more info)

```
docker run -d \
  --name=grocy \
  -e PUID=1000 \
  -e PGID=1000 \
  -e TZ=Etc/UTC \
  -p 9283:80 \
  -v /path/to/grocy/config:/config \
  --restart unless-stopped \
  lscr.io/linuxserver/grocy:latest
```

Parameters

Containers are configured using parameters passed at runtime (such as those above). These parameters are separated by a colon and indicate `<external>:<internal>` respectively. For example, `-p 8080:80` would expose port `80` from inside the container to be accessible from the host's IP on port `8080` outside the container.

Ports (`-p`)

Parameter	Function
`9283:80`	will map the container's port 80 to port 9283 on the host

Environment Variables (`-e`)

Env	Function
`PUID=1000`	for UserID - see below for explanation
`PGID=1000`	for GroupID - see below for explanation
`TZ=Etc/UTC`	specify a timezone to use, see this list.

Volume Mappings (`-v`)

Volume	Function
`/config`	Persistent config files

Miscellaneous Options

Parameter	Function

Environment variables from files (Docker secrets)

You can set any environment variable from a file by using a special prepend `FILE__`.

As an example:

```
-e FILE__MYVAR=/run/secrets/mysecretvariable
```

Will set the environment variable `MYVAR` based on the contents of the `/run/secrets/mysecretvariable` file.

Umask for running applications

For all of our images we provide the ability to override the default umask settings for services started within the containers using the optional `-e UMASK=022` setting. Keep in mind umask is not chmod it subtracts from permissions based on it's value it does not add. Please read up here before asking for support.

User / Group Identifiers

When using volumes (`-v` flags), permissions issues can arise between the host OS and the container, we avoid this issue by allowing you to specify the user `PUID` and group `PGID`.

Ensure any volume directories on the host are owned by the same user you specify and any permissions issues will vanish like magic.

In this instance `PUID=1000` and `PGID=1000`, to find yours use `id your_user` as below:

```
id your_user
```

Example output:

```
uid=1000(your_user) gid=1000(your_user) groups=1000(your_user)
```

Docker Mods

Docker Mods Docker Universal Mods

We publish various Docker Mods to enable additional functionality within the containers. The list of Mods available for this image (if any) as well as universal mods that can be applied to any one of our images can be accessed via the dynamic badges above.

Support Info

- Shell access whilst the container is running:

```
docker exec -it grocy /bin/bash
```

- To monitor the logs of the container in realtime:

```
docker logs -f grocy
```

- Container version number:

```
docker inspect -f '{{ index .Config.Labels "build_version" }}' grocy
```

- Image version number:

```
docker inspect -f '{{ index .Config.Labels "build_version" }}' lscr.io/linuxserver/grocy:latest
```

Updating Info

Most of our images are static, versioned, and require an image update and container recreation to update the app inside. With some exceptions (noted in the relevant readme.md), we do not recommend or support updating apps inside the container. Please consult the Application Setup section above to see if it is recommended for the image.

Below are the instructions for updating containers:

Via Docker Compose

- Update images:
  - All images:

    ```
    docker-compose pull
    ```

  - Single image:

    ```
    docker-compose pull grocy
    ```

- Update containers:
  - All containers:

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

  - Single container:

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

- You can also remove the old dangling images:

  ```
  docker image prune
  ```

Via Docker Run

- Update the image:

```
docker pull lscr.io/linuxserver/grocy:latest
```

- Stop the running container:

```
docker stop grocy
```

- Delete the container:

```
docker rm grocy
```

- Recreate a new container with the same docker run parameters as instructed above (if mapped correctly to a host folder, your `/config` folder and settings will be preserved)
- You can also remove the old dangling images:

```
docker image prune
```

Image Update Notifications - Diun (Docker Image Update Notifier)

Tip

We recommend Diun for update notifications. Other tools that automatically update containers unattended are not recommended or supported.

Building locally

If you want to make local modifications to these images for development purposes or just to customize the logic:

```
git clone https://github.com/linuxserver/docker-grocy.git
cd docker-grocy
docker build \
  --no-cache \
  --pull \
  -t lscr.io/linuxserver/grocy:latest .
```

The ARM variants can be built on x86_64 hardware and vice versa using `lscr.io/linuxserver/qemu-static`

```
docker run --rm --privileged lscr.io/linuxserver/qemu-static --reset
```

Once registered you can define the dockerfile to use with `-f Dockerfile.aarch64`.

To help with development, we generate this dependency graph.

Init dependency graph

Versions

- **30.06.24:** - Rebase to Alpine 3.20. Existing users should update their nginx confs to avoid http2 deprecation warnings.
- **29.03.24:** - Add `clear_env = no` to `php-fpm` to pass on environment variables to workers threads
- **06.03.24:** - Existing users should update: site-confs/default.conf - Cleanup default site conf.
- **06.03.24:** - Rebase to Alpine 3.19 with php 8.3.
- **25.05.23:** - Rebase to Alpine 3.18, deprecate armhf.
- **13.04.23:** - Move ssl.conf include to default.conf.
- **19.01.23:** - Rebase to alpine 3.17 with php8.1.
- **20.08.22:** - Rebasing to alpine 3.15 with php8. Restructure nginx configs (see changes announcement).
- **22.08.21:** - Rebase to Alpine 3.14 and PHP 8.
- **25.07.21:** - Add 'int','json' and 'zlib' PHP extensions.
- **10.05.21:** - Reduce image size.
- **08.04.21:** - Update docs to reflect jenkins builder changes.
- **17.02.21:** - Rebasing to alpine 3.13.
- **26.01.21:** - Add 'ldap' PHP extension.
- **22.12.20:** - Add 'ctype' PHP extension.
- **01.06.20:** - Rebasing to alpine 3.12.
- **19.12.19:** - Rebasing to alpine 3.11.
- **22.09.19:** - Add 'gd' PHP extension.
- **28.06.19:** - Rebasing to alpine 3.10.
- **23.03.19:** - Switching to new Base images, shift to arm32v7 tag.
- **22.02.19:** - Rebasing to alpine 3.9.
- **27.12.18:** - Initial Release.

January 31, 2025 ebruary 6, 2019

Revision #3
Created 2025-04-18 21:47:51 UTC by Administrador
Updated 2025-04-18 22:11:53 UTC by Administrador