Red Hat build of Eclipse Vert.x 4.1

Eclipse Vert.x 4.1 リリースノート

Eclipse Vert.x 4.1.8 向け

Last Updated: 2022-04-03

Red Hat build of Eclipse Vert.x 4.1 Eclipse Vert.x 4.1 リリースノート

Eclipse Vert.x 4.1.8 向け

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

本リリースノートには、Eclipse Vert.x 4.1.8 に関連する重要な情報が含まれています。

目次

前書き

リリース日： 2022-01-17

RED HAT ドキュメントへのフィードバック (英語のみ)

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。

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。

第1章 必要なインフラストラクチャーコンポーネントのバージョン

Red Hat ビルドの Eclipse Vert.x を使用する場合は、以下のコンポーネントを使用できます。ただし、Red Hat は、Red Hat OpenShift クラスターを除く以下のコンポーネントのサポートは提供していません。

必要なコンポーネント

Eclipse Vert.x を使用してアプリケーションをビルドし、開発するには、以下のコンポーネントが必要です。

コンポーネント名	Version
Maven	3.6.0 以降
JDK[a]	8 または 11

[a] JRE はソースから Java アプリケーションをコンパイルするツールを提供しないため、完全な JDK インストールが必要です。

オプションのコンポーネント

Red Hat は、開発環境および実稼働環境に応じて、以下のコンポーネントを使用することを推奨します。

コンポーネント名	Version
git	2.0 以降
OpenShift Maven Plugin	1.1.1
oc コマンドラインツール	3.11 以降[a]
Red Hat OpenShift クラスターへのアクセス[b]	3.11 以降

[a] CLI ツール **oc** のバージョンは、使用している OCP のバージョンに対応する必要があります。

[b] OpenShiftCluster は Red Hat によってサポートされます。

第2章 サポートされる ECLIPSE VERT.X RUNTIME コンポーネント設定および統合

以下のリソースは、Eclipse Vert.x と Red Hat 製品のサポートされる設定および統合を定義します。

- 実稼働環境で Eclipse Vert.x との統合 でサポートされる技術の一覧は、「サポートされる Eclipse Vert.x 設定および統合」を参照してください。
- Eclipse Vert.x ランタイムアーティファクトとそのバージョンの一覧は、コンポーネントの詳細ページ を参照してください。

第3章 特長

3.1. 新機能および変更された機能

本項では、本リリースで導入された新機能を説明します。また、既存の機能の変更に関する情報も含まれます。

3.1.1. OpenJDK11 OpenShift イメージは複数のアーキテクチャーをサポートします

IBM Z および IBM Power Systems の OpenJ9 イメージが非推奨になりました。次の OpenJDK11 イメージは、複数のアーキテクチャーをサポートするように更新されています。

- **ubi8/openjdk-11**

OpenJDK11 イメージは、次のアーキテクチャーで使用できます。

- x86 (x86_64)
- s390x (IBM Z)
- ppc64le (IBM Power Systems)

3.1.2. FIPS が有効な Red Hat Enterprise Linux(RHEL)システムでの Eclipse Vert.x Runtime のサポート

Red Hat ビルドの Eclipse Vert.x は FIPS が有効な RHEL システムで実行され、RHEL が提供する FIPS 認定ライブラリーを使用します。

3.1.3. HTTP クライアントのリダイレクトハンドラーのヘッダー伝播

Eclipse Vert.x 4.1.0 以降では、HTTP リダイレクトにヘッダーがある場合は、HTTP クライアントのリダイレクトハンドラーはヘッダーを次の要求に伝播します。この変更により、リダイレクトハンドラーは、リダイレクトされた要求全体をより詳細に制御できるようになります。

以前のリリースの Eclipse Vert.x では、ヘッダーのあるリダイレクト要求があったため、HTTP クライアントはリダイレクト後にヘッダーを処理していました。

以下の例は、Eclipse Vert.x 4.1.0 でリダイレクトを処理する方法を示しています。

```
RequestOptions options = new RequestOptions();
options.setMethod(HttpMethod.GET);
options.setHost(uri.getHost());
options.setPort(port);
options.setSsl(ssl);
options.setURI(requestURI);

// From 4.1.0 propagate headers
options.setHeaders(resp.request().headers());
options.removeHeader(CONTENT_LENGTH);
```

3.1.4. Infinispan 12 へのアップグレード

Eclipse Vert.x 4.1.0 では、Infinispan クラスターマネージャーが更新され、Infinispan 12 をベースにしています。

Infinispan 11 には、マルチマップキャッシュにバイトアレイを保存できないバグがありました。Eclipse Vert.x クラスターマネージャーは、内部 Infinispan クラス **WrappedBytes** を使用して eventbus サブスクリプションデータを保存する必要がありました。この問題は Infinispan 12 で修正されました。

3.1.5. JSON 設定は、MongoDB クライアントの接続文字列オプションよりも優先されます。

Eclipse Vert.x 4.1.0 では、**connection_string** オプションが利用可能な場合でも JSON 設定オプションが適用されます。

次の設定オプションが適用されるようになりました。

```
{
   mongo:{
      db_name: "mydb"
      connection_string: "mongodb://localhost:27017"
      maxPoolSize: 10
      minPoolSize: 3
   }
}
```

以前のリリースの Eclipse Vert.x では、接続文字列が利用可能な場合に JSON 設定オプションは無視されていました。たとえば、前述の例を見てみましょう。以前のリリースの Eclipse Vert.x では、**db_name**、**maxPoolSize**、および **minPoolSize** オプションは無視されていました。

3.1.6. 非推奨の JWT オプションメソッドの削除

Eclipse Vert.x 4.0 以降では、スコープの処理に JWT および OAuth2 ハンドラーが使用されます。

Eclipse Vert.x 4.1.0 以降、**JWTOptions.setScopes(List<String>)**、**JWTOptions.addScope(String)**、および **JWTOptions.withScopeDelimiter(String)** メソッドが削除されました。これらのメソッドは仕様に準拠していませんでした。

以下の例は、Eclipse Vert.x 4.1.0 でスコープを処理する方法を示しています。

```
// before 4.1.0
JWTAuthOptions authConfig = new JWTAuthOptions()
  .setJWTOptions(new JWTOptions()
    .addScope("a")
    .addScope("b")
    .withScopeDelimiter(" ")));

JWTAuth authProvider = JWTAuth.create(vertx, authConfig);

router.route("/protected/*").handler(JWTAuthHandler.create(authProvider));

// in 4.1.0
JWTAuth authProvider = JWTAuth.create(vertx, new JWTAuthOptions());

router.route("/protected/*").handler(
  JWTAuthHandler.create(authProvider)
```

```
    .addScope("a")
    .addScope("b")
    .withScopeDelimiter(" "));
```

3.1.7. 関数を受け入れるカスタムフォーマッターメソッドの非推奨

Eclipse Vert.x 4.1.0 から **LoggerHandler.customFormatter(Function)** メソッドが非推奨になりました。この関数は **HttpServerRequest** を入力として取り、フォーマットされたログ文字列を返します。出力は文字列であるため、コンテキストにアクセスすることはできません。

代わりに新しいメソッド **LoggerHandler customFormatter(LoggerFormatter formatter)** を使用してください。このメソッドは、コンテキストへのアクセスを提供するカスタムフォーマッターを入力として取ります。

3.1.8. HTTP の失敗を処理する新しい例外

Eclipse Vert.x 4.1.0 以降では、HTTP の失敗の処理に使用できる新しい例外クラス **io.vertx.ext.web.handler.HttpException** を利用できます。この例外を使用して、500 以外のカスタムステータスコードを指定できます。たとえば、新規の **HttpException(401, "Forbidden")** は、禁止されているリクエストがステータスコード 401 を返す必要があることを示します。

3.1.9. RxJava 3 のサポート

Eclipse Vert.x 4.1.0 以降では、RxJava 3 がサポートされます。

- 新しい rxified API が **io.vertx.rxjava3** パッケージで利用できます。
- Eclipse Vert.x JUnit5 との統合は、**vertx -junit5-rx-java3** バインディングによって提供されます。

3.1.10. コンテキストサーバーインターセプターはすべてのタイプのデータをバインドし、より安全です。

Eclipse Vert.x 4.0.3 以降、**ContextServerInterceptor.bind ()** メソッドはすべてのタイプのデータをコンテキストにバインドします。このメソッドはストレージの詳細を公開しないため、より安全になりました。

Eclipse Vert.x 4.0.3 より前のリリースでは、このメソッドは 'String' データ型のみをコンテキストにバインドしていました。また、ストレージの詳細も公開しました。

更新された **ContextServerInterceptor.bind ()** メソッドを使用するには、アプリケーションを更新する必要があります。

以下の例は、Eclipse Vert.x 4.0.3 より前のリリースのコードを示しています。

```
// Example code from previous releases

class X extends ContextServerInterceptor {
  @Override
  public void bind(Metadata metadata, ConcurrentMap<String, String> context) {
```

次の例は、Eclipse Vert.x 4.0.3 リリースの置換コードを示しています。

```
// Replacing code for Eclipse Vert.x 4.0.3 release
```

```
class X extends ContextServerInterceptor {
  @Override
  public void bind(Metadata metadata) {
```

3.1.11. ワイルドカード文字で終わるルートパスの末尾のスラッシュ (/) の一致が不要

Eclipse Vert.x 4.0.3 より前のリリースでは、ルートがスラッシュで終わるパスとワイルドカード /* で定義されている場合、一致するリクエストにも末尾のスラッシュ/ が含まれている場合にのみ、ルートが呼び出されました。このルールは、ワイルドカードが空の場合に問題を引き起こしました。

Eclipse Vert.x 4.0.3 以降では、このルールは適用されなくなりました。パスがスラッシュ (/) で終わるルートを作成できます。ただし、リクエスト URL にスラッシュを指定することは必須ではありません。

また、リクエスト URL を作成および使用し、パスにスラッシュ (/) ではなく、ワイルドカードで終わるルートを呼び出すこともできます。たとえば、ワイルドカードが含まれるルートは **/foo/*** として定義できます。ここでは、ルートはパスの最後にあるオープンワイルドカードと一致する必要があります。リクエスト URL は **/foo** にすることができます。

この表は、Eclipse Vert.x 4.0.3 以前のリリースでのリクエスト URL **/foo/** * を送信するときの動作を示しています。Eclipse Vert.x 4.0.3 では終了スラッシュが任意であり、要求はルートに一致することがわかります。

Route	Eclipse Vert.x 4.0.3	Eclipse Vert.x 4.0.3 より前のリリース
/foo	Match	No Match
/foofighters	No Match	No Match
/foo/	Match	Match
/foo/bar	Match	Match

3.1.12. サービス検出オプションから **autoRegistrationOfImporters** 属性を削除

AutoRegistrationOfImporters 属性はサービス検出オプションから削除されました。

3.1.13. 入力クレデンシャルとして **token** をサポートするように認証プロバイダークラスの認証メソッドを更新

Eclipse Vert.x 4.0.3 より前のリリースでは、AuthenticationProvider.authenticate () メソッドが入力クレデンシャルとして **jwt: someValue** を誤って取得していました。

Eclipse Vert.x 4.0.3 以降、AuthenticationProvider.authenticate() メソッドが更新され、**token: someValue** が入力クレデンシャルとして取得されます。この変更により、JSON API と型指定された API の両方が一貫性を保ち、同じ意味で使用できるようになります。

以下のコードは、Eclipse Vert.x 4.0.3 より前のリリースでの認証メソッドの実装を示しています。

```
new JsonObject().put("jwt", "token...");
```

以下のコードは、Eclipse Vert.x 4.0.3 リリースの認証メソッドの実装を示しています。

```
new JsonObject().put("token", "token...");
```

3.1.14. PEM キーの Get メソッドが **String**ではなく **Buffer** を返す

The **PubSecKeyOptions.getBuffer（）** メソッドは、PEM またはシークレットキーバッファーを返します。Eclipse Vert.x 4.0.2 より前のリリースでは、キーバッファーが保存され、String として返されました 。ただし、シークレットを **Buffer** として保存することが推奨されます。Eclipse Vert.x 4.0.2 以降では、メソッドはキーバッファーを保存し、バッファーとして保存し返します 。この変更により、シークレットのセキュリティーおよび処理が改善されます。

The **PubSecKeyOptions.setBuffer（）** メソッドは **String** 引数を受け入れます。Set メソッドでは、ASCII 以外のシークレット資料を安全に処理するために、バッファーのオーバーロードが追加されました。この変更には、既存のコードへの変更は必要ありません。

3.1.15. Kubernetes サービスインポーターが自動的に登録されなくなる

Eclipse Vert.x 4 以降、**KubernetesServiceImporter** 検出ブリッジは自動的に登録されなくなりました。Maven プロジェクトのクラスパスにブリッジを追加しても、自動的に登録されません。

ServiceDiscovery インスタンスの作成後にブリッジを手動で登録する必要があります。

3.1.16. 非同期操作に future メソッドを使用

Eclipse Vert.x 4 は、非同期操作に future を使用します。すべての callback メソッドには、対応する future メソッドがあります。

Future は非同期操作の作成に使用できます。Future を使用する場合は、エラー処理の方が優れています。したがって、アプリケーションでコールバックと future の組み合わせを使用することが推奨されます。

3.1.17. Jackson Databind ライブラリーの依存関係がない

Eclipse Vert.x 4 では、Jackson Databind はオプションの Maven 依存関係です。この依存関係を使用する場合は、クラスパスに明示的に追加する必要があります。たとえば、オブジェクトマッピング JSON の場合は、依存関係を明示的に追加する必要があります。

3.1.18. 非推奨と削除の処理

Eclipse Vert.x 4 では、新機能が追加されています。以前の機能および機能は Eclipse Vert.x 4 で非推奨または削除されました。アプリケーションを Eclipse Vert.x 4 に移行する前に、非推奨および削除を確認します。

Java コンパイラーは、非推奨の API が使用されたときに警告を生成します。アプリケーションを Eclipse Vert.x 4 に移行する際に、コンパイラーを使用して非推奨のメソッドを確認できます。

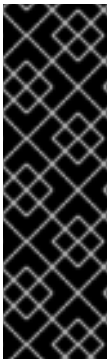
3.1.19. 分散トレースのサポート

Eclipse Vert.x 4 は分散トレースをサポートします。トレースを使用してマイクロサービスを監視し、パフォーマンスの問題を特定することができます。

Eclipse Vert.x 4 は OpenTracing システムと統合します。

以下の Eclipse Vert.x コンポーネントはトレースをログに記録できます。

- HTTP サーバーおよび HTTP クライアント
- Eclipse Vert.x SQL クライアント
- Eclipse Vert.x Kafka クライアント



重要

トレースはテクノロジープレビューとして利用できます。テクノロジープレビュー機能は、Red Hat の実稼働環境でのサービスレベルアグリーメント (SLA) ではサポートされていないため、Red Hat では実稼働環境での使用を推奨していません。これらの機能は、近々発表予定の製品機能をリリースに先駆けてご提供することにより、お客様は機能性をテストし、開発プロセス中にフィードバックをお寄せいただくことができます。

テクノロジープレビュー機能のサポート範囲については、Red Hat カスタマーポータルの「テクノロジープレビュー機能のサポート範囲」を参照してください。

3.1.20. EventBus JavaScript Client の新しい公開場所

Eclipse Vert.x 4 では、EventBus JavaScript クライアント **vertx-web-client.js** は Maven リポジトリーの Red Hat アーティファクトとして公開されません。

クライアントは npm リポジトリーに公開されます。@vertx/eventbus-bridge-client.jsからクライアントにアクセスできます。

3.1.21. OpenShift Maven プラグインを使用した Eclipse Vert.x アプリケーションのデプロイ

OpenShift Maven プラグインを使用して、OpenShift に Eclipse Vert.x アプリケーションをデプロイします。Fabric8 Maven プラグインはサポート対象外になりました。詳細は「Fabric8 Maven Plugin から Eclipse JKube への移行」を参照してください。

3.1.22. OpenShift の Eclipse Vert.x メータリングラベル

メータリングラベルを Eclipse Vert.x Pod に追加し、OpenShift Metering Operator で Red Hat サブスクリプションの詳細を確認できます。

注記

- メータリングラベルは、Operator またはテンプレートがデプロイおよび管理する Pod に追加しないでください。
- OpenShift Container Platform バージョン 4.8 よりも前のメータリング Operator を使用してラベルを Pod に適用できます。バージョン 4.9 以降では、メータリング Operator は直接置き換えずに利用できなくなりました。

Eclipse Vert.x は、以下のメータリングラベルを使用する必要があります。

- **com.company: Red_Hat**

- **rht.prod_name: Red_Hat_Runtimes**
- **rht.prod_ver: 2022-Q1**
- **rht.comp: Vert.x**
- **rht.comp_ver: 4.1.8**
- **rht.subcomp: <leave_blank>**
- **rht.subcomp_t: application**

関連情報

- OpenShift Container Platform でのメータリングの設定および使用

3.1.23. OpenJDK 8 および OpenJDK 11 RHEL 8 Universal Base Images (UBI8) のサポート

Eclipse Vert.x は、RHEL 8 上の Red Hat OpenJDK 8 および Red Hat OpenJDK 11 用の OCI 準拠の ユニバーサル ベースイメージ を使用して、Eclipse Vert.x アプリケーションをビルドして OpenShift にデプロイするためのサポートを導入します。

RHEL 8 OpenJDK Universal Base イメージは、RHEL 8 OpenJDK ビルダーイメージを置き換えます。RHEL 8 OpenJDK ベースイメージは、Eclipse Vert.x との使用に対応しなくなりました。

3.2. 非推奨の機能

本セクションでは、本リリースで非推奨または削除された機能を紹介します。

- **HttpServerOptions**

削除されたメソッド	置き換えメソッド
getMaxWebsocketFrameSize()	**getMaxWebSocketFrameSize()**
setMaxWebsocketFrameSize()	**setMaxWebSocketFrameSize()**
getMaxWebsocketMessageSize()	**getMaxWebSocketMessageSize()**
setMaxWebsocketMessageSize()	**setMaxWebSocketMessageSize()**
getPerFrameWebsocketCompressionSupported()	**getPerFrameWebSocketCompressionSupported()**
setPerFrameWebsocketCompressionSupported()	**setPerFrameWebSocketCompressionSupported()**
getPerMessageWebsocketCompressionSupported()	**getPerMessageWebSocketCompressionSupported()**

削除されたメソッド	置き換えメソッド
setPerMessageWebsocketCompressionSupported()	setPerMessageWebSocketCompressionSupported()
getWebsocketAllowServerNoContext()	getWebSocketAllowServerNoContext()
setWebsocketAllowServerNoContext()	setWebSocketAllowServerNoContext()
getWebsocketCompressionLevel()	getWebSocketCompressionLevel()
setWebsocketCompressionLevel()	setWebSocketCompressionLevel()
getWebsocketPreferredClientNoContext()	getWebSocketPreferredClientNoContext()
setWebsocketPreferredClientNoContext()	setWebSocketPreferredClientNoContext()
getWebsocketSubProtocols()	getWebSocketSubProtocols()
setWebsocketSubProtocols()	setWebSocketSubProtocols()

- Eclipse Vert.x Web

削除された要素	置き換え要素
io.vertx.ext.web.Cookie	io.vertx.core.http.Cookie
io.vertx.ext.web.handler.CookieHandler	io.vertx.core.http.Cookie
io.vertx.ext.web.Locale	io.vertx.ext.web.LanguageHeader
RoutingContext.acceptableLocales()	RoutingContext.acceptableLanguages()
StaticHandler.create(String, ClassLoader)	---
SessionHandler.setAuthProvider(AuthProvider)	SessionHandler.addAuthProvider()

削除された要素	置き換え要素
HandlebarsTemplateEngine.getHandlebars() HandlebarsTemplateEngine.getResolvers() HandlebarsTemplateEngine.setResolvers() JadeTemplateEngine.getJadeConfiguration() ThymeleafTemplateEngine.getThymeleafTemplateEngine() ThymeleafTemplateEngine.setMode()	TemplateEngine.unwrap()

- Messaging

削除されたメソッド	置き換えメソッド
MessageProducer<T>.send(T)	MessageProducer<T>.write(T)
MessageProducer.send(T,Handler)	EventBus.request(String,Object,Handler)

- EventBus

削除されたメソッド	置き換えメソッド
EventBus.send(…, Handler<AsyncResult<Message<T>>>) Message.reply(…, Handler<AsyncResult<Message<T>>>)	replyAndRequest

- ハンドラー

削除されたメソッド	置き換えメソッド
Future<T>.setHandler()	Future<T>.onComplete() Future<T>.onSuccess() Future<T>.onFailure()
HttpClientRequest.connectionHandler()	HttpClient.connectionHandler()

- JSON

削除されたフィールド/メソッド	新しいメソッド
Json.mapper() field	DatabindCodec.mapper()
Json.prettyMapper() field	DatabindCodec.prettyMapper()

削除されたフィールド/メソッド	新しいメソッド
Json.decodeValue(Buffer, TypeReference<T>)	JacksonCodec.decodeValue(Buffer, TypeReference)
Json.decodeValue(String, TypeReference<T>)	JacksonCodec.decodeValue(String, TypeReference)

- JUnit5

非推奨となったメソッド	新しいメソッド
VertxTestContext.succeeding()	VertxTestContext.succeedingThenComplete()
VertxTestContext.failing()	VertxTestContext.failingThenComplete()

- リアクティブエクステンション(Rx)

非推奨となったメソッド	新しいメソッド
WriteStreamSubscriber.onComplete()	WriteStreamSubscriber.onWriteStreamEnd() WriteStreamSubscriber.onWriteStreamError()

- Circuit Breaker

削除されたメソッド	置き換えメソッド
CircuitBreaker.executeCommand()	CircuitBreaker.execute()
CircuitBreaker.executeCommandWithFallback()	CircuitBreaker.executeWithFallback()

- MQTT

削除されたメソッド	置き換えメソッド
MqttWill.willMessage()	MqttWill.getWillMessage()
MqttWill.willTopic()	MqttWill.getWillTopic()
MqttWill.willQos()	MqttWill.getWillQos()
MqttAuth.username()	MqttAuth.getUsername()
MqttAuth.password()	MqttAuth.getPassword()

削除されたメソッド	置き換えメソッド
MqttClientOptions.setKeepAliveTimeSeconds()	MqttClientOptions.setKeepAliveInterval()

- AMQP クライアント

削除されたメソッド	置き換えメソッド
AmqpClient.createReceiver(String address, Handler<AmqpMessage> messageHandler, …)	AmqpClient createReceiver(String address, Handler<AsyncResult<AmqpReceiver>> completionHandler)
AmqpConnection createReceiver(…, Handler<AsyncResult<AmqpReceiver>> completionHandler)	AmqpConnection createReceiver(String address, Handler<AsyncResult<AmqpReceiver>> completionHandler)
AmqpConnection createReceiver(.., Handler<AmqpMessage> messageHandler, Handler<AsyncResult<AmqpReceiver>> completionHandler)	AmqpConnection createReceiver(String address, Handler<AsyncResult<AmqpReceiver>> completionHandler)

- 認証および認可

削除された要素	置き換え要素
OAuth2Options.isUseBasicAuthorizationHeader()	置き換えるメソッドなし
OAuth2Options.setUseBasicAuthorizationHeader()	置き換えるメソッドなし
OAuth2Options.getClientSecretParameterName()	置き換えるメソッドなし
OAuth2Options.setClientSecretParameterName()	置き換えるメソッドなし
OAuth2Auth.createKeycloak()	KeycloakAuth.create(vertx, JsonObject)()
OAuth2Auth.create(Vertx, OAuth2FlowType, OAuth2ClientOptions)()	OAuth2Auth.create(vertx, new OAuth2ClientOptions().setFlow(YOUR_DESIRED_FLOW))

削除された要素	置き換え要素
OAuth2Auth.create(Vertx, OAuth2FlowType)	**OAuth2Auth.create(vertx, new OAuth2ClientOptions().setFlow(YOUR_DESIRED_FLOW))**
User.isAuthorised()	**User.isAuthorized()**
AccessToken.refreshToken()	**AccessToken.opaqueRefreshToken()**
io.vertx.ext.auth.jwt.JWTOptions data object	**io.vertx.ext.jwt.JWTOptions** data object
SecretOptions クラス	**PubSecKeyOptions** class

非推奨となったメソッド	置き換えメソッド
OAuth2Auth.decodeToken()	**AuthProvider.authenticate()**
OAuth2Auth.introspectToken()	**AuthProvider.authenticate()**
OAuth2Auth.getFlowType()	置き換えるメソッドなし
OAuth2Auth.loadJWK()	**OAuth2Auth.jwkSet()**
Oauth2ClientOptions.isUseAuthorizationHeader()	置き換えるメソッドなし

非推奨のクラス	置き換えクラス
AbstractUser	'User.create(JsonObject)' メソッドを使用してユーザーオブジェクトを作成します。
AuthOptions	置き換えクラスなし
JDBCAuthOptions	認証用 **JDBCAuthenticationOptions**、および承認用 **JDBCAuthorizationOptions**
JDBCHashStrategy	置き換えクラスなし
OAuth2RBAC	**AuthorizationProvider**
Oauth2Response	**WebClient クラスの使用が推奨**
KeycloakHelper	置き換えクラスなし

- サービス検出

削除されたメソッド	置き換えメソッド
ServiceDiscovery.create(…, Handler<ServiceDiscovery> completionHandler)	ServiceDiscovery.create(Vertx)
ServiceDiscovery.create(…, Handler<ServiceDiscovery> completionHandler)	ServiceDiscovery.create(Vertx, ServiceDiscoveryOptions)

- Eclipse Vert.x 設定

削除されたメソッド	置き換えメソッド
ConfigRetriever.getConfigAsFuture()	retriever.getConfig()

- MongoDB クライアント

削除されたメソッド	置き換えメソッド
MongoClient.update()	MongoClient.updateCollection()
MongoClient.updateWithOptions()	MongoClient.updateCollectionWithOptions()
MongoClient.replace()	MongoClient.replaceDocuments()
MongoClient.replaceWithOptions()	MongoClient.replaceDocumentsWithOptions()
MongoClient.remove()	MongoClient.removeDocuments()
MongoClient.removeWithOptions()	MongoClient.removeDocumentsWithOptions()
MongoClient.removeOne()	MongoClient.removeDocument()
MongoClient.removeOneWithOptions	MongoClient.removeDocumentsWithOptions()

- 共有データソースのないクライアント

非推奨となったメソッド	新しいメソッド
MongoClient.createNonShared()	MongoClient.create()

非推奨となったメソッド	新しいメソッド
JDBCClient.createNonShared()	wJDBCClient.create()
CassandraClient.createNonShared()	CassandraClient.create()
MailClient.createNonShared()	MailClient.create()

- フックメソッド

削除されたメソッド	新しいメソッド
Context.addCloseHook()	置き換えるメソッドなし
Context.removeCloseHook()	置き換えるメソッドなし

- クローンメソッド

削除されたメソッド	新しいメソッド
KeyCertOptions.clone()	KeyCertOptions.copy()
TrustOptions.clone()	TrustOptions.copy()
SSLEngineOptions.clone()	SSLEngineOptions.copy()

- VertxOptions

削除されたメソッド	新しいメソッド
VertxOptions.equals()	置き換えるメソッドなし
VertxOptions.hashCode()	置き換えるメソッドなし
VertxOptions.fileResolverCachingEnabled()	FileSystemOptions.isFileCachingEnabled()

- プールされたバッファー

削除されたメソッド	新しいメソッド
TCPSSLOptions.isUsePooledBuffers()	置き換えるメソッドなし
TCPSSLOptions.setUsePooledBuffers()	置き換えるメソッドなし

第4章 リリースコンポーネント

4.1. 本リリースで導入されたサポート対象のアーティファクト

本リリースでは、以下のアーティファクトがテクノロジープレビューからフルサポートになりました。

- **vertx-auth-webauthn**
- **vertx-config-vault**
- **vertx-sql-client-templates**
- **vertx-web-api-contract**

4.2. 本リリースで導入されたテクノロジープレビュー機能のアーティファクト

本リリースでは、以下のアーティファクトがテクノロジープレビューとして提供されます。

- **vertx-mssql-client**
  Eclipse Vert.x リアクティブ MSSQL クライアントは、Microsoft SQL サーバーのクライアントです。これは、データベースのスケーラビリティーに役立ち、オーバーヘッドが低い API です。API はリアクティブおよび非ブロッキングであるため、単一のスレッドで複数のデータベース接続を処理できます。

- **vertx-http-proxy**
  Eclipse Vert.x HTTP プロキシーはリバースプロキシーです。このモジュールを使用すると、プロキシーを簡単に作成できます。プロキシーサーバーは、オリジンサーバーからの DNS クエリを動的に解決することもできます。

- **vertx-web-proxy**
  Eclipse Vert.x Web プロキシーを使用すると、Eclipse Vert.x Web ルーターに Eclipse Vert.x HTTP プロキシーをマウントできます。

- **vertx-opentelemetry**
  Open Telemetry のトレースがサポートされます。HTTP およびイベントバストレースに Open Telemetry を使用できます。

- **vertx-web-sstore-infinispan**
  この Eclipse Vert.x Web セッションストアを使用すると、セッションデータを Infinispan キャッシュに保存できます。このモジュールは Infinispan クライアントに実装されているため、スタンドアロンまたはクラスター化された Eclipse Vert.X アプリケーションで使用できます。

- **vertx-auth-webauthn**
  Eclipse Vert.x 認証モジュール **io.vertx.ext.auth.AuthProvider** インターフェースは、2 つの新しいインターフェースに分割されました。

  - **io.vertx.ext.auth.authentication.AuthenticationProvider**
  - **io.vertx.ext.auth.authorization.AuthorizationProvider**
    認証は Eclipse Vert.x 4 の新機能です。以前のリリースでは、ユーザーが **User** オブジェクトでタスクを実行することが許可されているかどうかを確認できました。そのため、プロバイダーはユーザーの認証と承認の両方を行います。

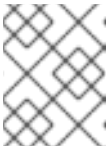
Eclipse Vert.x 4 では、**User** オブジェクトインスタンスは特定の認証プロバイダーに関連付けられていません。そのため、異なるプロバイダーを使用してユーザーを認証および承認できます。

- **vertx-opentracing**
  Eclipse Vert.x 4 は分散トレースをサポートします。トレースを使用してマイクロサービスを監視し、パフォーマンスの問題を特定することができます。

  Eclipse Vert.x 4 は OpenTracing システムと統合します。

  以下の Eclipse Vert.x コンポーネントはトレースをログに記録できます。

  - HTTP サーバーおよび HTTP クライアント
  - Eclipse Vert.x SQL クライアント
  - Eclipse Vert.x Kafka クライアント



注記

Red Hat のテクノロジープレビュー機能のサポート範囲に関する詳細は、「テクノロジプレビュー機能のサポート範囲」を参照してください。

4.3. 本リリースで削除されたアーティファクト

本リリースでは削除されたアーティファクトはありません。

4.4. 本リリースで非推奨となったアーティファクト

本リリースでは、非推奨となったアーティファクトはありません。

第5章 修正された問題

この Eclipse Vert.x リリースには、バージョン 4.1.8 のコミュニティーリリースからのバグ修正がすべて組み込まれています。コミュニティーリリースで解決された問題は Eclipse Vert.x 4.1.8 に一覧表示されます。

5.1. GRAPHQL ビルドに含まれる GOOGLE GUAVA クラス

Eclipse Vert.x 4.0.0 および 4.0.2 リリースでは、**vertx-web-graphql** 依存関係を使用できませんでした。これは、バージョン 16.1.0.redhat-00001 を持つ GraphQL Java の不完全なビルドが使用されたためです。不完全な GraphQL ビルドには、Guava クラスがありませんでした。

この問題は Eclipse Vert.x 4.0.3 リリースで解決されました。このリリースには、Guava クラスの完全なビルドである GraphQL Java 16.1.0.redhat-00002 バージョンが含まれます。これらの Guava クラスは shaded jar になります。

5.2. ECLIPSE VERT.X ビルドで利用可能な VERTX-OPENTRACING

vertx-opentracing 依存関係は、Eclipse Vert.x 4.0.0 でテクノロジープレビュー機能として導入されました。ただし、この依存関係は Eclipse Vert.x 4.0.0 および 4.0.2 リリースでは利用できませんでした。

この問題は Eclipse Vert.x 4.0.3 リリースで解決されました。このリリースには **vertx-opentracing** 依存関係が含まれます。

第6章 既知の問題

6.1. KUBERNETESSERVICEIMPORTER() を ECLIPSE VERT.X REACTIVE EXTENSIONS(RX)に直接登録できない

説明

Eclipse Vert.x の Reactive Extensions(Rx)で **KubernetesServiceImporter()** を直接登録することはできません。

原因

サービスインポーターには生成された RxJava 2 実装がありません。

回避策

以下の例のように、**KubernetesServiceImporter のインスタンスを作成し、{ @link io.vertx.reactivex.servicediscovery.spi.ServiceImporter}** でカプセル化する必要があります。

```
{@link
examples.RxServiceDiscoveryExamples#register(io.vertx.reactivex.servicediscovery.ServiceDiscovery)}
```

以下の例は、Eclipse Vert.x Reactive Extensions(Rx)で **KubernetesServiceImporter()** を登録する方法を示しています。

```
ServiceDiscovery discovery = ServiceDiscovery.create(vertx);
discovery.getDelegate().registerServiceImporter(new KubernetesServiceImporter(), new
JsonObject());
```

6.2. IBM Z および IBM POWER SYSTEMS では、RED HAT AMQ STREAMS イメージが利用できない

Red Hat AMQ Streams Operator イメージおよび Kafka イメージは、IBM Z および IBM Power Systems では使用できません。イメージは利用できないため、vertx **-kafka-client** モジュールは IBM Z および IBM Power Systems の AMQ Streams と動作することが認定されていません。

6.3. TLS プロトコルバージョンが一致しないため、RHEL 8 ベースのデータベースアプリケーションと RHEL 7 ベースの MYSQL 5.7 データベース間の接続が失敗する

説明

RHEL 8 ベースの OpenJDK ビルダーイメージで構築されたアプリケーションコンテナーと、RHEL 7 ベースの MySQL 5.7 コンテナーイメージに構築されたデータベースコンテナーとの間で、OpenSSL を使用して TLS でセキュア化された接続を開くと、ランタイム時に **javax.net.ssl.SSLHandshakeException** によって接続に失敗します。詳細は、JIRA の問題 を参照してください。

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

原因

この問題は、RHEL 7 から RHEL 8 のサポートされる最新の TLS プロトコルバージョンが異なるために発生します。RHEL 7 の TLS 実装は、TLS プロトコルバージョン 1.0（非推奨）、1.1、および 1.2 に対応しています。RHEL 8 の TLS 実装は、TLS プロトコルバージョン 1.3 にも対応しています。これは、RHEL 8 ベースのビルダーイメージで使用されるデフォルトの TLS バージョンです。この不一致により、TLS ハンドシェイクが中断している間にアプリケーションコンポーネント間で TLS プロトコルバージョンの不一致が発生し、アプリケーションとデータベースコンテナー間の接続が失敗します。

回避策

上記の問題を防ぐには、データベース接続文字列の両方のオペレーティングシステムバージョンでサポートされる TLS プロトコルバージョンを手動で指定します。例:

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

6.4. FALSE CONNECTION がアプリケーションエンドポイントを呼び出す際にピアエラーメッセージによってリセットされる

Curl **ツールまたは** Java HTTP クライアントのいずれかを使用して Eclipse Vert.x アプリケーションのエンドポイントで HTTP 要求を行うと、リクエストごとに以下のエラーがログに表示されます。

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

この動作は、Netty アプリケーションフレームワークと、OpenShift によって使用される HAProxy ロードバランサーの対話によって生じます。このエラーは、HAProxy が閉じずに既存の HTTP 接続が再使用されるために発生します。エラーメッセージがログに記録されても、エラー状態は発生しません。HTTP リクエストが正しく処理され、アプリケーションは予想通りに応答します。

第7章 本リリースに関連するアドバイザリー

本リリースに含まれる拡張機能、バグ修正、および CVE 修正を文書化するために、以下のアドバイザリーが発行されています。

- RHSA-2022:0083