Spring Cloud Data Flow for VMware GemFire Documentation

Spring Cloud Data Flow for VMware GemFire 1.0

vmware by Broadcom

You can find the most up-to-date technical documentation on the VMware by Broadcom website at:

https://docs.vmware.com/

VMware by Broadcom
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

Copyright © 2024 Broadcom. All Rights Reserved. The term “Broadcom” refers to Broadcom Inc. and/or its subsidiaries. For more information, go to https://www.broadcom.com. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies. Copyright and trademark information.

Contents

Spring Cloud Data Flow for VMware GemFire Documentation

Spring Cloud Dataflow for VMware Tanzu GemFire is a project that defines integration with the Spring Cloud Stream and Spring Cloud Stream Applications projects.

The published artifacts are:

- gemfire-source-rabbit
- gemfire-sink-rabbit
- gemfire-source-kafka
- gemfire-sink-kafka

These artifacts are then “installed” into a Spring Cloud Dataflow Server in order to receive or send data from VMware Tanzu GemFire instances.

Release Notes

This topic contains the release notes for Spring Cloud Dataflow for VMware GemFire.

1.0.0

- Initial release of Spring Cloud Stream AppsFor VMware Tanzu GemFire, for Spring Cloud Dataflow Server `2.10.x` and `2.11.1` and VMware GemFire `9.15.x` and `10.0.x`.
    - This includes bindings for RabbitMQ and Kafka.
    - gemfire-source-rabbit
    - gemfire-sink-rabbit
    - gemfire-source-kafka
    - gemfire-sink-kafka

Compatibility and Versions

This topic list Spring Cloud Dataflow for VMware GemFire compatibility and versions.

Compatibility

Spring Cloud Stream App Artifact	Latest Versions	Compatible GemFire Versions	Compatible Spring Cloud Dataflow Server	Compatible Spring Boot Versions
gemfire-source-rabbit	1.0.0	9.15+, 10.0+	2.10+ , 2.11+	2.7.x
gemfire-source-kafka	1.0.0	9.15+, 10.0+	2.10+, 2.11+	2.7.x
gemfire-sink-rabbit	1.0.0	9.15+, 10.0+	2.10+, 2.11+	2.7.x
gemfire-sink-kafka	1.0.0	9.15+, 10.0+	2.10+, 2.11+	2.7.x

Getting Started

This topic explains how to download Spring Cloud Dataflow for VMware GemFire libraries to a project.

The Spring Cloud Dataflow for VMware GemFire libraries are available from Docker Hub or the Pivotal Commercial Maven Repository. Access to the Pivotal Commercial Maven Repository requires a one-time registration step to create an account.

Pull Images from Docker Hub

The images can be retrieved from Docker Hub simply by running `docker pull gemfire/<image name>:<version>` with the desired image

Access the Commercial Maven Repository

1. In a browser, navigate to the Pivotal Commercial Maven Repository.
2. Click the **Create Account** link.
3. Complete the information in the registration page.
4. Click **Register**.
5. After registering, you will receive a confirmation email. Follow the instruction in this email to activate your account.
6. After account activation, log in to the Pivotal Commercial Maven Repository to access the configuration information found in gemfire-release-repo.

Installing Into Spring Cloud Data Flow

This project provides implementations for VMware Tanzu GemFire with Kafka and RabbitMQ bindings.

For detailed information on installing each implementation into a Spring Cloud Data Flow server, refer to the following:

- GemFire Source RabbitMQ
- GemFire Sink RabbitMQ
- GemFire Source Kafka
- GemFire Sink Kafka

GemFire Source Rabbit

Getting started:

There are two methods to deploy the artifacts into the Spring Cloud Dataflow Server: using the images from Docker Hub or the artifacts downloaded from the commercial Maven repository.

Docker Hub Images

1. Log in to your Spring Cloud Dataflow Server.
2. Add the Application using the **Add Application** button.
3. Select the first option **Register one or more applications**.
4. Add a name in the `Name` field - for example `gemfire`, or `gemfire-source-rabbit`.
5. For `Type`, select `source`.
6. For `Spring Boot version`, select the appropriate version.
7. For `URI`, enter `docker://docker.io/gemfire/gemfire-source-rabbit:1.0.0`
8. Click **Import Application** to import the GemFire Source for Rabbit Stream Application.

Commercial Maven Repository Artifacts

In order to use this Spring Cloud Stream App you need to deploy the artifacts into the SCFD Server use the following steps:

1. Follow the Getting Started guide to get access to the Commercial Maven Repository.
2. Download the GemFire Source Rabbit artifact

   ```
   wget https://commercial-repo.pivotal.io/data3/gemfire-release-repo/gemfire/com/vmware/gemfire/spring/cloud/stream/app/gemfire-source-rabbit/1.0.0/gemfire-source-rabbit-1.0.0.jar --user {commercialMavenRepoUsername} --ask-password
   ```

3. Log in to your Spring Cloud Dataflow Server.
4. Add the Application using the **Add Application** button.
5. Select the third option, **Import application coordinates from properties file**.
6. Add the `source` properties, replacing `{artifactFileName}` with the location of the downloaded artifacts from step 2. You can change the application name (`gemfire`) as needed.

   ```
   source.gemfire-source=file://{artifactFileName}
   ```

7. Click **Import Application** to import the GemFire Source for Rabbit Stream Application.

Properties:

gemfire.client

Property Name	Description	Type	Defaults
pdx-read-serialized	Deserialize the GemFire objects into PdxInstance instead of the domain class.	Boolean	false

gemfire.pool

Property Name	Description	Type	Defaults
connect-type	Specifies connection type: 'server' or 'locator'.	ConnectType	
host-addresses	Specifies one or more GemFire locator or server addresses formatted as [host]:[port].	InetSocketAddress[]	
subscription-enabled	Set to true to enable subscriptions for the client pool. Required to sync updates to the client cache.	Boolean	false

gemfire.region

Property Name	Description	Type	Defaults
region-name	The region name.	String	

gemfire.security

Property Name	Description	Type	Defaults
password	The cache password.	String	
username	The cache username.	String	

gemfire.security.ssl

Property Name	Description	Type	Defaults
ciphers	Configures the SSL ciphers used for secure Socket connections as an array of valid cipher names.	String	any
keystore-type	Identifies the type of Keystore used for SSL communications (e.g. JKS, PKCS11, etc.).	String	JKS
keystore-uri	Location of the pre-created Keystore URI to be used for connecting to the GemFire cluster.	Resource	
ssl-keystore-password	Password for accessing the keys truststore.	String	
ssl-truststore-password	Password for accessing the trust store.	String	
truststore-type	Identifies the type of truststore used for SSL communications (e.g. JKS, PKCS11, etc.).	String	JKS

truststore-uri	Location of the pre-created truststore URI to be used for connecting to the GemFire cluster.	Resource	
user-home-directory	Local directory to cache the truststore and keystore files downloaded form the truststoreUri and keystoreUri locations.	String	user.home

gemfire.supplier

Property Name	Description	Type	Defaults
event-expression	SpEL expression to extract data from an {@link org.apache.geode.cache.EntryEvent} or {@link org.apache.geode.cache.query.CqEvent}.	Expression	
query	An OQL query. This will enable continuous query if provided.	String	

GemFire Sink Rabbit**

Getting started:

There are two methods to deploy the artifacts into the Spring Cloud Dataflow Server: using the images from Docker Hub or the artifacts downloaded from the commercial Maven repository.

Docker Hub Images

1. Log in to your Spring Cloud Dataflow Server.
2. Add the Application using the **Add Application** button.
3. Select the first option **Register one or more applications**.
4. Add a name in the `Name` field - for example `gemfire`, or `gemfire-sink-rabbit`.
5. For `Type`, select `sink`.
6. For `Spring Boot version`, select the appropriate version.
7. For `URI`, enter `docker://docker.io/gemfire/gemfire-sink-rabbit:1.0.0`
8. Click **Import Application** to import the GemFire Sink for Rabbit Stream Application.

Commercial Maven Repository Artifacts

In order to use this Spring Cloud Stream App you need to deploy the artifacts into the SCFD Server use the following steps:

1. Follow the Getting Started guide to get access to the Commercial Maven Repository.
2. Download the GemFire Sink Rabbit artifact

   ```
   wget https://commercial-repo.pivotal.io/data3/gemfire-release-repo/gemfire/com/vmware/gemfire/spring/cloud/stream/app/gemfire-sink-rabbit/1.0.0/gemfire-sink-rabbit-1.0.0.jar --user {commercialMavenRepoUsername} --ask-password
   ```

3. Log in to your Spring Cloud Dataflow Server.

4. Add the Application using the **Add Application** button.
5. Select the third option, **Import application coordinates from properties file**.
6. Add the `sink` properties, replacing `{artifactFileName}` with the location of the downloaded artifacts from step 2. You can change the application name (`gemfire`) as needed.

```
sink.gemfire-sink=file://{artifactFileName}
```

7. Click **Import Application** to import the GemFire Sink for Rabbit Stream Application.

Properties:

gemfire.consumer

Property Name	Description	Type	Defaults
json	Indicates if the GemFire region stores json objects as PdxInstance.	Boolean	false
key-expression	SpEL expression to use as a cache key.	String	

gemfire.pool

Property Name	Description	Type	Defaults
connect-type	Specifies connection type: 'server' or 'locator'.	ConnectType	
host-addresses	Specifies one or more GemFire locator or server addresses formatted as [host]:[port].	InetSocketAddress[]	
subscription-enabled	Set to true to enable subscriptions for the client pool. Required to sync updates to the client cache.	Boolean	false

gemfire.region

Property Name	Description	Type	Defaults
region-name	The region name.	String	

gemfire.security

Property Name	Description	Type	Defaults
password	The cache password.	String	
username	The cache username.	String	

gemfire.security.ssl

Property Name	Description	Type	Defaults
ciphers	Configures the SSL ciphers used for secure Socket connections as an array of valid cipher names.	String	any

keystore-type	Identifies the type of Keystore used for SSL communications (e.g. JKS, PKCS11, etc.).	String	JKS
keystore-uri	Location of the pre-created Keystore URI to be used for connecting to the GemFire cluster.	Resource	
ssl-keystore-password	Password for accessing the keys truststore.	String	
ssl-truststore-password	Password for accessing the trust store.	String	
truststore-type	Identifies the type of truststore used for SSL communications (e.g. JKS, PKCS11, etc.).	String	JKS
truststore-uri	Location of the pre-created truststore URI to be used for connecting to the GemFire cluster.	Resource	
user-home-directory	Local directory to cache the truststore and keystore files downloaded form the truststoreUri and keystoreUri locations.	String	user.home

GemFire Source Kafka

Getting started:

There are two methods to deploy the artifacts into the Spring Cloud Dataflow Server: using the images from Docker Hub or the artifacts downloaded from the commercial Maven repository.

Docker Hub Images

1. Log in to your Spring Cloud Dataflow Server.
2. Add the Application using the **Add Application** button.
3. Select the first option **Register one or more applications**.
4. Add a name in the `Name` field - for example `gemfire`, or `gemfire-source-kafka`.
5. For `Type`, select `source`.
6. For `Spring Boot version`, select the appropriate version.
7. For `URI`, enter `docker://docker.io/gemfire/gemfire-source-kafka:1.0.0`
8. Click **Import Application** to import the GemFire Source for Kafka Stream Application.

Commercial Maven Repository Artifacts

In order to use this Spring Cloud Stream App you need to deploy the artifacts into the SCFD Server use the following steps:

1. Follow the Getting Started guide to get access to the Commercial Maven Repository.
2. Download the GemFire Source Kafka artifact

```
wget https://commercial-repo.pivotal.io/data3/gemfire-release-repo/gemfire/com/vmware/gemfire/spring/cloud/stream/app/gemfire-source-
```

`kafka/1.0.0/gemfire-source-kafka-1.0.0.jar --user {commercialMavenRepoUsername} --ask-password`

3. Log in to your Spring Cloud Dataflow Server.
4. Add the Application using the **Add Application** button.
5. Select the third option, **Import application coordinates from properties file**.
6. Add the `source` properties, replacing `{artifactFileName}` with the location of the downloaded artifacts from step 2. You can change the application name (`gemfire`) as needed.

```
source.gemfire-source=file://{artifactFileName}
```

7. Click **Import Application** to import the GemFire Source for Kafka Stream Application.

Properties:

gemfire.client

Property Name	Description	Type	Defaults
pdx-read-serialized	Deserialize the GemFire objects into PdxInstance instead of the domain class.	Boolean	false

gemfire.pool

Property Name	Description	Type	Defaults
connect-type	Specifies connection type: 'server' or 'locator'.	ConnectType	
host-addresses	Specifies one or more GemFire locator or server addresses formatted as [host]:[port].	InetSocketAddress[]	
subscription-enabled	Set to true to enable subscriptions for the client pool. Required to sync updates to the client cache.	Boolean	false

gemfire.region

Property Name	Description	Type	Defaults
region-name	The region name.	String	

gemfire.security

Property Name	Description	Type	Defaults
password	The cache password.	String	
username	The cache username.	String	

gemfire.security.ssl

Property Name	Description	Type	Defaults
ciphers	Configures the SSL ciphers used for secure Socket connections as an array of valid cipher names.	String	any
keystore-type	Identifies the type of Keystore used for SSL communications (e.g. JKS, PKCS11, etc.).	String	JKS
keystore-uri	Location of the pre-created Keystore URI to be used for connecting to the GemFire cluster.	Resource	
ssl-keystore-password	Password for accessing the keys truststore.	String	
ssl-truststore-password	Password for accessing the trust store.	String	
truststore-type	Identifies the type of truststore used for SSL communications (e.g. JKS, PKCS11, etc.).	String	JKS
truststore-uri	Location of the pre-created truststore URI to be used for connecting to the GemFire cluster.	Resource	
user-home-directory	Local directory to cache the truststore and keystore files downloaded form the truststoreUri and keystoreUri locations.	String	user.home

gemfire.supplier

Property Name	Description	Type	Defaults
event-expression	SpEL expression to extract data from an {@link org.apache.geode.cache.EntryEvent} or {@link org.apache.geode.cache.query.CqEvent}.	Expression	
query	An OQL query. This will enable continuous query if provided.	String	

GemFire Sink Kafka

Getting started:

There are two methods to deploy the artifacts into the Spring Cloud Dataflow Server: using the images from Docker Hub or the artifacts downloaded from the commercial Maven repository.

Docker Hub Images

1. Log in to your Spring Cloud Dataflow Server.
2. Add the Application using the **Add Application** button.
3. Select the first option **Register one or more applications**.
4. Add a name in the `Name` field - for example `gemfire`, or `gemfire-sink-kafka`.
5. For `Type`, select `sink`.
6. For `Spring Boot version`, select the appropriate version.

7. For `URI`, enter `docker://docker.io/gemfire/gemfire-sink-kafka:1.0.0`
8. Click **Import Application** to import the GemFire Sink for Kafka Stream Application.

Commercial Maven Repository Artifacts

In order to use this Spring Cloud Stream App you need to deploy the artifacts into the SCFD Server use the following steps:

1. Follow the Getting Started guide to get access to the Commercial Maven Repository.
2. Download the GemFire Sink Kafka artifact

   ```
   wget https://commercial-repo.pivotal.io/data3/gemfire-release-repo/gemfire/com/vmware/gemfire/spring/cloud/stream/app/gemfire-sink-kafka/1.0.0/gemfire-sink-kafka-1.0.0.jar --user {commercialMavenRepoUsername} --ask-password
   ```

3. Log in to your Spring Cloud Dataflow Server.
4. Add the Application using the **Add Application** button.
5. Select the third option, **Import application coordinates from properties file**.
6. Add the `sink` properties, replacing `{artifactFileName}` with the location of the downloaded artifacts from step 2. You can change the application name (`gemfire`) as needed.

   ```
   sink.gemfire-sink=file://{artifactFileName}
   ```

7. Click **Import Application** to import the GemFire Sink for Kafka Stream Application.

Properties:

gemfire.consumer

Property Name	Description	Type	Defaults
json	Indicates if the GemFire region stores json objects as PdxInstance.	Boolean	false
key-expression	SpEL expression to use as a cache key.	String	

gemfire.pool

Property Name	Description	Type	Defaults
connect-type	Specifies connection type: 'server' or 'locator'.	ConnectType	
host-addresses	Specifies one or more GemFire locator or server addresses formatted as [host]:[port].	InetSocketAddress[]	
subscription-enabled	Set to true to enable subscriptions for the client pool. Required to sync updates to the client cache.	Boolean	false

gemfire.region

Property Name	Description	Type	Defaults
region-name	The region name.	String	

gemfire.security

Property Name	Description	Type	Defaults
password	The cache password.	String	
username	The cache username.	String	

gemfire.security.ssl

Property Name	Description	Type	Defaults
ciphers	Configures the SSL ciphers used for secure Socket connections as an array of valid cipher names.	String	any
keystore-type	Identifies the type of Keystore used for SSL communications (e.g. JKS, PKCS11, etc.).	String	JKS
keystore-uri	Location of the pre-created Keystore URI to be used for connecting to the GemFire cluster.	Resource	
ssl-keystore-password	Password for accessing the keys truststore.	String	
ssl-truststore-password	Password for accessing the trust store.	String	
truststore-type	Identifies the type of truststore used for SSL communications (e.g. JKS, PKCS11, etc.).	String	JKS
truststore-uri	Location of the pre-created truststore URI to be used for connecting to the GemFire cluster.	Resource	
user-home-directory	Local directory to cache the truststore and keystore files downloaded form the truststoreUri and keystoreUri locations.	String	user.home