



# Red Hat Data Grid 8.1

## 『Hot Rod C++ Client Guide』

Hot Rod C++ クライアントの設定および使用



## Red Hat Data Grid 8.1 『Hot Rod C++ Client Guide』

---

Hot Rod C++ クライアントの設定および使用

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

Copyright © 2021 | You need to change the HOLDER entity in the en-US/Hot\_Rod\_Cpp\_Client\_Guide.ent file |.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux<sup>®</sup> is the registered trademark of Linus Torvalds in the United States and other countries.

Java<sup>®</sup> is a registered trademark of Oracle and/or its affiliates.

XFS<sup>®</sup> is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL<sup>®</sup> is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js<sup>®</sup> is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack<sup>®</sup> Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

## 概要

ホット Rod C++ クライアントにより、C++ ランタイムアプリケーションはリモート Data Grid クラスタに接続し、対話することができます。

---

## 目次

<b>第1章 RED HAT DATA GRID .....</b>	<b>3</b>
1.1. DATA GRID ドキュメント .....	3
1.2. DATA GRID のダウンロード .....	3
1.3. 多様性を受け入れるオープンソースの強化 .....	3
<b>第2章 HOT ROD C++ クライアントのインストール .....</b>	<b>4</b>
2.1. C++ コンパイラーの要件 .....	4
2.2. RED HAT ENTERPRISE LINUX(RHEL)への HOT ROD C++ クライアントのインストール .....	4
2.3. MICROSOFT WINDOWS への HOT ROD C++ クライアントのインストール .....	4
<b>第3章 PROTOBUF スキーマのコンパイル .....</b>	<b>6</b>
3.1. RED HAT ENTERPRISE LINUX(RHEL)での PROTOBUF スキーマのコンパイル .....	6
3.2. MICROSOFT WINDOWS での PROTOBUF スキーマのコンパイル .....	6
<b>第4章 HOT ROD C++ クライアントの設定 .....</b>	<b>8</b>
4.1. 設定およびリモートキャッシュマネージャー API .....	8



# 第1章 RED HAT DATA GRID

Data Grid は、高パフォーマンスの分散型インメモリーデータストアです。

## スキーマのないデータ構造

異なるオブジェクトをキーと値のペアとして保存する柔軟性。

## グリッドベースのデータストレージ

クラスターにデータを分散し、複製するように設計されています。

## Elastic scaling (Elastic のスケーリング)

サービスの中断なしで要求を満たすようにノード数を動的に調整します。

## データ相互運用性

異なるエンドポイントからグリッドのデータの保存、取得、クエリーを行います。

## 1.1. DATA GRID ドキュメント

Data Grid のドキュメントは、Red Hat カスタマーポータルから入手できます。

- [Data Grid 8.1 のドキュメント](#)
- [Data Grid 8.1 コンポーネントの詳細](#)
- [Data Grid 8.1 でサポートされる構成](#)
- [Data Grid 8 の機能サポート](#)
- [Data Grid で非推奨となった機能および機能](#)

## 1.2. DATA GRID のダウンロード

Red Hat カスタマーポータルで [Data Grid の Software Downloads](#) にアクセスします。



### 注記

Data Grid ソフトウェアにアクセスしてダウンロードするには、Red Hat アカウントが必要です。

## 1.3. 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。これは大規模な取り組みであるため、これらの変更は今後の複数のリリースで段階的に実施されます。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2章 HOT ROD C++ クライアントのインストール

ホストシステムに Hot Rod C++ クライアントを動的ライブラリーとしてインストールします。

### 2.1. C++ コンパイラーの要件

オペレーティングシステム	必要なコンパイラー
Red Hat Enterprise Linux(RHEL)7、64 ビット	C++ 11 コンパイラー(GCC 4.8.1)
RHEL 8、64 ビット	C++ 11 コンパイラー(GCC 4.8.1)
Microsoft Windows 7 x64	C 11 コンパイラー (Visual Studio 2015、Microsoft Visual C 2013 Redistributable Package for the x64 プラットフォーム)

### 2.2. RED HAT ENTERPRISE LINUX(RHEL)への HOT ROD C++ クライアントのインストール

Data Grid は、RHEL 用の Hot Rod C++ クライアントの RPM ディストリビューションを提供します。

#### 手順

1. RHEL で Hot Rod C++ クライアントのリポジトリを有効にします。

RHEL バージョン	リポジトリ
RHEL 7	<b>jb-datagrid-8.1-for-rhel-7-server-rpms</b>
RHEL 8	<b>jb-datagrid-8.1-for-rhel-8-x86_64-rpms</b>

2. Hot Rod C++ クライアントをインストールします。

```
# yum install jdgcpp-client
```

#### 関連情報

- [Red Hat Subscription Management \(Red Hat ナレッジベース\)](#) を使用したリポジトリの有効化または無効化
- [Red Hat パッケージブラウザー](#)

### 2.3. MICROSOFT WINDOWS への HOT ROD C++ クライアントのインストール

Data Grid は、Windows へのインストール用に Hot Rod C++ クライアントのアーカイブされたバージョンを提供します。

## 手順

1. [Data Grid Software Downloads](#) から Hot Rod C++ クライアントの ZIP アーカイブをダウンロードします。
2. ZIP アーカイブをファイルシステムに展開します。

## 第3章 PROTOBUF スキーマのコンパイル

Data Grid は ProtoStream API を使用して、データを Protobuf エンコードエントリーとして保存します。

protobuf は言語に依存しない形式で、Hot Rod と REST エンドポイントの両方を使用して、クライアントがリモートキャッシュにエントリーを作成および取得できるようにします。

### 3.1. RED HAT ENTERPRISE LINUX(RHEL)での PROTOBUF スキーマのコンパイル

データを Data Grid に記述するために、compile Protobuf スキーマ **.proto** ファイルを C++ ヘッダーおよびソースファイルに記述します。

#### 前提条件

- Protobuf ライブラリーおよび **protobuf-devel** パッケージをインストールします。

```
# yum install protobuf
# yum install protobuf-devel
```

#### 手順

1. **LD\_LIBRARY\_PATH** 環境変数が設定されていない場合は、その環境変数を設定します。

```
# export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/lib64
```

2. 必要に応じて Hot Rod C++ クライアントの compile Protobuf スキーマ。

```
# /bin/protoc --cpp_out dllexport_decl=HR_PROTO_EXPORT:/path/to/output/ $FILE
```

**HR\_PROTO luatex** は、Hot Rod C++ クライアントが Protobuf スキーマをコンパイルしたときに拡張するマクロです。

3. クエリーを使用する予定がある場合は、Protobuf スキーマを Data Grid に登録します。

#### 関連情報

- [Protobuf スキーマの登録](#)

### 3.2. MICROSOFT WINDOWS での PROTOBUF スキーマのコンパイル

データを Data Grid に記述するために、compile Protobuf スキーマ **.proto** ファイルを C++ ヘッダーおよびソースファイルに記述します。

#### 手順

1. Hot Rod C++ クライアントのインストールディレクトリーに対してコマンドプロンプトを開きます。
2. 必要に応じて Hot Rod C++ クライアントの compile Protobuf スキーマ。

```
bin\protoc --cpp_out dllexport_decl=HR_PROTO_EXPORT:path\to\output\ $FILE
```

**HR\_PROTO luatex** は、Sot Rod C++ クライアントが Protobuf スキーマをコンパイルしたときに拡張するマクロです。

3. クエリーを使用する予定がある場合は、Protobuf スキーマを Data Grid に登録します。

## 関連情報

- [Protobuf スキーマの登録](#)

## 第4章 HOT ROD C++ クライアントの設定

ホット Rod C++ クライアントは、RemoteCache **API** 経由でリモート Data Grid クラスターと対話します。

### 4.1. 設定およびリモートキャッシュマネージャー API

ConfigurationBuilder API を **使用** して Hot Rod C++ クライアント接続および RemoteCacheManager API を設定して リモートキャッシュを取得し、設定します。

#### 設定ビルダー

```
#include "infinispan/hotrod/ConfigurationBuilder.h"
#include "infinispan/hotrod/RemoteCacheManager.h"
#include <infinispan/hotrod/RemoteCache.h>
#include <iostream>
int main () {
    ConfigurationBuilder builder;
    // Configure a cache manager to connect with Hot Rod version 2.8
    builder.protocolVersion(Configuration::PROTOCOL_VERSION_28);
    // Connect to a server at localhost with the default port.
    builder.addServer().host("127.0.0.1").port(11222);
    // Create and start a RemoteCacheManager to interact with caches.
    RemoteCacheManager cacheManager(builder.build(), false);
    cacheManager.start();
    ...
}
```

#### クロスサイトレプリケーション

```
ConfigurationBuilder builder;
builder.addServer().host("127.0.0.1").port(11222);
// Configure a remote cluster and node when using cross-site replication.
builder.addCluster("NYC").addClusterNode("192.0.2.0", 11322);
```

#### ニアキャッシュ

```
ConfigurationBuilder builder;
builder.addServer().host("127.0.0.1").port(11222);
// Enable near-caching for the client.
builder.nearCache().mode(NearCacheMode::INVALIDATED).maxEntries(4);
```

#### 関連情報

- [Hot Rod C++ クライアント API](#)