OpenJDK 17

RHEL での OpenJDK 17 のインストールおよび使用

ガイド

Last Updated: 2022-04-19

OpenJDK 17 RHEL での OpenJDK 17 のインストールおよび使用

ガイド

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律上の通知

概要

OpenJDK は、Red Hat Enterprise Linux プラットフォーム上の Red Hat 製品です。『OpenJDK 17 のインストールと使用』では、この製品の概要と、ソフトウェアをインストールして使用を開始する方法を説明します。

目次

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、弊社の CTO、Chris Wright のメッセージ を参照してください。

RED HAT ドキュメントへのフィードバック

弊社のドキュメントに関するご意見やご感想をお寄せください。フィードバックをお寄せいただくには、ドキュメントのテキストを強調表示し、コメントを追加できます。

本セクションでは、フィードバックの送信方法を説明します。

前提条件

- Red Hat カスタマーポータルにログインしている。
- Red Hat カスタマーポータルで、**マルチページ HTML** 形式でドキュメントを表示します。

手順

フィードバックを提供するには、以下の手順を実施します。

1. ドキュメントの右上隅にある **フィードバック** ボタンをクリックして、既存のフィードバックを確認します。

注記

フィードバック機能は、**マルチページ HTML** 形式でのみ有効です。

2. フィードバックを提供するドキュメントのセクションを強調表示します。
3. ハイライトされたテキスト近くに表示される **Add Feedback** ポップアップをクリックします。ページの右側のフィードバックセクションにテキストボックスが表示されます。
4. テキストボックスにフィードバックを入力し、**Submit** をクリックします。ドキュメントに関する問題が作成されます。
5. 問題を表示するには、フィードバックビューで問題トラッカーリンクをクリックします。

第1章 OPENJDK 17 の概要

OpenJDK (Open Java Development Kit) は、Java Platform Standard Edition (Java SE) のオープンソース実装です。OpenJDK の Red Hat ビルドは、OpenJDK 8u、OpenJDK 11u と OpenJDK 17u の 3 つのバージョンで利用できます。

Red Hat ビルドの OpenJDK 向けパッケージは、Red Hat Enterprise Linux および Microsoft Windows で利用でき、Red Hat Ecosystem Catalog の JDK および JRE として同梱されています。

第2章 RED HAT ENTERPRISE LINUX での OPENJDK 17 のインストール

OpenJDK は、モバイルアプリケーションからデスクトップアプリケーション、Web アプリケーション、エンタープライズシステムまで、プラットフォームに依存しない幅広いアプリケーションを開発および実行するための環境です。Red Hat は、OpenJDK と呼ばれる Java Platform SE(Standard Edition) のオープンソース実装を提供します。

アプリケーションは、JDK (Java Development Kit) を使用して開発されます。アプリケーションは、JRE (Java ランタイム環境) および JDK に含まれている JVM (Java 仮想マシン) で実行されます。フットプリントが最小で、ユーザーインターフェースに必要なライブラリーが含まれていないヘッドレスバージョンの Java もあります。ヘッドレスバージョンは、ヘッドレスサブパッケージにパッケージ化されています。

注記

JRE と JDK のどちらが必要かわからない場合は、JDK をインストールすることが推奨されます。

以下のセクションでは、Red Hat Enterprise Linux に OpenJDK をインストールする手順を説明します。

注記

OpenJDK の複数のメジャーバージョンをローカルシステムにインストールできます。あるメジャーバージョンから別のメジャーバージョンに切り替える必要がある場合は、コマンドラインインターフェース(CLI)で以下のコマンドを実行し、画面のプロンプトに従います。

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

2.1. YUM を使用して RHEL に JRE をインストール

システムパッケージマネージャー (**yum**) を使用して、OpenJDK Java Runtime Environment (JRE) をインストールできます。

前提条件

- root 権限があるユーザーとしてシステムにログインしている。
- ローカルシステムを Red Hat Subscription Manager アカウントに登録している。Registering a system using Red Hat Subscription Managerユーザーガイドを参照してください。

手順

1. インストールするパッケージを指定して、**yum** コマンドを実行します。

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

2. インストールが機能することを確認します。

```
$ 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)
```

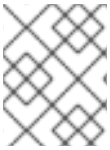
注記

直前のコマンドの出力で、システムで OpenJDK の別のメジャーバージョンがチェックアウトされていることが分かった場合には、CLI で以下のコマンドを入力して、システムを OpenJDK 17 を使用するように切り替えることができます。

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

2.2. アーカイブを使用した RHEL への JRE のインストール

アーカイブを使用して OpenJDK Java Runtime Environment (JRE) をインストールできます。これは、Java 管理者が root 権限を持たない場合に便利です。



注記

後続バージョンのアップグレードを容易にするために、JRE を含む親ディレクトリーを作成し、汎用パスを使用して最新の JRE へのシンボリックリンクを作成します。

手順

1. アーカイブファイルをダウンロードするディレクトリーを作成し、コマンドラインインターフェース(CLI)でそのディレクトリーに移動します。以下に例を示します。

```
$ mkdir ~/jres

$ cd ~/jres
```

2. Red Hat カスタマーポータルの Software Downloads ページに移動します。
3. **Version** ドロップダウンリストから OpenJDK 11 の最新バージョンを選択し、Linux 用の JRE アーカイブをローカルシステムにダウンロードします。
4. アーカイブのコンテンツを任意のディレクトリーに展開します。

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

5. アップグレードを容易にするために、JRE へのシンボリックリンクを使用して汎用パスを作成します。

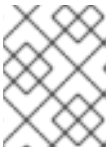
```
$ ln -s ~/jres/java-17-openjdk-17.0.2.0.8-3.portable.jdk.el7.x86_64 ~/jres/java-17
```

6. **JAVA_HOME** 環境変数を設定します。

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

7. **JAVA_HOME** 環境変数が正しく設定されていることを確認します。

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



注記

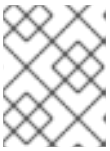
この方法でインストールした場合、Java は現在のユーザーのみが使用できます。

8. 一般的な JRE パスの **bin** ディレクトリーを **PATH** 環境変数に追加します。

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

9. フルパスを指定せずに **java -version** が機能することを確認します。

```
$ 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)
```



注記

~/**.bashrc** に環境変数をエクスポートすることで、**JAVA_HOME** 環境変数が現在のユーザーに対して持続することを確認できます。

2.3. YUM を使用した RHEL への OPENJDK のインストール

OpenJDK は、システムパッケージマネージャー **yum** を使用してインストールできます。

要件

- root 権限を持つユーザーとしてログインします。
- ローカルシステムを Red Hat Subscription Manager アカウントに登録している。Registering a system using Red Hat Subscription Managerユーザーガイドを参照してください。

手順

1. インストールするパッケージを指定して、**yum** コマンドを実行します。

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

2. インストールが機能することを確認します。

```
$ javac -version

javac 17.0.2
```

2.4. アーカイブを使用した RHEL への OPENJDK のインストール

OpenJDK はアーカイブでインストールできます。これは、Java 管理者が root 権限を持たない場合に便利です。

注記

アップグレードを容易にするために、JRE を含む親ディレクトリーを作成し、汎用パスを使用して最新の JRE へのシンボリックリンクを作成します。

手順

1. アーカイブファイルをダウンロードするディレクトリーを作成し、コマンドラインインターフェース(CLI)でそのディレクトリーに移動します。以下に例を示します。

```
$ mkdir ~/jdks

$ cd ~/jdks
```

2. Red Hat カスタマーポータルの Software Downloads ページに移動します。
3. **Version** ドロップダウンリストから OpenJDK 17 の最新バージョンを選択し、Linux 用の JDK アーカイブをローカルシステムにダウンロードします。
4. アーカイブのコンテンツを任意のディレクトリーに展開します。

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

5. アップグレードを容易にするために、JDK へのシンボリックリンクを使用して汎用パスを作成します。

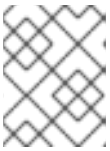
```
$ ln -s ~/jdks/java-17-openjdk-17.0.2.0.8-3.portable.jdk.el7.x86_64 ~/jdks/java-17
```

6. **JAVA_HOME** 環境変数を設定します。

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

7. **JAVA_HOME** 環境変数が正しく設定されていることを確認します。

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



注記

この方法でインストールした場合、Java は現在のユーザーのみが使用できます。

8. 一般的な JRE パスの **bin** ディレクトリーを **PATH** 環境変数に追加します。

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

9. フルパスを指定せずに **java -version** が機能することを確認します。

```
$ 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)
```

注記

~/**.bashrc** に環境変数をエクスポートすることで、**JAVA_HOME** 環境変数が現在のユーザーに対して持続することを確認できます。

2.5. YUM を使用した RHEL への OPENJDK のメジャーバージョンの複数インストール

システムパッケージマネージャー **yum** を使用して、OpenJDK の複数バージョンをインストールできます。

前提条件

- インストールする OpenJDK を提供するリポジトリーへのアクセスを提供するアクティブなサブスクリプションを持つ Red Hat Subscription Manager (RHSM) アカウント。
- システムに対する root 権限がある。

手順

1. 以下の **yum** コマンドを実行してパッケージをインストールします。
   For OpenJDK 17 の場合

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

   For OpenJDK 11 の場合

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

   OpenJDK 8 の場合

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

2. インストール後に、利用可能な Java バージョンを確認します。

   ```
   $ 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. 現在の Java バージョンを確認します。

   ```
   $ 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)
```

注記

OpenJDK の複数のメジャーバージョンをローカルシステムにインストールできます。あるメジャーバージョンから別のメジャーバージョンに切り替える必要がある場合は、コマンドラインインターフェース(CLI)で以下のコマンドを実行し、画面のプロンプトに従います。

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

関連情報

- **java --alternatives** を使用して、使用するデフォルトの Java バージョンを構成できます。詳細は、RHEL でシステム全体の OpenJDK バージョンを非対話的に選択 を参照してください。

2.6. アーカイブを使用して RHEL に OPENJDK の複数のメジャーバージョンをインストール

OpenJDK の複数のメジャーバージョンをインストールするには、**アーカイブを使用した RHEL への JRE のインストール**と同じ手順を使用するか、複数のメジャーバージョンを使用して**アーカイブを使用して RHEL 8 に OpenJDK をインストール**できます。

注記

システムのデフォルトの OpenJDK バージョンを設定する方法は、「Selecting a system-wide java version」を参照してください。

関連情報

- JRE のインストール方法は、Installing a JRE on RHEL using an archive を参照してください。
- JDK のインストール方法は、「アーカイブを使用した RHEL 8 への OpenJDK のインストール」を参照してください。

2.7. YUM を使用して RHEL に OPENJDK の複数のマイナーバージョンをインストール

RHEL には、OpenJDK の複数のマイナーバージョンをインストールできます。これは、インストールされているマイナーバージョンが更新されないようにすることで行われます。

前提条件

- RHEL でシステム全体の OpenJDK バージョンを非対話的に選択 から、システム全体の OpenJDK バージョンを選択します。

手順

1. **/etc/yum.conf** ディレクトリーに**installonlypkgs** オプションを追加して、**yum** がインストール可能でも更新できない OpenJDK パッケージを指定します。

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

更新は、システムに古いバージョンを残したまま、新しいパッケージをインストールします。

```
$ 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. OpenJDK のさまざまなマイナーバージョンは、**/usr/lib/jvm/<minor version>** ファイルにあります。
たとえば、以下は **/usr/lib/jvm/java-17.0.2-openjdk** の一部を示しています。

```
$ /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. アーカイブを使用して RHEL に OPENJDK の複数のマイナーバージョンをインストール

複数のマイナーバージョンのインストールは、複数のマイナーバージョンを使用したアーカイブを使用した RHEL への JRE のインストール またはアーカイブを使用した RHEL 8 への OpenJDK のインストールと同じです。

注記

システムのデフォルトのマイナーバージョンを選択する方法は、「RHEL でシステム全体の OpenJDK バージョンを非対話的に選択」を参照してください。

関連情報

- JRE のインストール方法は、Installing a JRE on RHEL using an archive を参照してください。
- JDK のインストール方法は、「アーカイブを使用した RHEL 8 への OpenJDK のインストール」を参照してください。

第3章 OPENJDK 17 のデバッグシンボル

OpenJDK アプリケーションでクラッシュの調査に役立つシンボルのデバッグに役立ちます。

3.1. デバッグシンボルのインストール

この手順では、OpenJDK のデバッグシンボルをインストールする方法を説明します。

前提条件

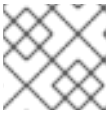
- ローカルの sytem に **gdb** パッケージをインストールしている。
  - CLI で **sudo yum install gdb** コマンドを実行して、ローカルシステムにこのパッケージをインストールできます。

手順

1. デバッグシンボルをインストールするには、以下のコマンドを入力します。

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

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

これらのコマンドは、**java-17-openjdk-debuginfo**、**java-17-openjdk-headless-debuginfo**、および OpenJDK 17 バイナリーのデバッグシンボルを提供する追加パッケージをインストールします。これらのパッケージは自己完全ではなく、実行可能なバイナリーは**含まれません**。



注記

debuginfo-install は、**yum-utils** パッケージで提供されます。

2. デバッグシンボルがインストールされていることを確認するには、以下のコマンドを入力します。

```
$ 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. デバッグシンボルのインストール場所の確認

この手順では、デバッグシンボルの場所を見つける方法を説明します。

注記

debuginfo パッケージがインストールされていても、パッケージのインストール場所を取得できない場合は、正しいパッケージと java バージョンがインストールされているかどうかを確認します。バージョンを確認した後、再度デバッグシンボルの場所を確認してください。

前提条件

- ローカルの sytem に **gdb** パッケージをインストールしている。
  - CLI で **sudo yum install gdb** コマンドを実行して、ローカルシステムにこのパッケージをインストールできます。
  - デバッグシンボルパッケージをインストールしている。Installing the debug symbols を参照してください。

手順

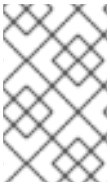
1. デバッグシンボルの場所を見つけるには、**which java** コマンドで **gdb** を使用します。

```
$ 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. 以下のコマンドを使用して ***-debug** ディレクトリーを調べて、**java**、**javac**、および **javah** を含むライブラリーのデバッグバージョンをすべて表示します。

```
$ 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
│   ...
```



注記

javac および **javah** ツールは、**java-17-openjdk-devel** パッケージで提供されます。**$ sudo debuginfo-install java-17-openjdk-devel** コマンドを使用してパッケージをインストールできます。

3.3. デバッグシンボルの設定の確認

デバッグシンボルの設定を確認および設定できます。

- 以下のコマンドを入力して、インストールされているパッケージの一覧を取得します。

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

- デバッグ情報パッケージがインストールされていない場合は、以下のコマンドを実行して、足りないパッケージをインストールします。

```
$ 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
```

- 特定のブレークポイントに到達する場合は、以下のコマンドを実行します。

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

上記のコマンドは、以下のタスクを完了します。

  - JVM はスタックオーバーフローチェックに SEGV を使用するため、SIGSEGV エラーを処理します。
  - 保留中のブレークポイントを **yes** に設定します。
  - **JavaCalls::call** 関数で break ステートメントを呼び出します。HotSpot(libjvm.so)でアプリケーションを起動する関数。

3.4. 致命的なエラーログファイルでのデバッグシンボルの設定

JVM クラッシュが原因で Java アプリケーションがダウンすると、致命的なエラーのログファイルが生成されます (例: **hs_error**、**java_error**)。これらのエラーログファイルは、アプリケーションの現在の作業ディレクトリーに生成されます。クラッシュファイルには、スタックに関する情報が含まれます。

手順

1. **strip -g** コマンドを使用すると、デバッグシンボルをすべて削除できます。
以下のコードは、展開されていない **hs_error** ファイルの例を示しています。

```
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
```

以下のコードは、ストライピング **hs_error** ファイルの例を示しています。

```
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. 以下のコマンドを入力して、同じバージョンのデバッグシンボルと致命的なエラーログファイルがあることを確認します。

```
$ java -version
```

注記

このチェックを完了するには、**sudo update-alternatives --config 'java'** を使用することもできます。

3. **nm** コマンドを使用して、**libjvm.so** に ELF データおよびテキストシンボルがあることを確認します。

```
$ 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
```

関連情報

- クラッシュファイル **hs_error** は、デバッグシンボルがインストールされない状態で不完全です。詳細は、「Java application down due to JVM crash 」を参照してください。

第4章 RED HAT ENTERPRISE LINUX での OPENJDK 17 のインストール

次のセクションでは、Red Hat Enterprise Linux で OpenJDK 17 を更新する手順について説明します。

4.1. YUM を使用して RHEL で OPENJDK 17 を更新

インストールされている OpenJDK パッケージは、**yum** システムパッケージマネージャーを使用して更新できます。

要件

- システムに対する root 権限がある。

手順

1. 現在の OpenJDK バージョンを確認します。

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

インストールされている OpenJDK パッケージの一覧が表示されます。

```
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. 特定のパッケージを更新します。以下に例を示します。

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

3. 現在の OpenJDK バージョンをチェックして、更新が機能していることを確認します。

```
$ 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)
```

注記

OpenJDK の複数のメジャーバージョンをローカルシステムにインストールできます。あるメジャーバージョンから別のメジャーバージョンに切り替える必要がある場合は、コマンドラインインターフェース(CLI)で以下のコマンドを実行し、画面のプロンプトに従います。

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

4.2. アーカイブを使用して RHEL での OPENJDK 17 の更新

アーカイブを使用して OpenJDK を更新できます。これは、OpenJDK 管理者が root 権限を持たない場合に便利です。

要件

- JDK または JRE のインストールを指定する一般的なパスを把握している。例: **~/jdks/java-17**

手順

1. JDK または JRE への汎用パスの既存のシンボリックリンクを削除します。
以下に例を示します。

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

2. インストール場所に最新バージョンの JDK または JRE をインストールします。

関連情報

- JRE のインストール方法は、Installing a JRE on RHEL using an archive を参照してください。
- JDK のインストール方法は、「アーカイブを使用した RHEL 8 への OpenJDK のインストール」を参照してください。

改定日時: 2022-04-16 17:17:23 +1000