Red Hat build of OpenJDK 17

Installing and using Red Hat build of OpenJDK 17 on RHEL

Last Updated: 2024-10-29

Legal Notice

Abstract

Red Hat build of OpenJDK is a Red Hat offering on the Red Hat Enterprise Linux platform. The Installing and using Red Hat build of OpenJDK 17 guide provides an overview of this product and explains how to install the software and start using it.

Table of Contents

PROVIDING FEEDBACK ON RED HAT BUILD OF OPENJDK DOCUMENTATION

To report an error or to improve our documentation, log in to your Red Hat Jira account and submit an issue. If you do not have a Red Hat Jira account, then you will be prompted to create an account.

Procedure

1. Click the following link to **create a ticket**.
2. Enter a brief description of the issue in the **Summary**.
3. Provide a detailed description of the issue or enhancement in the **Description**. Include a URL to where the issue occurs in the documentation.
4. Clicking **Create** creates and routes the issue to the appropriate documentation team.

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message.

CHAPTER 1. RED HAT BUILD OF OPENJDK 17 OVERVIEW

OpenJDK (Open Java Development Kit) is a free and open source implementation of the Java Platform, Standard Edition (Java SE). The Red Hat build of OpenJDK is available in four versions: 8u, 11u, 17u, and 21u.

Packages for the Red Hat build of OpenJDK are made available on Red Hat Enterprise Linux and Microsoft Windows and shipped as a JDK and JRE in the Red Hat Ecosystem Catalog.

CHAPTER 2. INSTALLING RED HAT BUILD OF OPENJDK 17 ON RED HAT ENTERPRISE LINUX

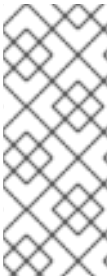
Red Hat build of OpenJDK is an environment for developing and running a wide range of platform-agnostic applications, from mobile applications to desktop and web applications and enterprise systems. Red Hat provides an open source implementation of the Java Platform SE (Standard Edition) called Red Hat build of OpenJDK.

Applications are developed using the JDK (Java Development Kit). Applications are run on a JVM (Java Virtual Machine), which is included in the JRE (Java Runtime Environment) and the JDK. There is also a headless version of Java which has the smallest footprint and does not include the libraries needed for a user interface. The headless version is packaged in the headless subpackage.

NOTE

If you are unsure whether you need the JRE or the JDK, it is recommended that you install the JDK.

The following sections provide instructions for installing Red Hat build of OpenJDK on Red Hat Enterprise Linux.



NOTE

You can install multiple major versions of Red Hat build of OpenJDK on your local system. If you need to switch from one major version to another major version, issue the following command in your command-line interface (CLI) and then follow the onscreen prompts:

```
$ sudo update-alternatives --config 'java'
```

2.1. INSTALLING A JRE ON RHEL BY USING YUM

You can install Red Hat build of OpenJDK Java Runtime Environment (JRE) by using the **yum** system package manager.

Prerequisites

- Logged in as a user with root privileges on the system.
- Registered your local system to your Red Hat Subscription Manager account. See Using Red Hat Subscription Manager in the *Getting Started with RHEL System Registration* guide.

Procedure

1. Run the **yum** command, specifying the package you want to install:

```
$ sudo yum install java-17-openjdk
```

2. Check that the installation works:

```
$ java -version

openjdk version "17.0.2" 2022-01-18 LTS
```

```
OpenJDK Runtime Environment 21.9 (build 17.0.2+8-LTS)
OpenJDK 64-Bit Server VM 21.9 (build 17.0.2+8-LTS, mixed mode, sharing)
```

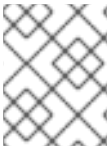
NOTE

If the output from the previous command shows that you have a different major version of Red Hat build of OpenJDK checked out on your system, you can enter the following command in your CLI to switch your system to use Red Hat build of OpenJDK 17:

```
$ sudo update-alternatives --config 'java'
```

2.2. INSTALLING A JRE ON RHEL BY USING AN ARCHIVE

You can install Red Hat build of OpenJDK Java Runtime Environment (JRE) by using an archive file. This is useful if the Java administrator does not have root privileges.



NOTE

To ease the upgrades for later versions create a parent directory to contain your JREs and create a symbolic link to the latest JRE using a generic path.

Procedure

1. Create a directory to where you want to download the archive file, and then navigate to that directory on your command-line interface (CLI). For example:

```
$ mkdir ~/jres

$ cd ~/jres
```

2. Navigate to the Software Downloads page on the Red Hat Customer Portal.
3. Select the latest version of Red Hat build of OpenJDK 17 from the **Version** drop-down list, and then download the JRE archive for Linux to your local system.
4. Extract the contents of the archive to a directory of your choice:

```
$ tar -xf java-17-openjdk-17.0.2.0.8-3.portable.jre.el7.x86_64.tar.xz -C ~/jres
```

5. Create a generic path by using symbolic links to your JRE for easier upgrades:

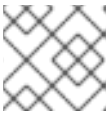
```
$ ln -s ~/jres/java-17-openjdk-17.0.2.0.8-3.portable.jdk.el7.x86_64 ~/jres/java-17
```

6. Configure the **JAVA_HOME** environment variable:

```
$ export JAVA_HOME=~/jres/java-17
```

7. Verify that **JAVA_HOME** environment variable is set correctly:

```
$ printenv | grep JAVA_HOME
JAVA_HOME=~/jres/java-17
```



NOTE

When installed using this method, Java will only be available for the current user.

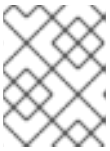
8. Add the **bin** directory of the generic JRE path to the **PATH** environment variable:

```
$ export PATH="$JAVA_HOME/bin:$PATH"
```

9. Verify that **java -version** works without supplying the full path:

```
$ java -version

openjdk version "17.0.2" 2022-01-18 LTS
OpenJDK Runtime Environment 21.9 (build 17.0.2+8-LTS)
OpenJDK 64-Bit Server VM 21.9 (build 17.0.2+8-LTS, mixed mode, sharing)
```



NOTE

You can ensure that **JAVA_HOME** environment variable persists for the current user by exporting the environment variable in ~/**.bashrc**.

2.3. INSTALLING RED HAT BUILD OF OPENJDK ON RHEL BY USING YUM

You can install Red Hat build of OpenJDK by using the **yum** system package manager.

Prerequisites

- Logged in as a user with root privileges.
- Registered your local system to your Red Hat Subscription Manager account. See Using the Red Hat Subscription Manager in the *Getting Started with RHEL System Registration* guide.

Procedure

1. Run the **yum** command, specifying the package you want to install:

```
$ sudo yum install java-17-openjdk-devel
```

2. Check that the installation works:

```
$ javac -version

javac 17.0.2
```

2.4. INSTALLING RED HAT BUILD OF OPENJDK ON RHEL BY USING AN ARCHIVE

You can install Red Hat build of OpenJDK with an archive. This is useful if the Java administrator does not have root privileges.

NOTE

To ease upgrades, create a parent directory to contain your JREs and create a symbolic link to the latest JRE using a generic path.

Procedure

1. Create a directory to where you want to download the archive file, and then navigate to that directory on your command-line interface (CLI). For example:

```
$ mkdir ~/jdks

$ cd ~/jdks
```

2. Navigate to the Software Downloads page on the Red Hat Customer Portal.
3. Select the latest version of Red Hat build of OpenJDK 17 from the **Version** drop-down list, and then download the JDK archive for Linux to your local system.
4. Extract the contents of the archive to a directory of your choice:

```
$ tar -xf java-17-openjdk-17.0.2.0.8-3.portable.jre.el7.x86_64.tar.xz -C ~/jdks
```

5. Create a generic path by using symbolic links to your JDK for easier upgrades:

```
$ ln -s ~/jdks/java-17-openjdk-17.0.2.0.8-3.portable.jdk.el7.x86_64 ~/jdks/java-17
```

6. Configure the **JAVA_HOME** environment variable:

```
$ export JAVA_HOME=~/jdks/java-17
```

7. Verify that **JAVA_HOME** environment variable is set correctly:

```
$ printenv | grep JAVA_HOME
JAVA_HOME=~/jdks/java-17
```

NOTE

When installed using this method, Java will only be available for the current user.

8. Add the **bin** directory of the generic JRE path to the **PATH** environment variable:

```
$ export PATH="$JAVA_HOME/bin:$PATH"
```

9. Verify that **java -version** works without supplying the full path:

```
$ java -version

openjdk version "17.0.2" 2022-01-18 LTS
OpenJDK Runtime Environment 21.9 (build 17.0.2+8-LTS)
OpenJDK 64-Bit Server VM 21.9 (build 17.0.2+8-LTS, mixed mode, sharing)
```

NOTE

You can ensure that **JAVA_HOME** environment variable persists for the current user by exporting the environment variable in ~/**.bashrc**.

2.5. INSTALLING MULTIPLE MAJOR VERSIONS OF RED HAT BUILD OF OPENJDK ON RHEL BY USING YUM

You can install multiple versions of Red Hat build of OpenJDK by using the **yum** system package manager.

Prerequisites

- A Red Hat Subscription Manager (RHSM) account with an active subscription that provides access to a repository that provides the Red Hat build of OpenJDK you want to install.
- You must have root privileges on the system.

Procedure

1. Run the following **yum** commands to install the package:
   For Red Hat build of OpenJDK 17

   ```
   $ sudo yum install java-17-openjdk
   ```

   For Red Hat build of OpenJDK 11

   ```
   $ sudo yum install java-11-openjdk
   ```

   For Red Hat build of OpenJDK 8

   ```
   $ sudo yum install java-1.8.0-openjdk
   ```

2. After installing, check the available Java versions:

   ```
   $ sudo yum list installed "java*"

   Installed Packages

   java-1.8.0-openjdk.x86_64    1:1.8.0.322.b06-2.el8_5    @rhel-8-for-x86_64-appstream-rpms
   java-11-openjdk.x86_64    1:11.0.14.0.9-2.el8_5    @rhel-8-for-x86_64-appstream-rpms
   java-17-openjdk.x86_64    1:17.0.2.0.8-4.el8_5    @rhel-8-for-x86_64-appstream-rpms
   ```

3. Check the current java version:

   ```
   $ java -version

   openjdk version "17.0.2" 2022-01-18 LTS
   OpenJDK Runtime Environment 21.9 (build 17.0.2+8-LTS)
   OpenJDK 64-Bit Server VM 21.9 (build 17.0.2+8-LTS, mixed mode, sharing)
   ```

NOTE

You can install multiple major versions of Red Hat build of OpenJDK on your local system. If you need to switch from one major version to another major version, issue the following command in your command-line interface (CLI) and then follow the onscreen prompts:

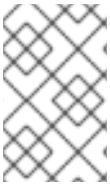
```
$ sudo update-alternatives --config 'java'
```

Additional resources

- You can configure the default Java version to use by using **java --alternatives**. For more information, see Non-interactively selecting a system-wide Red Hat build of OpenJDK version on RHEL.

2.6. INSTALLING MULTIPLE MAJOR VERSIONS OF RED HAT BUILD OF OPENJDK ON RHEL BY USING AN ARCHIVE

You can install multiple major versions of Red Hat build of OpenJDK by using the same procedures found in *Installing a JRE on RHEL by using an archive* or *Installing Red Hat build of OpenJDK on RHEL by using an archive using multiple major versions*.



NOTE

For instructions how to configure the default Red Hat build of OpenJDK version for the system, see Non-interactively selecting a system-wide Red Hat build of OpenJDK version on RHEL.

Additional resources

- For instructions on installing a JRE, see Installing a JRE on RHEL by using an archive .
- For instructions on installing a JDK, see Installing Red Hat build of OpenJDK on RHEL by using an archive.

2.7. INSTALLING MULTIPLE MINOR VERSIONS OF RED HAT BUILD OF OPENJDK ON RHEL BY USING YUM

You can install multiple minor versions of Red Hat build of OpenJDK on RHEL. This is done by preventing the installed minor versions from being updated.

Prerequisites

- Choose system-wide version of Red Hat build of OpenJDK from Non-interactively selecting a system-wide Red Hat build of OpenJDK version on RHEL.

Procedure

1. Add the **installonlypkgs** option in the **/etc/yum.conf** directory to specify the Red Hat build of OpenJDK packages that **yum** can install but not update.

```
installonlypkgs=java-<version>-openjdk,java-<version>-openjdk-headless,java-<version>-openjdk-devel
```

Updates will install new packages while leaving the old versions on the system.

```
$ rpm -qa | grep java-17.0.2-openjdk

java-17-openjdk-17.0.1.0.12-2.el8_5.x86_64
java-17-openjdk-17.0.2.0.8-4.el8_5.x86_64
```

2. The different minor versions of Red Hat build of OpenJDK can be found in the **/usr/lib/jvm/*<minor version>*** files.
For example, the following shows part of **/usr/lib/jvm/java-17.0.2-openjdk**:

```
$ /usr/lib/jvm/java-17-openjdk-17.0.2.0.8-4.el8_5.x86_64/bin/java -version
openjdk version "17.0.2" 2022-01-18 LTS
OpenJDK Runtime Environment 21.9 (build 17.0.2+8-LTS)
OpenJDK 64-Bit Server VM 21.9 (build 17.0.2+8-LTS, mixed mode, sharing)

$ /usr/lib/jvm/java-17-openjdk-17.0.1.0.12-2.el8_5.x86_64/bin/java -version
openjdk version "17" 2021-10-19
OpenJDK Runtime Environment 21.9 (build 17+35)
OpenJDK 64-Bit Server VM 21.9 (build 17+35, mixed mode, sharing)
```

2.8. INSTALLING MULTIPLE MINOR VERSIONS OF RED HAT BUILD OF OPENJDK ON RHEL BY USING AN ARCHIVE

Installing multiple minor versions is the same as Installing a JRE on RHEL using an archive or Installing Red Hat build of OpenJDK on RHEL 8 using an archive using multiple minor versions.

NOTE

For instructions how to choose a default minor version for the system, see Non-interactively selecting a system-wide Red Hat build of OpenJDK version on RHEL.

Additional resources

- For instructions on installing a JRE, see Installing a JRE on RHEL by using an archive .
- For instructions on installing a JDK, see Installing Red Hat build of OpenJDK on RHEL by using an archive.

CHAPTER 3. DEBUG SYMBOLS FOR RED HAT BUILD OF OPENJDK 17

Debug symbols help in investigating a crash in Red Hat build of OpenJDK applications.

3.1. INSTALLING THE DEBUG SYMBOLS

This procedure describes how to install the debug symbols for Red Hat build of OpenJDK.

Prerequisites

- Installed the **gdb** package on your local sytem.
  - You can issue the **sudo yum install gdb** command on your CLI to install this package on your local system.

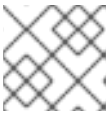
Procedure

1. To install the debug symbols, enter the following command:

```
$ sudo debuginfo-install java-17-openjdk

$ sudo debuginfo-install java-17-openjdk-headless
```

   These commands install **java-17-openjdk-debuginfo**, **java-17-openjdk-headless-debuginfo**, and additional packages that provide debug symbols for Red Hat build of OpenJDK 17 binaries. These packages are not self-sufficient and *do not* contain executable binaries.



NOTE

The **debuginfo-install** is provided by the **yum-utils** package.

2. To verify that the debug symbols are installed, enter the following command:

```
$ gdb which java

Reading symbols from /usr/bin/java...Reading symbols from /usr/lib/debug/usr/lib/jvm/java-17-openjdk-17.0.2.0.8-4.el8_5/bin/java-17.0.2.0.8-4.el8_5.x86_64.debug...done.
(gdb)
```

3.2. CHECKING THE INSTALLATION LOCATION OF DEBUG SYMBOLS

This procedure explains how to find the location of debug symbols.

NOTE

If the **debuginfo** package is installed, but you cannot get the installation location of the package, then check if the correct package and java versions are installed. After confirming the versions, check the location of debug symbols again.

Prerequisites

- Installed the **gdb** package on your local sytem.
  - You can issue the **sudo yum install gdb** command on your CLI to install this package on your local system.
  - Installed the debug symbols package. See Installing the debug symbols.

Procedure

1. To find the location of debug symbols, use **gdb** with **which java** commands:

```
$ gdb which java

Reading symbols from /usr/bin/java...Reading symbols from /usr/lib/debug/usr/lib/jvm/java-17-openjdk-17.0.2.0.8-4.el8_5/bin/java-17.0.2.0.8-4.el8_5.x86_64.debug...done.
(gdb)
```

2. Use the following commands to explore the ***-debug** directory to see all the debug versions of the libraries, which include **java**, **javac**, and **javah**:

```
$ cd /usr/lib/debug/lib/jvm/java-17-openjdk-17.0.2.0.8-4.el8_5
```

```
$ tree

OJDK 17 version:
└── java-17-openjdk-17.0.2.0.8-4.el8_5
├── bin
│   ...
│ │── java-java-17.0.2.0.8-4.el8_5.x86_64.debug
│   ├── javac-java-17.0.2.0.8-4.el8_5.x86_64.debug
│   ├── javadoc-java-17.0.2.0.8-4.el8_5.x86_64.debug
│   ...
└── lib
├── jexec-java-17.0.2.0.8-4.el8_5.x86_64.debug
├── jli
│   └── libjli.so-java-17.0.2.0.8-4.el8_5.x86_64.debug
├── jspawnhelper-java-17.0.2.0.8-4.el8_5.x86_64.debug
│   ...
```

NOTE

The **javac** and **javah** tools are provided by the **java-17-openjdk-devel** package. You can install the package using the command: **$ sudo debuginfo-install java-17-openjdk-devel**.

3.3. CHECKING THE CONFIGURATION OF DEBUG SYMBOLS

You can check and set configurations for debug symbols.

- Enter the following command to get a list of the installed packages:

```
$ sudo yum list installed | grep 'java-17-openjdk-debuginfo'
```

- If some debug information packages have not been installed, enter the following command to install the missing packages:

```
$ sudo yum debuginfo-install glibc-2.28-151.el8.x86_64 libgcc-8.4.1-1.el8.x86_64 libstdc++-8.4.1-1.el8.x86_64 sssd-client-2.4.0-9.el8.x86_64 zlib-1.2.11-17.el8.x86_64
```

- Run the following command if you want to hit a specific breakpoint:

```
$ gdb -ex 'handle SIGSEGV noprint nostop pass' -ex 'set breakpoint pending on' -ex 'break JavaCalls::call' -ex 'run' --args java ./HelloWorld
```

The above command completes the following tasks:

  - Handles the SIGSEGV error as the JVM uses SEGV for stack overflow check.
  - Sets pending breakpoints to **yes**.
  - Calls the break statement in **JavaCalls::call** function. The function to starts the application in HotSpot (libjvm.so).

3.4. CONFIGURING THE DEBUG SYMBOLS IN A FATAL ERROR LOG FILE

When a Java application is down due to a JVM crash, a fatal error log file is generated, for example: **hs_error**, **java_error**. These error log files are generated in current working directory of the application. The crash file contains information from the stack.

Procedure

1. You can remove all the debug symbols by using the **strip -g** command.
   The following code shows an example of non-stripped **hs_error** file:

```
Native frames: (J=compiled Java code, j=interpreted, Vv=VM code, C=native code)
V  [libjvm.so+0xb83d2a]  Unsafe_SetLong+0xda
j  sun.misc.Unsafe.putLong(Ljava/lang/Object;JJ)V+0
j  Crash.main([Ljava/lang/String;)V+8
v  ~StubRoutines::call_stub
V  [libjvm.so+0x6c0e65]  JavaCalls::call_helper(JavaValue*, methodHandle*, JavaCallArguments*, Thread*)+0xc85
V  [libjvm.so+0x73cc0d]  jni_invoke_static(JNIEnv_*, JavaValue*, _jobject*, JNICallType, _jmethodID*, JNI_ArgumentPusher*, Thread*) [clone .constprop.1]+0x31d
V  [libjvm.so+0x73fd16]  jni_CallStaticVoidMethod+0x186
C  [libjli.so+0x48a2]  JavaMain+0x472
C  [libpthread.so.0+0x9432]  start_thread+0xe2
```

   The following code shows an example of stripped **hs_error** file:

```
Stack: [0x00007ff7e1a44000,0x00007ff7e1b44000],  sp=0x00007ff7e1b42850,  free space=1018k
Native frames: (J=compiled Java code, j=interpreted, Vv=VM code, C=native code)
V  [libjvm.so+0xa7ecab]
j  sun.misc.Unsafe.putAddress(JJ)V+0
j  Crash.crash()V+5
j  Crash.main([Ljava/lang/String;)V+0
```

```
v  ~StubRoutines::call_stub
V  [libjvm.so+0x67133a]
V  [libjvm.so+0x682bca]
V  [libjvm.so+0x6968b6]
C  [libjli.so+0x3989]
C  [libpthread.so.0+0x7dd5]  start_thread+0xc5
```

2. Enter the following command to check that you have the same version of debug symbols and the fatal error log file:

```
$ java -version
```

NOTE

You can also use the **sudo update-alternatives --config 'java'** to complete this check.

3. Use the **nm** command to ensure that **libjvm.so** has ELF data and text symbols:

```
$ nm /usr/lib/debug/usr/lib/jvm/java-17-openjdk-17.0.2.0.8-4.el8_5/lib/server/libjvm.so-17.0.2.0.8-4.el8_5.x86_64.debug
```

Additional resources

- The crash file **hs_error** is incomplete without the debug symbols installed. For more information, see Java application down due to JVM crash .

CHAPTER 4. UPDATING RED HAT BUILD OF OPENJDK 17 ON RED HAT ENTERPRISE LINUX

The following sections provide instructions for updating Red Hat build of OpenJDK 17 on Red Hat Enterprise Linux.

4.1. UPDATING RED HAT BUILD OF OPENJDK 17 ON RHEL BY USING YUM

You can update the installed Red Hat build of OpenJDK packages by using the **yum** system package manager.

Prerequisites

- You must have root privileges on the system.

Procedure

1. Check the current Red Hat build of OpenJDK version:

```
$ sudo yum list installed "java*"
```

   A list of installed Red Hat build of OpenJDK packages displays.

```
Installed Packages

java-1.8.0-openjdk.x86_64    1:1.8.0.322.b06-2.el8_5    @rhel-8-for-x86_64-appstream-rpms
java-11-openjdk.x86_64    1:11.0.14.0.9-2.el8_5    @rhel-8-for-x86_64-appstream-rpms
java-17-openjdk.x86_64    1:17.0.2.0.8-4.el8_5    @rhel-8-for-x86_64-appstream-rpms
```

2. Update a specific package. For example:

```
$ sudo yum update java-17-openjdk
```

3. Verify that the update worked by checking the current Red Hat build of OpenJDK versions:

```
$ java -version

openjdk version "17.0.2" 2022-01-18 LTS
OpenJDK Runtime Environment 21.9 (build 17.0.2+8-LTS)
OpenJDK 64-Bit Server VM 21.9 (build 17.0.2+8-LTS, mixed mode, sharing)
```

NOTE

You can install multiple major versions of Red Hat build of OpenJDK on your local system. If you need to switch from one major version to another major version, issue the following command in your command-line interface (CLI) and then follow the onscreen prompts:

```
$ sudo update-alternatives --config 'java'
```

4.2. UPDATING RED HAT BUILD OF OPENJDK 17 ON RHEL BY USING AN ARCHIVE

You can update Red Hat build of OpenJDK by using an archive file. This is useful if the Red Hat build of OpenJDK administrator does not have root privileges.

Prerequisites

- Know the generic path pointing to your JDK or JRE installation. For example, **~/jdks/java-17**

Procedure

1. Remove the existing symbolic link of the generic path to your JDK or JRE.
   For example:

   ```
   $ unlink ~/jdks/java-17
   ```

2. Install the latest version of the JDK or JRE in your installation location.

Additional resources

- For instructions on installing a JRE, see Installing a JRE on RHEL by using an archive .
- For instructions on installing a JDK, see Installing Red Hat build of OpenJDK on RHEL by using an archive.

Revised on 2024-10-29 18:38:39 UTC