# Contributing

Oracle welcomes contributions to this repository from anyone.

If you want to submit a pull request to fix a bug or enhance an existing

`Dockerfile`, please first open an issue and link to that issue when you

submit your pull request.

If you have any questions about a possible submission, feel free to open

an issue too.

## Contributing to the Oracle Docker Images repository

Pull requests can be made under

[The Oracle Contributor Agreement](https://www.oracle.com/technetwork/community/oca-486395.html) (OCA).

For pull requests to be accepted, the bottom of your commit message must have

the following line using your name and e-mail address as it appears in the

OCA Signatories list.

```

Signed-off-by: Your Name <you@example.org>

```

This can be automatically added to pull requests by committing with:

```

git commit --signoff

```

Only pull requests from committers that can be verified as having

signed the OCA can be accepted.

## Oracle Product Ownership and Responsibility

For any new product content, \*you must obtain internal Oracle approvals for the

distribution of this content prior to submitting a pull request\*. If you are

unfamiliar with the approval process to submit code to an existing GitHub

repository, please contact the [Oracle Open Source team](mailto:opensource\_ww\_grp@oracle.com)

for details.

The GitHub user who submits the initial pull request to add a new product image

should add themselves to the [code owner](./CODEOWNERS) file in that same

request. This will flag the user as the owner of the content and any future pull

requests that affect the conten will need to be approved by this user.

The code owner will also be assigned to any issues relating to their content.

You must ensure that you check the [issues](https://github.com/oracle/docker-images/issues)

on at least a weekly basis, though daily is preferred.

If you wish to nominate additional or alternative users, they must be a visible

member of the [Oracle GitHub Organisation](https://github.com/orgs/oracle/people/).

Contact [Avi Miller](https://github.com/Djelibeybi) for more information.

### Pull request process

1. Fork this repository

1. Create a branch in your fork to implement the changes. We recommend using

the issue number as part of your branch name, e.g. `1234-fixes`

1. Ensure that any documentation is updated with the changes that are required

by your fix.

1. Ensure that any samples are updated if the base image has been changed.

1. Submit the pull request. \*Do not leave the pull request blank\*. Explain exactly

what your changes are meant to do and provide simple steps on how to validate

your changes. Ensure that you reference the issue you created as well.

We will assign the pull request to 2-3 people for review before it is merged.

## Golden Rules

We have some golden rules that we require all submitted `Dockerfiles` to abide

by. These rules are provided by Oracle Global Product Security and may change

at any time.

Most of these are targeted at Oracle employees, but apply to anyone who submits

a pull request.

### Base Image Rules

1. Extend an existing product image wherever possible. For example, if your

product requires WebLogic, then extend the WebLogic image instead of creating

your own WebLogic installation.

1. If you can't extend an existing image, your image must use the

`oraclelinux:7-slim` base image as this image provides the smallest

attack surface and is updated whenever a CVE errata is published.

1. Re-use existing scripts wherever possible. If a particular base image or

script doesn't have the functionality you need, open an issue and work with

the image owner to implement it.

1. Specify a version in the `FROM` directive, i.e. use

`FROM oraclelinux:7-slim` or `FROM java/serverjre:8`.

1. All images must provide a `CMD` or `ENTRYPOINT`. If your image is designed

to be extended, then this should output documentation on how to extend the

image to be useful.

1. Use `LABEL` instructions for additional information such as ports and volumes. The following are common label instructions that should be present in all images where applicable:

| Label | Value | Applicability |

| -------- | ----- | ------------- |

| provider | `Oracle` | All images |

| issues | `https://github.com/oracle/docker-images/issues` | All images |

| maintainer | Name of the maintainer | At the discretion of the author. |

| volume(.`purpose`) | Use `volume` labels to describe the volumes of an image.<br/>If your image has multiple volumes, use qualified names to specify the purpose of each volume, for example `volume.data` for data to be persisted outside the container.<br/>Use hierarchical nesting for multiple volumes of the same type, for example:<br/><ul><li>`volume.data.dir1`</li><li>`volume.data.dir2`</li></ul> | Mandatory for all images that require persistent storage beyond the life of an individual container. |

| port(.`purpose`) | Use `port` labels to describe the ports of an image.<br/>If your images has multiple ports, use qualified names to specify the purpose of each port, for example `port.app` for the port on which your application is reachable.<br/>Use hierarchical nesting for multiple ports of the same type, for example:<br/><ul><li>`port.app.http`</li><li>`port.app.https`</li></ul> | Mandatory for all images that require externally accessible port mappings. |

For example, for the Oracle Database 18c XE image we use the following labels:

```

LABEL "provider"="Oracle" \

"issues"="https://github.com/oracle/docker-images/issues" \

"volume.data"="/opt/oracle/oradata" \

"volume.setup.location1"="/opt/oracle/scripts/setup" \

"volume.setup.location2"="/docker-entrypoint-initdb.d/setup" \

"volume.startup.location1"="/opt/oracle/scripts/startup" \

"volume.startup.location2"="/docker-entrypoint-initdb.d/startup" \

"port.listener"="1521" \

"port.oemexpress"="5500" \

"port.apex"="8080"

```

### Security-related Rules

1. Do not require the use of the `--privileged` flag when running a container.

1. Do not run an SSH daemon (`sshd`) inside a container.

1. Do not use host networking mode (`--net=host`) for a container.

1. Do not hard-code any passwords. If passwords are required, generate them

on container startup using `openssl rand` or accept a password argument during

container startup (via `-e`).

### Guidelines and Recommendations

The following are some guidelines that will not prevent an image from being

merged, but are generally frowned upon if breached.

- Always aim to produce the smallest possible image. This means the least amount

of layers (combine directives wherever possible) and cleaning up as much as

possible inside a single directive so the layer only stores the binary changes.

- Don't install all possible required RPMs, even if the product

documentation says so. Some RPMs aren't applicable inside a container, e.g

filesystem utilities (`btrfs-progs`, `ocfs2-tools`, `nfs-utils`).

- Don't install any interactive/user tools, e.g. things like `vim`, `less` or

`man`. Debugging should be done prior to the image submission.

- Don't install `wget` as the base images already include `curl`.

- Always remember to run `rm -rf /var/cache/yum` in the same `RUN` directive as a

`yum install` so that the yum metadata is not stored in the layer.

- Always document any inputs (via `--build-arg` or `-e`) required by

`docker build` or `docker run`. This documentation should also clearly state

any defaults that are used if no input is provided.

- If a custom value must be provided by the end-user, the build or run should

gracefully fail if that value is not provided.

\*Copyright (c) 2017, 2020 Oracle and/or its affiliates.\*