Red Hat build of Eclipse Vert.x 3.8

Release Notes for Eclipse Vert.x 3.8

For use with Eclipse Vert.x 3.8.5

Last Updated: 2020-02-28

Red Hat build of Eclipse Vert.x 3.8 Release Notes for Eclipse Vert.x 3.8

For use with Eclipse Vert.x 3.8.5

Legal Notice

Abstract

This Release Note contains important information related to Eclipse Vert.x 3.8.5

Table of Contents

PREFACE

Date of release: 2020-02-24

CHAPTER 1. REQUIRED INFRASTRUCTURE COMPONENT VERSIONS

Red Hat does not provide support for components listed below, with the exception of components explicitly designated as supported.

Component name	Version
Maven	3.6.0
Fabric8 Maven Plugin	4.3.1
JDK[a][b]	OpenJDK 8, OpenJDK 11[c]
Red Hat Enterprise Linux 7[d]	7.7
Red Hat Enterprise Linux 8[e]	8.1
OpenShift Container Platform (OCP)[f]	3.11, 4.3
Minishift	1.34.2 or later
CDK[g]	3.11.0
git	2.0 or later
oc command line tool	3.11 or later[h]

[a] A full JDK installation is required, as JRE does not provide tools for compiling Java applications from source.

[b] Red Hat OpenJDK is supported by Red Hat

[c] OpenJDK 9 is not supported by Red Hat.

[d] For deploying RHOAR-based applications on stand-alone RHEL in a production environment.

[e] For deploying RHOAR-based applications on stand-alone RHEL in a production environment.

[f] OCP is supported by Red Hat

[g] CDK is supported by Red Hat

[h] The version of the **oc** CLI tool should correspond to the version of OCP that you are using.

CHAPTER 2. SUPPORTED ECLIPSE VERT.X RUNTIME COMPONENT CONFIGURATIONS AND INTEGRATIONS

The following resource defines the supported configurations and integrations of Red Hat products with Eclipse Vert.x:

- For a list of technologies that are supported for integration with Eclipse Vert.x in production environments see the Supported Eclipse Vert.x configurations and integrations.

- For a list of Eclipse Vert.x runtime artifacts and their versions see the Eclipse Vert.x 3.8.5 component details page.

CHAPTER 3. FEATURES

3.1. NEW AND CHANGED FEATURES

No new features or functionalities are introduced in this release.

3.2. DEPRECATED FEATURES

3.2.1. New connection handler method

The **HttpClientRequest.connectionHandler()** method is deprecated and will be removed in a future release. Use **HttpClient.connectionHandler()** method instead to call connection handlers for client requests in your application. For example, use the following code to work with **HttpClient.connectionHandler()** method:

```
client.connectionHandler(conn -> {
 // Connection related code
});
```

3.2.2. Updates in HTTP Methods for WebSocket

The updates in the HTTP methods for WebSocket are:

- The usage of the term *WebSocket* in method names was inconsistent. The method names had incorrect capitalization, for example, *WebSocket*, instead of *WebSocket*. The methods that had inconsistent usage of *WebSocket* in the following classes have been deprecated and will be removed in a future release. Use the new methods that have correct capitalization instead.
  - The following methods in **HttpServerOptions** class are deprecated.

Deprecated Methods	New Methods
getMaxWebsocketFrameSize()	**getMaxWebSocketFrameSize()**
setMaxWebsocketFrameSize()	**setMaxWebSocketFrameSize()**
getMaxWebsocketMessageSize()	**getMaxWebSocketMessageSize()**
setMaxWebsocketMessageSize()	**setMaxWebSocketMessageSize()**
getPerFrameWebsocketCompressionSupported()	**getPerFrameWebSocketCompressionSupported()**
setPerFrameWebsocketCompressionSupported()	**setPerFrameWebSocketCompressionSupported()**
getPerMessageWebsocketCompressionSupported()	**getPerMessageWebSocketCompressionSupported()**

Deprecated Methods	New Methods
setPerMessageWebsocketCompressionSupported()	**setPerMessageWebSocketCompressionSupported()**
getWebsocketAllowServerNoContext()	**getWebSocketAllowServerNoContext()**
setWebsocketAllowServerNoContext()	**setWebSocketAllowServerNoContext()**
getWebsocketCompressionLevel()	**getWebSocketCompressionLevel()**
setWebsocketCompressionLevel()	**setWebSocketCompressionLevel()**
getWebsocketPreferredClientNoContext()	**getWebSocketPreferredClientNoContext()**
setWebsocketPreferredClientNoContext()	**setWebSocketPreferredClientNoContext()**
getWebsocketSubProtocols()	**getWebSocketSubProtocols()**
setWebsocketSubProtocols()	**setWebSocketSubProtocols()**

The new methods for WebSocket subprotocols use **List<String>** data type instead of a comma separated string to store items.

- The following methods in **HttpClientOptions** class are deprecated.

Deprecated Methods	New Methods
getTryUsePerMessageWebsocketCompression()	**getTryUsePerMessageWebSocketCompression()**
setTryUsePerMessageWebsocketCompression()	**setTryUsePerMessageWebSocketCompression()**
getTryWebsocketDeflateFrameCompression()	**getTryWebSocketDeflateFrameCompression()**
getWebsocketCompressionAllowClientNoContext()	**getWebSocketCompressionAllowClientNoContext()**
setWebsocketCompressionAllowClientNoContext()	**setWebSocketCompressionAllowClientNoContext()**
getWebsocketCompressionLevel()	**getWebSocketCompressionLevel()**
setWebsocketCompressionLevel()	**setWebSocketCompressionLevel()**

Deprecated Methods	New Methods
getWebsocketCompressionRequestServerNoContext()	**getWebSocketCompressionRequestServerNoContext()**
setWebsocketCompressionRequestServerNoContext()	**setWebSocketCompressionRequestServerNoContext()**

- The following handler methods in **HttpServer** class are deprecated.

Deprecated Methods	New Methods
websocketHandler()	**webSocketHandler()**
websocketStream()	**webSocketStream()**

- **WebsocketRejectedException** is deprecated. The methods will throw the **UpgradeRejectedException** instead.

3.2.3. Deprecated authentication and authorization classes and methods

The following authentication and authorization classes and methods are deprecated and will be replaced in a future release.

- Classes:
  - **AbstractUser**
  - **PubSecKeyOptions**
  - **JDBCAuthOptions**
  - **JDBCHashStrategy**
  - **AccessToken**
  - **KeycloakHelper**
  - **ShiroAuth**
  - **AuthProviderInternal**
- Methods:
  - **User.isAuthorized()**
  - **User.clearCache()**
  - **User.setAuthProvider()**
  - **Oauth2Auth.introspectToken()**
  - **Oauth2Auth.getFlowType()**

- **Oauth2Auth.loadJWK()**
- **Oauth2Auth.rbacHandler()**
- **Oauth2ClientOptions.isUseBasicAuthorization()**
- **Oauth2ClientOptions.setUseBasicAuthorizationHeader()**
- **Oauth2ClientOptions.getScopeSeparator()**
- **Oauth2ClientOptions.setScopeSeparator()**

3.2.4. Methods to create clients that have no shared data sources

Use the following new methods to create clients that do not have shared data sources with other clients. These methods maintain their own data sources.

Deprecated Methods	New Methods
MongoClient.WebsocketRejectedExceptioncreateNonShared()	**MongoClient.create()**
JDBCClient.createNonShared()	**wJDBCClient.create()**
CassandraClient.createNonShared()	**CassandraClient.create()**
MailClient.createNonShared()	**MailClient.create()**

CHAPTER 4. RELEASE COMPONENTS

4.1. SUPPORTED ARTIFACTS INTRODUCED IN THIS RELEASE

No artifacts have been moved form Technology Preview to fully supported in this release:

4.2. TECHNOLOGY PREVIEW ARTIFACTS INTRODUCED IN THIS RELEASE

No new artifacts are provided as Technology Preview in this release.

NOTE

For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

4.3. ARTIFACTS REMOVED IN THIS RELEASE

No artifacts are removed in this release.

4.4. ARTIFACTS DEPRECATED IN THIS RELEASE

No artifacts are marked as deprecated in this release.

CHAPTER 5. FIXED ISSUES

This Eclipse Vert.x release incorporates all bugfixes from community release of versions 3.8.4 and 3.8.5. Issues resolved in the community releases are listed in the Eclipse Vert.x 3.8.4 Community Release Notes and the Eclipse Vert.x 3.8.5 Community Release Notes .

5.1. THE STARR.VERSION PROPERTY IS NO LONGER INDICATED AS MISSING IN PROJECTS THAT INCLUDE THE VERTX-KAFKA-CLIENT ARTIFACT

In previous releases of Eclipse Vert.x, when building a project that includes the **vertx-kafka-client** artifact, Maven generated the following warning, potentially causing the build to fail:

```
The POM for org.scala-lang:scala-compiler:jar:${starr.version} is missing, no dependency information available
```

This issue is resolved in the Eclipse Vert.x 3.8.5 release and no longer occurs.

5.2. VERT.X AMQP CLIENT:`AMQPRECEIVER` LOGS RECEIVED MESSAGES TO STANDARD OUTPUT AS EXPECTED

In previous releases of Eclipse Vert.x, a **messageHandler** attached to an **AmqpReceiver** instance that was set to print a message received from a **Sender** instance located on the same address to standard output would fail to print the message to the log despite having received it. The issue resulted from using a deprecated **createReceiver** method to initialize the **AmqpReceiver** instance. The deprecated method is removed from the **vertx-amqp-client** module in the Eclipse Vert.x 3.8.5 release and the issue no longer occurs.

CHAPTER 6. KNOWN ISSUES

6.1. CONNECTION BETWEEN A RHEL 8-BASED DATABASE APPLICATION AND A RHEL 7-BASED MYSQL 5.7 DATABASE FAILS DUE TO TLS PROTOCOL VERSION MISMATCH

Description

Attempting to open a TLS-secured connection using OpenSSL between an application container built on a RHEL 8-based OpenJDK builder image and a database container built on a RHEL 7-based MySQL 5.7 container image results in a connection failure due to a **javax.net.ssl.SSLHandshakeException** at runtime: For more detail, view the issue in JIRA .

```
...
Caused by: javax.net.ssl.SSLHandshakeException: No appropriate protocol (protocol is disabled or cipher suites are inappropriate)
...
```

Cause

The issue occurs due to a difference in the latest supported TLS protocol version between RHEL 7 and RHEL 8. The TLS implementation on RHEL 7 supports TLS protocol versions 1.0 (deprecated), 1.1, and 1.2. The TLS implementation on RHEL 8 also supports TLS protocol version 1.3, which is also the default TLS version used in RHEL 8-based builder images. This discrepancy may cause a TLS protocol version mismatch between application components while negotiating a TLS handshake, which in turn causes the connection between the application and database containers to fail.

Workaround

To prevent the issue described above, manually specify a TLS protocol version that is supported on both operating system versions in your database connection string. For example:

```
jdbc:mysql://testdb-mysql:3306/testdb?enabledTLSProtocols=TLSv1.2
```

6.2. FALSE CONNECTION RESET BY PEER ERROR MESSAGES WHEN CALLING APPLICATION ENDPOINT

Making an HTTP request on an endpoint of a Vert.x application using either curl or a Java HTTP client, produces the following error in the logs after each request:

```
io.vertx.core.net.impl.ConnectionBase
SEVERE: java.io.IOException: Connection reset by peer
```

This behavior is caused by the interaction of the Netty application framework and the HAProxy load-balancer used by OpenShift. The error occurs due to existing HTTP connections being re-used by HAProxy without closing. Even though the error message is logged, no error condition occurs. HTTP requests are handled correctly and the application responds as expected.

CHAPTER 7. ADVISORIES RELATED TO THIS RELEASE

The following advisories have been issued to document enhancements, bugfixes, and CVE fixes included in this release.

- RHSA-2020:0567