Abstract

This document provides feature support information for Data Grid 7.3.
Table of Contents

1. DATA GRID .................................................................................................................... 2
   1.1. Data Grid Usage Modes ......................................................................................... 2

2. SUPPORTED DATA GRID FEATURES ............................................................................. 2
   2.1. Supported Features in Data Grid 7.3 ................................................................. 2
   2.2. Features in Remote Client–Server Mode (or Server Mode) .................................. 4
   2.3. Features in Library Mode (or Embedded Mode) .................................................. 5
   2.4. Technology Previews ............................................................................................ 5
   2.5. Deprecated Features ............................................................................................. 5

3. SUPPORTED HOT ROD FEATURES ............................................................................... 5
1. DATA GRID

Data Grid is a high-performance, distributed in-memory data store.

**Schemaless data structure**
- Flexibility to store different objects as key-value pairs.

**Grid-based data storage**
- Designed to distribute and replicate data across clusters.

**Elastic scaling**
- Dynamically adjust the number of nodes to meet demand without service disruption.

**Data interoperability**
- Store, retrieve, and query data in the grid from different endpoints.

1.1. Data Grid Usage Modes

Red Hat Data Grid supports two usage modes:

**Remote Client-Server Mode (or Server Mode)**
- Run Data Grid as managed and distributed cluster that you can scale to meet demand. Client applications remotely access Data Grid through endpoints such as Hot Rod or REST.

**Library Mode (or Embedded Mode)**
- Build and deploy Data Grid as a single node in your application runtime, where the application process and cache share the same JVM memory. In Library Mode, Data Grid can remotely access nodes hosted in other JVMs across your environment.

Data Grid supports Library Mode in application containers such as JBoss Enterprise Application Platform (EAP) or as standalone applications. Find out more about supported application containers in the Supported Configurations.

2. SUPPORTED DATA GRID FEATURES

2.1. Supported Features in Data Grid 7.3

- File Cache Store and Loading
- JDBC Cache Store and Loading
- RocksDB Cache Store and Loading (Replaces LevelDB)
- Cache Passivation
- Remote Cache Store
- Cluster Cache Store
- Asynchronous Store
- Cluster Configuration Using UDP
- Cluster Configuration Using TCP
- Mortal and Immortal Data
- Eviction Strategy
- Expiration
- Unscheduled Write-behind Cache Store
- Write-through Cache Store
- Local Cache Mode (local)
- Clustering Mode (replicated)
- Clustering Mode (invalidation)
- Clustering Mode (distribution)
- Asynchronous Clustering Modes
- Marshalling
- JMX Management
- Cross-Datacenter Replication and State Transfer
- Role-based Access Control
- Node Authentication and Authorization
- Encrypted Communication Within the Cluster
- Per Invocation Flags
- Customizable Network Partition Handling
- Spring Integration
- Querying (by values)
- Continuous Queries
- Clustered Listeners and Notifications for Cache Events
- Near Caching
- JSR-107 Support
- CDI
- Asynchronous API
- Distributed Streams
  Distributed Streams are available in Remote Client-Server Mode via Remote Task Execution.
Off Heap Cache Storage

Ickle Query Language

EAP Integration

Cassandra Cache Store and Loading

Command Line Interface (CLI)

Apache Camel Component for JBoss Fuse

NOTE

Red Hat Fuse 7.3 and later provide a camel-infinispan component that replaces the Data Grid Apache Camel component, jboss-datagrid-7.3-camel-library.

You should download and use the camel-infinispan component that Red Hat Fuse provides instead of jboss-datagrid-7.3-camel-library.

Refer to the Infinispan Component section in the Apache Camel Component Reference document for more information. You can find the Apache Camel Component Reference in the Red Hat Fuse documentation.

2.2. Features in Remote Client-Server Mode (or Server Mode)

- Deploy custom cache store to RHDG Server
- Datasources with JDBC Cache Stores
- REST Interface
- Hot Rod Java client
- Hot Rod C++ Client
- Hot Rod .NET Client
- Hot Rod Node.js Client
- Data Compatibility Between Client-server Protocols
- Data Compatibility Between Hot Rod Java and C++ Client
- Rolling Upgrades for Hot Rod Cluster
- Controlled Shutdown and Restart of Cluster
- Hot Rod Client as a JBoss EAP Module
- Externalizing HTTP sessions from JBoss EAP 7 to remote RHDG cluster
- Externalizing HTTP sessions from JBoss Web Server to remote RHDG cluster
- Remote Task Execution
- Apache Spark with Scala 2.11
2.3. Features in Library Mode (or Embedded Mode)

- READ_COMMITTED and REPEATABLE_READ Isolation Modes
- Lazy Deserialization
- Configuration format (infinispan.xml) is shared between Library Mode and Remote Client-Server Mode.
- Grouping API
- Java Transactional API (JTA) Support and Configuration
- Java Transactional API (JTA) Deadlock Detection
- Transaction Recovery
- Transaction and Batching
- Key Affinity
- RHDG as Lucene Directory
- JPA Cache Store
- Data Grid as a JBoss EAP Module

**NOTE**

Data Grid now provides Library and Java Hot Rod Client modules for EAP as a single package.

2.4. Technology Previews

Technology Preview Features in Data Grid 7.3

2.5. Deprecated Features

Features and Functionality Deprecated in Data Grid 7.3

3. SUPPORTED HOT ROD FEATURES

Table 1. Hot Rod Client Feature Support by Language

<table>
<thead>
<tr>
<th>Feature</th>
<th>Java</th>
<th>C++</th>
<th>C#</th>
<th>Node.js</th>
</tr>
</thead>
</table>


<table>
<thead>
<tr>
<th>Feature</th>
<th>Java</th>
<th>C++</th>
<th>C#</th>
<th>Node.js</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transactions</td>
<td>Supported with <strong>FULL_XA</strong> and <strong>NON_XA</strong> transaction modes.</td>
<td>Supported with <strong>NON_XA</strong> transactions only.</td>
<td>Supported with <strong>NON_XA</strong> transactions only.</td>
<td>N/A</td>
</tr>
<tr>
<td>Clustered Counters</td>
<td>Supported</td>
<td>Tech Preview</td>
<td>Tech Preview</td>
<td>N/A</td>
</tr>
<tr>
<td>Statistics via JMX</td>
<td>Supported</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Cross-site Failover</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Authentication</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>TLS-based Encryption</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Server Name Indication (SNI)</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Asynchronous API</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
<td>N/A</td>
</tr>
<tr>
<td>Near Caching</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
<td>N/A</td>
</tr>
<tr>
<td>Continuous Queries</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
<td>N/A</td>
</tr>
<tr>
<td>Remote Event Listeners</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Remote Querying</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
<td>N/A</td>
</tr>
<tr>
<td>Remote Execution</td>
<td>Supported</td>
<td>Supported</td>
<td>Supported</td>
<td>N/A</td>
</tr>
</tbody>
</table>