# Red Hat build of OpenJDK 11

## Eclipse Temurin 11.0.17 リリースノート

Last Updated: 2024-05-11

## 法律上の通知



## 概要

本リリースノートを確認して、Eclipse Temurin で提供される OpenJDK 11 の最新ビルドに含まれる新機能および機能拡張について説明します。

# 目次

# はじめに

Open Java Development Kit (OpenJDK) は、Java Platform Standard Edition (Java SE) のオープンソース実装です。Eclipse Temurin は、OpenJDK 8u、OpenJDK 11u、および OpenJDK 17u の 3 つの LTS バージョンで利用できます。

Eclipse Temurin 用のパッケージは、Microsoft Windows および Red Hat Enterprise Linux および Ubuntu を含む複数の Linux x86 オペレーティングシステムで利用できます。

# RED HAT BUILD OF OPENJDK ドキュメントへのフィードバック

エラーを報告したり、ドキュメントを改善したりするには、Red Hat Jira アカウントにログインし、課題を送信してください。Red Hat Jira アカウントをお持ちでない場合は、アカウントを作成するように求められます。

**手順**

1. 次のリンクをクリックして **チケットを作成します**。
2. **Summary** に課題の簡単な説明を入力します。
3. **Description** に課題や機能拡張の詳細な説明を入力します。問題があるドキュメントのセクションへの URL を含めてください。
4. **Submit** をクリックすると、課題が作成され、適切なドキュメントチームに転送されます。

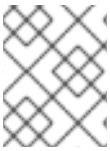
# 多様性を受け入れるオープンソースの強化

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、Red Hat CTO である Chris Wright のメッセージ をご覧ください。

# 第1章 ECLIPSE TEMURIN のサポートポリシー

Red Hat は、一部の Eclipse Temurin のメジャーバージョンをサポートします。一貫性を保つために、これらのバージョンは、Oracle が Oracle JDK 向けに長期サポート (LTS) を指定しているバージョンと同じになります。

Eclipse Temurin のメジャーバージョンは、最初に導入された時点から少なくとも 6 年間サポートされます。詳細は、Eclipse Temurin のライフサイクルおよびサポートポリシー を参照してください。



**注記**

RHEL 6 のライフサイクルは 2020 年 11 月に終了します。このため、Eclipse Temurin はサポート対象の構成として RHEL 6 をサポートしません。

# 第2章 ECLIPSE TEMURIN の機能

Eclipse Temurin には、OpenJDK のアップストリームディストリビューションの構造の変更は含まれません。

Eclipse Temurin の最新の OpenJDK 11.0.17 リリースに含まれる変更およびセキュリティー修正の一覧は、OpenJDK 11.0.17 Released を参照してください。

## 新機能および拡張された機能

次のリリースノートを確認して、Eclipse Temurin 11.0.17 リリースに含まれる新機能と機能拡張を理解してください。

### cpu.shares パラメーターが無効になっている

OpenJDK 11.0.17 リリースより前は、OpenJDK は、**cgroups** とも呼ばれる Linux コントロールグループに属する **cpu.shares** パラメーターの誤った解釈を使用していました。このパラメーターにより、Java 仮想マシン (JVM) が使用可能な CPU よりも少ない CPU を使用する可能性があり、コンテナー内で動作するときの JVM の CPU リソースとパフォーマンスに影響を与える可能性があります。

OpenJDK 11.0.17 リリースでは、スレッドプールのスレッド数を決定するときに **cpu.shares** パラメーターを使用しないように JVM が設定されます。この設定を元に戻したい場合は、JVM の起動時に **-XX:+UseContainerCpuShares** 引数を渡します。

**注記**

**-XX:+UseContainerCpuShares** 引数は非推奨の機能であり、将来の OpenJDK リリースで削除される可能性があります。

JDK-8281181 (JDK バグシステム) を参照してください。

### jdk.httpserver.maxConnections システムプロパティー

OpenJDK 11.0.17 は、新しいシステムプロパティー **jdk.httpserver.maxConnections** を追加します。これにより、**HttpServer** サービスに対して接続制限が指定されていないというセキュリティー上の問題が修正され、受け入れられた接続と確立された接続が無期限に開いたままになる可能性があります。

**jdk.httpserver.maxConnections** システムプロパティーを使用して、以下のように **HttpServer** サービスの動作を変更できます。

- **0** の値または **-1** などの負の値を設定して、サービスへの接続制限を指定します。
- **1** などの正の値を設定すると、サービスは、確立された接続の現在の数に対して、受け入れた接続をチェックします。サービスに確立された接続に達すると、サービスは受け入れた接続をすぐに終了します。

JDK-8286918 (JDK Bug System) を参照してください。

### JFR を使用してオブジェクトのデシリアライゼーションを監視する

JDK Flight Recorder (JFR) を使用して、オブジェクトのデシリアライズを監視できるようになりました。デフォルトでは、OpenJDK 11.0.17 は JFR の **jdk.deserialization** イベント設定を無効にします。この機能を有効にするには、JFR 設定の **event-name** 要素を更新します。以下に例を示します。

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration version="2.0" description="test">
 <event name="jdk.Deserialization">
 <setting name="enabled">true</setting>
```

```
 <setting name="stackTrace">false</setting>
 </event>
</configuration>
```

JFR を有効にし、デシリアライゼーションイベントを監視するように JFR を設定すると、監視対象のアプリケーションがオブジェクトをデシリアライズしようとするたびに、JFR によってイベントが作成されます。次に、JFR のシリアライゼーションフィルターメカニズムは、監視対象アプリケーションからデシリアライズされたオブジェクトを受け入れるか拒否するかを決定できます。

JDK-8261160 (JDK Bug System) を参照してください。

### SHA-1 署名 JAR

OpenJDK 11.0.17 リリースでは、**SHA-1** アルゴリズムで署名された JAR はデフォルトで制限され、署名されていないかのように扱われます。これらの制限は、次のアルゴリズムに適用されます。

- ダイジェスト、署名、およびオプションで JAR のタイムスタンプに使用されるアルゴリズム。
- コード署名者とタイムスタンプ機関の証明書チェーン内の証明書の署名アルゴリズムとダイジェストアルゴリズム、およびそれらの証明書が失効しているかどうかを確認するために使用される証明書失効リスト (CRL) またはオンライン証明書ステータスプロトコル (OCSP) 応答。

さらに、制限は署名済みの Java Cryptography Extension (JCE) プロバイダーにも適用されます。

以前にタイムスタンプが付けられた JAR の互換性リスクを軽減するために、この制限は、**SHA-1** アルゴリズムで署名され、**January 01, 2019** より前にタイムスタンプが付けられた JAR には適用されません。この例外は、将来の OpenJDK リリースで削除される可能性があります。

JAR ファイルが制限の影響を受けるかどうかを判断するには、CLI で次のコマンドを発行します。

```
$ jarsigner -verify -verbose -certs
```

前のコマンドの出力から、**SHA1**、**SHA-1**、または **disabled** のインスタンスを検索します。さらに、JAR が署名なしとして扱われることを示す警告メッセージを検索します。以下に例を示します。

```
Signed by "CN="Signer""
Digest algorithm: SHA-1 (disabled)
Signature algorithm: SHA1withRSA (disabled), 2048-bit key

WARNING: The jar will be treated as unsigned, because it is signed with a weak algorithm that is now disabled by the security property:

jdk.jar.disabledAlgorithms=MD2, MD5, RSA keySize < 1024, DSA keySize < 1024, SHA1 denyAfter 2019-01-01
```

新しい制限の影響を受けるすべての JAR をより強力なアルゴリズムに置き換えるか、再署名することを検討してください。

JAR ファイルがこの制限の影響を受ける場合は、アルゴリズムを削除して、**SHA-256** などのより強力なアルゴリズムでファイルに再署名できます。OpenJDK 11.0.17 の **SHA-1** 署名付き JAR に対する制限を削除する必要があり、セキュリティーリスクを受け入れる場合は、次のアクションを実行できます。

1. **java.security** 設定ファイルを変更します。または、このファイルを保存して、必要な設定で別のファイルを作成することもできます。
2. **SHA1 usage SignedJAR & denyAfter 2019 01 011** エントリーを **jdk.certpath.disabledAlgorithms** セキュリティープロパティーから削除します。

3. **jdk.jar.disabledAlgorithms** セキュリティープロパティーから **SHA1 denyAfter 2019-01-01** エントリーを削除します。

**注記**

**java.security** ファイルの **jdk.certpath.disabledAlgorithms** の値は、RHEL 8 および 9 のシステムセキュリティーポリシーによって上書きされる場合があります。システムセキュリティーポリシーで使用される値は、ファイル **/etc/crypto-policies/back-ends/java.config** で確認でき、java.security ファイルで **security.useSystemPropertiesFile** を false に設定するか、**-Djava.security.disableSystemPropertiesFile=true** を JVM 渡すことで無効にします。これらの値はこのリリースでは変更されていないため、値は OpenJDK の以前のリリースと同じままです。

**java.security** ファイルの設定例については、JBoss EAP for OpenShift の **java.security** プロパティーのオーバーライド (Red Hat カスタマーポータル) を参照してください。

JDK-8269039 (JDK バグシステム) を参照してください。

### HTTPURLConnection の keep-alive 動作を制御するためのシステムプロパティー

OpenJDK 11.0.17 リリースには、**HTTPURLConnection** の **keep-alive** 動作を制御するために使用できる次の新しいシステムプロパティーが含まれています。

- サーバーへの接続を制御する **http.keepAlive.time.server**。
- プロキシーへの接続を制御する **http.keepAlive.time.proxy**。

OpenJDK 11.0.17 リリースより前では、**keep-alive** 時間が指定されていないサーバーまたはプロキシーにより、ハードコーディングされたデフォルト値によって定義された期間、アイドル接続が開いたままになる場合がありました。

OpenJDK 11.0.17 では、システムプロパティーを使用して **keep-alive** 時間のデフォルト値を変更できます。**keep-alive** プロパティーは、サーバーまたはプロキシーのいずれかの HTTP **keep-alive** 時間を変更することでこの動作を制御します。これにより、OpenJDK の HTTP プロトコルハンドラーは、指定された秒数が経過した後にアイドル状態の接続を閉じます。

OpenJDK 11.0.17 リリースより前では、次の使用例は、**HTTPURLConnection** の特定の **keep-alive** 動作につながります。

- サーバーが **Connection:keep-alive** ヘッダーを指定し、サーバーの応答に **Keep-alive:timeout=N** が含まれている場合、クライアントの OpenJDK **keep-alive** キャッシュは **N** 秒のタイムアウトを使用します (**N** は整数値)。
- サーバーが **Connection:keep-alive** ヘッダーを指定しているが、サーバーの応答に **Keep-alive:timeout=N** のエントリーが含まれていない場合、クライアントの OpenJDK **keep-alive** キャッシュはプロキシーに対して **60** 秒のタイムアウトを使用し、**5** サーバーの秒。
- サーバーが **Connection:keep-alive** ヘッダーを指定しない場合、クライアントの OpenJDK **keep-alive** キャッシュは、すべての接続に対して 5 秒のタイムアウトを使用します。

OpenJDK 11.0.17 リリースでは、前述の動作が維持されていますが、2 番目と 3 番目に挙げた使用例におけるタイムアウトは、デフォルトの設定に依存するのではなく、**http.keepAlive.time.server** および **http.keepAlive.time.proxy** プロパティーを使用して指定できるようになっています。

**注記**

**keep-alive** プロパティーを設定し、サーバーが **Keep-Alive** 応答ヘッダーの **keep-alive** 時間を指定した場合、HTTP プロトコルハンドラーはサーバーによって指定された時間を使用します。この状況は、プロキシーと同じです。

JDK-8278067 (JDK バグシステム) を参照してください。

### デフォルトの PKCS #12 MAC アルゴリズムを更新

OpenJDK 11.0.17 は、PKCS #12 キーストアのデフォルトのメッセージ認証コード (MAC) アルゴリズムを更新して、**SHA-1** 関数ではなく **SHA-256** 暗号化ハッシュ関数を使用します。**SHA-256** 関数は、データを保護するためのより強力な方法を提供します。

この更新は、**keystore.pkcs12.macAlgorithm** および **keystore.pkcs12.maclerationCount** システムプロパティーで確認できます。

この更新された MAC アルゴリズムでキーストアを作成し、そのキーストアを OpenJDK 11.0.12 より前のバージョンの OpenJDK で使用しようとすると、**java.security.NoSuchAlgorithmException** メッセージが表示されます。

OpenJDK 11.0.12 より前の OpenJDK バージョンで以前のキーストアを使用するには、**keystore.pkcs12.legacy** システムプロパティーを **true** に設定して、MAC アルゴリズムを元に戻します。

JDK-8267880 (JDK Bug System) を参照してください。

### 非推奨および削除された機能

次のリリースノートを確認して、OpenJDK 11.0.17 リリースで非推奨または削除された既存の機能を理解してください。

### 非推奨の Kerberos 暗号化タイプ

OpenJDK 11.0.17 は、**des3-hmac-sha1** および **rc4-hmac** Kerberos 暗号化タイプを廃止します。デフォルトでは、OpenJDK 11.0.17 はこれらの暗号化タイプを無効にしますが、次のアクションを完了することで有効にすることができます。

- **krb5.conf** 設定ファイルで、**allow_weak_crypto** タブを **true** に設定します。この設定により、**des-cbc-crc** や **des-cbc-md5** などの他の暗号化タイプも有効になります。

**警告**

この設定を適用する前に、Kerberos の認証メカニズムに弱い暗号化アルゴリズムを導入するなど、これらの弱い Kerberos 暗号化タイプをすべて有効にするリスクを考慮してください。

次の **krb5.conf** 設定ファイルの設定のいずれかに暗号化タイプを明示的にリストすることで、弱い暗号化タイプのサブセットを無効にすることができます。

- **default_tkt_enctypes**
- **default_tgs_enctypes**

- **permitted_enctypes**

JDK-8139348 (JDK Bug System) を参照してください。

**改訂日時: 2024-05-10**