(/)

OFFICIAL REPOSITORY

perl (/r/_/perl/) ☆

Last pushed: 9 days ago

Repo Info (/_/perl/)

Short Description

Perl is a high-level, general-purpose, interpreted, dynamic programming language.

Full Description

Supported tags and respective `Dockerfile` links

- `latest`, `5`, `5.28`, `5.28.0` (*5.028.000-main/Dockerfile*) (https://github.com/perl/docker-perl/blob/892e2b4fbb58c48ee802cd13b34017300c630f18/5.028.000-main/Dockerfile)
- `slim`, `5-slim`, `5.28-slim`, `5.28.0-slim` (*5.028.000-slim/Dockerfile*) (https://github.com/perl/docker-perl/blob/892e2b4fbb58c48ee802cd13b34017300c630f18/5.028.000-slim/Dockerfile)
- `threaded`, `5-threaded`, `5.28-threaded`, `5.28.0-threaded` (*5.028.000-main.threaded/Dockerfile*) (https://github.com/perl/docker-perl/blob/892e2b4fbb58c48ee802cd13b34017300c630f18/5.028.000-main.threaded/Dockerfile)
- `slim-threaded`, `5-slim-threaded`, `5.28-slim-threaded`, `5.28.0-slim-threaded` (*5.028.000-slim.threaded/Dockerfile*) (https://github.com/perl/docker-perl/blob/892e2b4fbb58c48ee802cd13b34017300c630f18/5.028.000-slim.threaded/Dockerfile)
- `5.26`, `5.26.2` (*5.026.002-main/Dockerfile*) (https://github.com/perl/docker-perl/blob/892e2b4fbb58c48ee802cd13b34017300c630f18/5.026.002-main/Dockerfile)
- `5.26-slim`, `5.26.2-slim` (*5.026.002-slim/Dockerfile*) (https://github.com/perl/docker-perl/blob/892e2b4fbb58c48ee802cd13b34017300c630f18/5.026.002-slim/Dockerfile)
- `5.26-threaded`, `5.26.2-threaded` (*5.026.002-main.threaded/Dockerfile*) (https://github.com/perl/docker-

perl/blob/892e2b4fbb58c48ee802cd13b34017300c630f18/5.026.002-main,threaded/Dockerfile)

- `5.26-slim-threaded`, `5.26.2-slim-threaded` (*5.026.002-slim,threaded/Dockerfile*) (https://github.com/perl/docker-perl/blob/892e2b4fbb58c48ee802cd13b34017300c630f18/5.026.002-slim,threaded/Dockerfile)
- `5.24`, `5.24.4` (*5.024.004-main/Dockerfile*) (https://github.com/perl/docker-perl/blob/892e2b4fbb58c48ee802cd13b34017300c630f18/5.024.004-main/Dockerfile)
- `5.24-slim`, `5.24.4-slim` (*5.024.004-slim/Dockerfile*) (https://github.com/perl/docker-perl/blob/892e2b4fbb58c48ee802cd13b34017300c630f18/5.024.004-slim/Dockerfile)
- `5.24-threaded`, `5.24.4-threaded` (*5.024.004-main,threaded/Dockerfile*) (https://github.com/perl/docker-perl/blob/892e2b4fbb58c48ee802cd13b34017300c630f18/5.024.004-main,threaded/Dockerfile)
- `5.24-slim-threaded`, `5.24.4-slim-threaded` (*5.024.004-slim,threaded/Dockerfile*) (https://github.com/perl/docker-perl/blob/892e2b4fbb58c48ee802cd13b34017300c630f18/5.024.004-slim,threaded/Dockerfile)

Quick reference

- **Where to get help**:
  the Docker Community Forums (https://forums.docker.com/), the Docker Community Slack (https://blog.docker.com/2016/11/introducing-docker-community-directory-docker-community-slack/), or Stack Overflow (https://stackoverflow.com/search?tab=newest&q=docker)

- **Where to file issues**:
  https://github.com/Perl/docker-perl/issues (https://github.com/Perl/docker-perl/issues)

- **Maintained by**:
  the Perl Community (https://github.com/Perl/docker-perl)

- **Supported architectures**: (more info (https://github.com/docker-library/official-images#architectures-other-than-amd64))
  `amd64` (https://hub.docker.com/r/amd64/perl/), `arm32v7` (https://hub.docker.com/r/arm32v7/perl/), `arm64v8` (https://hub.docker.com/r/arm64v8/perl/), `i386` (https://hub.docker.com/r/i386/perl/), `ppc64le` (https://hub.docker.com/r/ppc64le/perl/), `s390x` (https://hub.docker.com/r/s390x/perl/)

- **Published image artifact details**:
  repo-info repo's `repos/perl/` directory (https://github.com/docker-library/repo-info/blob/master/repos/perl) (history (https://github.com/docker-library/repo-info/commits/master/repos/perl))
  (image metadata, transfer size, etc)

- **Image updates**:
  official-images PRs with label `library/perl` (https://github.com/docker-library/official-images/pulls?q=label%3Alibrary%2Fperl)
  official-images repo's `library/perl` file (https://github.com/docker-library/official-images/blob/master/library/perl) (history (https://github.com/docker-library/official-images/commits/master/library/perl))

- **Source of this description**:
  docs repo's `perl/` directory (https://github.com/docker-library/docs/tree/master/perl) (history (https://github.com/docker-library/docs/commits/master/perl))

- **Supported Docker versions**:
  the latest release (https://github.com/docker/docker-ce/releases/latest) (down to 1.6 on a best-effort basis)

What is Perl?

Perl is a high-level, general-purpose, interpreted, dynamic programming language. The Perl language borrows features from other programming languages, including C, shell scripting (sh), AWK, and sed.

> wikipedia.org/wiki/Perl (https://en.wikipedia.org/wiki/Perl)

How to use this image

Create a `Dockerfile` in your Perl app project

```
FROM perl:5.20
COPY . /usr/src/myapp
WORKDIR /usr/src/myapp
CMD [ "perl", "./your-daemon-or-script.pl" ]
```

Then, build and run the Docker image:

```
$ docker build -t my-perl-app .
$ docker run -it --rm --name my-running-app my-perl-app
```

Run a single Perl script

For many simple, single file projects, you may find it inconvenient to write a complete `Dockerfile`. In such cases, you can run a Perl script by using the Perl Docker image directly:

```
$ docker run -it --rm --name my-running-script -v "$PWD":/usr/src/m
```

Example: Creating a reusable Carton image for Perl projects

Suppose you have a project that uses Carton (https://metacpan.org/pod/Carton) to manage Perl dependencies. You can create a `perl:carton` image that makes use of the ONBUILD (https://docs.docker.com/engine/reference/builder/#onbuild) instruction in its `Dockerfile`, like this:

```
FROM perl:5.26

RUN cpanm Carton \
    && mkdir -p /usr/src/app
WORKDIR /usr/src/app

ONBUILD COPY cpanfile* /usr/src/myapp
ONBUILD RUN carton install

ONBUILD COPY . /usr/src/app
```

Then, in your Carton project, you can now reduce your project's `Dockerfile` into a single line of `FROM perl:carton`, which may be enough to build a stand-alone image.

Having a single `perl:carton` base image is useful especially if you have multiple Carton-based projects in development, to avoid "boilerplate" coding of installing Carton and/or copying the project source files into the derived image. Keep in mind, though, about certain things to consider when using the Perl image in this way:

- This kind of base image will hide the useful bits (such as the `COPY` / `RUN` above) in the image, separating it from more specific Dockerfiles using the base image. This might lead to confusion when creating further derived images, so be aware of how ONBUILD triggers (https://docs.docker.com/engine/reference/builder/#onbuild) work and plan appropriately.
- There is the cost of maintaining an extra base image build, so if you're working on a single Carton project and/or plan to publish it, then it may be more preferable to derive directly from a versioned `perl` image instead.

Image Variants

The `perl` images come in many flavors, each designed for a specific use case.

`perl:<version>`

This is the defacto image. If you are unsure about what your needs are, you probably want to use this one. It is designed to be used both as a throw away container (mount your source code and start the container to start your app), as well as the base to build other images off of.

This tag is based off of `buildpack-deps` (https://hub.docker.com/_/buildpack-deps/). `buildpack-deps` is designed for the average user of Docker who has many images on their system. It, by design, has a large number of extremely common Debian packages. This reduces the number of packages that images that derive from it need to install, thus reducing the overall size of all images on your system.

`perl:<version>-slim`

This image does not contain the common packages contained in the default tag and only contains the minimal packages needed to run `perl`. Unless you are working in an environment where *only* the `perl` image will be deployed and you have space constraints, we highly recommend using the default image of this repository.

License

View license information (http://dev.perl.org/licenses/) 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 `perl/` directory (https://github.com/docker-library/repo-info/tree/master/repos/perl).

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.

Docker Pull Command

```
docker pull perl
```