(/)

OFFICIAL REPOSITORY

maven (/r/_/maven/) ☆

Last pushed: 4 hours ago

Repo Info (/_/maven/)

Short Description

Apache Maven is a software project management and comprehension tool.

Full Description

Supported tags and respective `Dockerfile` links

- `3.6.0-jdk-10-slim`, `3.6-jdk-10-slim`, `3-jdk-10-slim` (*jdk-10-slim/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/05f4802aa5c253dcf75fe967c6f45b3fb1e2f26e/jdk-10-slim/Dockerfile)
- `3.6.0-jdk-10`, `3.6-jdk-10`, `3-jdk-10` (*jdk-10/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/05f4802aa5c253dcf75fe967c6f45b3fb1e2f26e/jdk-10/Dockerfile)
- `3.6.0-jdk-11-slim`, `3.6-jdk-11-slim`, `3-jdk-11-slim` (*jdk-11-slim/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/05f4802aa5c253dcf75fe967c6f45b3fb1e2f26e/jdk-11-slim/Dockerfile)
- `3.6.0-jdk-11`, `3.6-jdk-11`, `3-jdk-11` (*jdk-11/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/05f4802aa5c253dcf75fe967c6f45b3fb1e2f26e/jdk-11/Dockerfile)
- `3.6.0-jdk-12-alpine`, `3.6-jdk-12-alpine`, `3-jdk-12-alpine` (*jdk-12-alpine/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/0607c2d00e31df3d2166bb0ab4f5097bdf2ede0e/jdk-12-alpine/Dockerfile)
- `3.6.0-jdk-12`, `3.6-jdk-12`, `3-jdk-12` (*jdk-12/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/0607c2d00e31df3d2166bb0ab4f5097bdf2ede0e/jdk-12/Dockerfile)
- `3.6.0-jdk-7-alpine`, `3.6-jdk-7-alpine`, `3-jdk-7-alpine` (*jdk-7-alpine/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/05f4802aa5c253dcf75fe967c6f45b3fb1e2f26e/jdk-7-alpine/Dockerfile)

- `3.6.0-jdk-7-slim`, `3.6-jdk-7-slim`, `3-jdk-7-slim` (*jdk-7-slim/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/05f4802aa5c253dcf75fe967c6f45b3fb1e2f26e/jdk-7-slim/Dockerfile)
- `3.6.0-jdk-7`, `3.6-jdk-7`, `3-jdk-7` (*jdk-7/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/05f4802aa5c253dcf75fe967c6f45b3fb1e2f26e/jdk-7/Dockerfile)
- `3.6.0-jdk-8-alpine`, `3.6.0-alpine`, `3.6-jdk-8-alpine`, `3.6-alpine`, `3-jdk-8-alpine`, `alpine` (*jdk-8-alpine/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/05f4802aa5c253dcf75fe967c6f45b3fb1e2f26e/jdk-8-alpine/Dockerfile)
- `3.6.0-jdk-8-slim`, `3.6.0-slim`, `3.6-jdk-8-slim`, `3.6-slim`, `3-jdk-8-slim`, `slim` (*jdk-8-slim/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/05f4802aa5c253dcf75fe967c6f45b3fb1e2f26e/jdk-8-slim/Dockerfile)
- `3.6.0-jdk-8`, `3.6.0`, `3.6-jdk-8`, `3.6`, `3-jdk-8`, `3`, `latest` (*jdk-8/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/05f4802aa5c253dcf75fe967c6f45b3fb1e2f26e/jdk-8/Dockerfile)
- `3.6.0-ibmjava-8-alpine`, `3.6.0-ibmjava-alpine`, `3.6-ibmjava-8-alpine`, `3.6-ibmjava-alpine`, `3-ibmjava-8-alpine`, `ibmjava-alpine` (*ibmjava-8-alpine/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/05f4802aa5c253dcf75fe967c6f45b3fb1e2f26e/ibmjava-8-alpine/Dockerfile)
- `3.6.0-ibmjava-8`, `3.6.0-ibmjava`, `3.6-ibmjava-8`, `3.6-ibmjava`, `3-ibmjava-8`, `3-ibmjava`, `ibmjava` (*ibmjava-8/Dockerfile*) (https://github.com/carlossg/docker-maven/blob/05f4802aa5c253dcf75fe967c6f45b3fb1e2f26e/ibmjava-8/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/carlossg/docker-maven/issues (https://github.com/carlossg/docker-maven/issues)

- **Maintained by**:
  the Maven Project (https://github.com/carlossg/docker-maven)

- **Supported architectures**: (more info (https://github.com/docker-library/official-images#architectures-other-than-amd64))
  `amd64` (https://hub.docker.com/r/amd64/maven/), `arm32v5`

(https://hub.docker.com/r/arm32v5/maven/), `arm32v6` (https://hub.docker.com/r/arm32v6/maven/), `arm32v7` (https://hub.docker.com/r/arm32v7/maven/), `arm64v8` (https://hub.docker.com/r/arm64v8/maven/), `i386` (https://hub.docker.com/r/i386/maven/), `ppc64le` (https://hub.docker.com/r/ppc64le/maven/), `s390x` (https://hub.docker.com/r/s390x/maven/)

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

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

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

- **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 Maven?

Apache Maven (http://maven.apache.org) is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.

How to use this image

You can run a Maven project by using the Maven Docker image directly, passing a Maven command to `docker run` :

```
$ docker run -it --rm --name my-maven-project -v "$(pwd)":/usr/src/
```

Building local Docker image (optional)

This is a base image that you can extend, so it has the bare minimum packages needed. If you add custom package(s) to the `Dockerfile` , then you can build your local Docker image like this:

```
$ docker build --tag my_local_maven:3.5.2-jdk-8 .
```

Reusing the Maven local repository

The local Maven repository can be reused across containers by creating a volume and mounting it in `/root/.m2` .

```
docker volume create --name maven-repo
docker run -it -v maven-repo:/root/.m2 maven mvn archetype:generate
docker run -it -v maven-repo:/root/.m2 maven mvn archetype:generate
```

Or you can just use your home .m2 cache directory that you share e.g. with your Eclipse/IDEA:

```
$ docker run -it --rm -v "$PWD":/usr/src/mymaven -v "$HOME/.m2":/r
```

Packaging a local repository with the image

The `$MAVEN_CONFIG` dir (default to `/root/.m2`) could be configured as a volume so anything copied there in a Dockerfile at build time is lost. For that reason the dir `/usr/share/maven/ref/` exists, and anything in that directory will be copied on container startup to `$MAVEN_CONFIG` .

To create a pre-packaged repository, create a `pom.xml` with the dependencies you need and use this in your `Dockerfile` . `/usr/share/maven/ref/settings-docker.xml` is a settings file that changes the local repository to `/usr/share/maven/ref/repository` , but you can use your own settings file as long as it uses `/usr/share/maven/ref/repository` as local repo.

```
COPY pom.xml /tmp/pom.xml
RUN mvn -B -f /tmp/pom.xml -s /usr/share/maven/ref/settings-docker.
```

To add your custom `settings.xml` file to the image use

```
COPY settings.xml /usr/share/maven/ref/
```

For an example, check the `tests` dir

Running as non-root

Maven needs the user home to download artifacts to, and if the user does not exist in the image an extra `user.home` Java property needs to be set.

For example, to run as user `1000` mounting the host' Maven repo

```
$ docker run -v ~/.m2:/var/maven/.m2 -ti --rm -u 1000 -e MAVEN_CONF
```

Image Variants

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

`maven:<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.

`maven:<version>-slim`

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

`maven:<version>-alpine`

This image is based on the popular Alpine Linux project (http://alpinelinux.org), available in the `alpine` official image (https://hub.docker.com/_/alpine). Alpine Linux is much smaller than most distribution base images (~5MB), and thus leads to much slimmer images in general.

This variant is highly recommended when final image size being as small as possible is desired. The main caveat to note is that it does use musl libc (http://www.musl-libc.org) instead of glibc and friends (http://www.etalabs.net/compare_libcs.html), so certain software might run into issues depending on the depth of their libc requirements. However, most software doesn't have an issue with this, so this variant is usually a very safe choice. See this Hacker News comment thread (https://news.ycombinator.com/item?id=10782897) for more discussion of the issues that might arise and some pro/con comparisons of using Alpine-based images.

To minimize image size, it's uncommon for additional related tools (such as `git` or `bash`) to be included in Alpine-based images. Using this image as a base, add the things you need in your own Dockerfile (see the `alpine` image description

(https://hub.docker.com/_/alpine/) for examples of how to install packages if you are unfamiliar).

License

View license information (https://www.apache.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 `maven/` directory (https://github.com/docker-library/repo-info/tree/master/repos/maven).

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 maven
```