(/)

OFFICIAL REPOSITORY

java (/r/_/java/) ☆

Last pushed: 2 years ago

Repo Info (/_/java/)

Short Description

Java is a concurrent, class-based, and object-oriented programming language.

Full Description

DEPRECATED

This image is officially deprecated in favor of the `openjdk` image (https://hub.docker.com/_/openjdk/), and will receive no further updates after 2016-12-31 (Dec 31, 2016). Please adjust your usage accordingly.

The image has been OpenJDK-specific since it was first introduced, and as of 2016-08-10 we also have an `ibmjava` image (https://hub.docker.com/_/ibmjava/), which made it even more clear that each repository should represent one upstream instead of one language stack or community, so this rename reflects that clarity appropriately.

Supported tags and respective `Dockerfile` links

- `6b38-jdk`, `6b38`, `6-jdk`, `6`, `openjdk-6b38-jdk`, `openjdk-6b38`, `openjdk-6-jdk`, `openjdk-6` (*6-jdk/Dockerfile*) (https://github.com/docker-library/openjdk/blob/89851f0abc3a83cfad5248102f379d6a0bd3951a/6-jdk/Dockerfile)
- `6b38-jre`, `6-jre`, `openjdk-6b38-jre`, `openjdk-6-jre` (*6-jre/Dockerfile*) (https://github.com/docker-library/openjdk/blob/89851f0abc3a83cfad5248102f379d6a0bd3951a/6-jre/Dockerfile)
- `7u111-jdk`, `7u111`, `7-jdk`, `7`, `openjdk-7u111-jdk`, `openjdk-7u111`, `openjdk-7-jdk`, `openjdk-7` (*7-jdk/Dockerfile*) (https://github.com/docker-library/openjdk/blob/054cea7585e6c0e4e98d133378ea38061a2ae3ac/7-jdk/Dockerfile)
- `7u121-jdk-alpine`, `7u121-alpine`, `7-jdk-alpine`, `7-alpine`, `openjdk-7u121-jdk-alpine`, `openjdk-7u121-alpine`, `openjdk-7-jdk-alpine`,

`openjdk-7-alpine` (*7-jdk/alpine/Dockerfile*) (https://github.com/docker-library/openjdk/blob/5257acb51a1230a2dc46b1c349d674a725562f9d/7-jdk/alpine/Dockerfile)

- `7u111-jre`, `7-jre`, `openjdk-7u111-jre`, `openjdk-7-jre` (*7-jre/Dockerfile*) (https://github.com/docker-library/openjdk/blob/054cea7585e6c0e4e98d133378ea38061a2ae3ac/7-jre/Dockerfile)
- `7u121-jre-alpine`, `7-jre-alpine`, `openjdk-7u121-jre-alpine`, `openjdk-7-jre-alpine` (*7-jre/alpine/Dockerfile*) (https://github.com/docker-library/openjdk/blob/5257acb51a1230a2dc46b1c349d674a725562f9d/7-jre/alpine/Dockerfile)
- `8u111-jdk`, `8u111`, `8-jdk`, `8`, `jdk`, `latest`, `openjdk-8u111-jdk`, `openjdk-8u111`, `openjdk-8-jdk`, `openjdk-8` (*8-jdk/Dockerfile*) (https://github.com/docker-library/openjdk/blob/e6e9cf8b21516ba764189916d35be57486203c95/8-jdk/Dockerfile)
- `8u111-jdk-alpine`, `8u111-alpine`, `8-jdk-alpine`, `8-alpine`, `jdk-alpine`, `alpine`, `openjdk-8u111-jdk-alpine`, `openjdk-8u111-alpine`, `openjdk-8-jdk-alpine`, `openjdk-8-alpine` (*8-jdk/alpine/Dockerfile*) (https://github.com/docker-library/openjdk/blob/9a0822673dffd3e5ba66f18a8547aa60faed6d08/8-jdk/alpine/Dockerfile)
- `8u111-jre`, `8-jre`, `jre`, `openjdk-8u111-jre`, `openjdk-8-jre` (*8-jre/Dockerfile*) (https://github.com/docker-library/openjdk/blob/e6e9cf8b21516ba764189916d35be57486203c95/8-jre/Dockerfile)
- `8u111-jre-alpine`, `8-jre-alpine`, `jre-alpine`, `openjdk-8u111-jre-alpine`, `openjdk-8-jre-alpine` (*8-jre/alpine/Dockerfile*) (https://github.com/docker-library/openjdk/blob/9a0822673dffd3e5ba66f18a8547aa60faed6d08/8-jre/alpine/Dockerfile)
- `9-b149-jdk`, `9-b149`, `9-jdk`, `9`, `openjdk-9-b149-jdk`, `openjdk-9-b149`, `openjdk-9-jdk`, `openjdk-9` (*9-jdk/Dockerfile*) (https://github.com/docker-library/openjdk/blob/e2c8648d39ef1492df3482de3fda0ee3f8955fb1/9-jdk/Dockerfile)
- `9-b149-jre`, `9-jre`, `openjdk-9-b149-jre`, `openjdk-9-jre` (*9-jre/Dockerfile*) (https://github.com/docker-library/openjdk/blob/e2c8648d39ef1492df3482de3fda0ee3f8955fb1/9-jre/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/docker-library/java/issues (https://github.com/docker-library/java/issues)

- **Maintained by**:
  the Docker Community (https://github.com/docker-library/java)

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

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

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

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

What is Java?

Java is a concurrent, class-based, object-oriented language specifically designed to have as few implementation dependencies as possible. It is intended to allow application developers to "write once, run anywhere", meaning that code that runs on one platform does not need to be recompiled to run on another.

Java is a registered trademark of Oracle and/or its affiliates.

> wikipedia.org/wiki/Java_(programming_language) (http://en.wikipedia.org/wiki/Java_%28programming_language%29)

How to use this image

Start a Java instance in your app

The most straightforward way to use this image is to use a Java container as both the build and runtime environment. In your `Dockerfile`, writing something along the lines of the following will compile and run your project:

```
FROM java:7
COPY . /usr/src/myapp
WORKDIR /usr/src/myapp
RUN javac Main.java
CMD ["java", "Main"]
```

You can then run and build the Docker image:

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

Compile your app inside the Docker container

There may be occasions where it is not appropriate to run your app inside a container. To compile, but not run your app inside the Docker instance, you can write something like:

```
$ docker run --rm -v "$PWD":/usr/src/myapp -w /usr/src/myapp java:7
```

This will add your current directory as a volume to the container, set the working directory to the volume, and run the command `javac Main.java` which will tell Java to compile the code in `Main.java` and output the Java class file to `Main.class`.

Why is this only OpenJDK/OpenJRE?

As all of the major upstream Linux distributions are unwilling to redistribute Oracle Java in their own distribution channels, we have chosen to follow them. See references below on how each distribution does not distribute Oracle Java.

- Ubuntu stopped distributing it in the `sun-java6` package when Oracle retired the "Operating System Distributor License for Java" (lists.ubuntu.com

(https://lists.ubuntu.com/archives/ubuntu-security-announce/2011-December/001528.html)).

- Debian requires users to download the Java tar manually from oracle.com and then use `java-package` to install it (wiki.debian.net (https://wiki.debian.org/Java/Sun)).
- The webupd8 PPA for Ubuntu and Debian requires the user to accept the Oracle license in order for their software to download and install Oracle java (webupd8.org (http://www.webupd8.org/2012/09/install-oracle-java-8-in-ubuntu-via-ppa.html)).
- Gentoo has a *fetch-restriction* that requires the user to go to the Oracle website to download the Java tar manually which inclues accepting the license (wiki.gentoo.org (https://wiki.gentoo.org/wiki/Java)).
- CentOS requires users to go and download the rpm provided by Oracle at java.com and thus accept the Oracle license (wiki.centos.org (https://wiki.centos.org/HowTos/JavaRuntimeEnvironment)).
- RedHat provides instructions to add a repo that is maintained by Oracle (access.redhat.com (https://access.redhat.com/solutions/732883)).

Image Variants

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

`java:<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://registry.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.

`java: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 (http://openjdk.java.net/legal/gplv2+ce.html) for the software contained in this image.

Docker Pull Command

```
docker pull java
```