Embedding Infinispan 11.0

Table of Contents

1. Configuring the Infinispan Maven Repository 1
    1.1. Configuring Your Infinispan POM 1
2. Installing Infinispan in Library Mode 2
3. Running Infinispan as an Embedded Library 3

Chapter 1. Configuring the Infinispan Maven Repository

Infinispan Java distributions are available from Maven.

Infinispan artifacts are available from Maven central. See the org.infinispan group for available Infinispan artifacts.

1.1. Configuring Your Infinispan POM

Maven uses configuration files called Project Object Model (POM) files to define projects and manage builds. POM files are in XML format and describe the module and component dependencies, build order, and targets for the resulting project packaging and output.

Procedure

1. Open your project `pom.xml` for editing.
2. Define the `version.infinispan` property with the correct Infinispan version.
3. Include the `infinispan-bom` in a `dependencyManagement` section.

   The Bill Of Materials (BOM) controls dependency versions, which avoids version conflicts and means you do not need to set the version for each Infinispan artifact you add as a dependency to your project.

4. Save and close `pom.xml`.

The following example shows the Infinispan version and BOM:

```
<properties>
  <version.infinispan>11.0.0.Final</version.infinispan>
</properties>

<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>org.infinispan</groupId>
      <artifactId>infinispan-bom</artifactId>
      <version>${version.infinispan}</version>
      <type>pom</type>
      <scope>import</scope>
    </dependency>
  </dependencies>
</dependencyManagement>
```

Next Steps

Add Infinispan artifacts as dependencies to your `pom.xml` as required.

Chapter 2. Installing Infinispan in Library Mode

Add Infinispan as an embedded library in your project.

Procedure

- Add the `infinispan-core` artifact as a dependency in your `pom.xml` as follows:

```
<dependencies>
  <dependency>
    <groupId>org.infinispan</groupId>
    <artifactId>infinispan-core</artifactId>
  </dependency>
</dependencies>
```

Chapter 3. Running Infinispan as an Embedded Library

Learn how to run Infinispan as an embedded data store in your project.

Procedure

- Initialize the default Cache Manager and add a cache definition as follows:

```
GlobalConfigurationBuilder global = GlobalConfigurationBuilder.
defaultClusteredBuilder();
DefaultCacheManager cacheManager = new DefaultCacheManager(global.build());
ConfigurationBuilder builder = new ConfigurationBuilder();
builder.clustering().cacheMode(CacheMode.DIST_SYNC);
cacheManager.administration().withFlags(CacheContainerAdmin.AdminFlag.VOLATILE).getOrC
reateCache("myCache", builder.build());
```

The preceding code initializes a default, clustered Cache Manager. Cache Managers contain your cache definitions and control cache lifecycles.

Infinispan does not provide default cache definitions so after initializing the default Cache Manager, you need to add at least one cache instance. This example uses the `ConfigurationBuilder` class to create a cache definition that uses the distributed, synchronous cache mode. You then call the `getOrCreateCache()` method that either creates a cache named "myCache" on all nodes in the cluster or returns it if it already exists.

Next steps

Now that you have a running Cache Manager with a cache created, you can add some more cache definitions, put some data into the cache, or configure Infinispan as needed.

Reference

- Configuring Infinispan Programmatically
- Setting Up Cluster Transport
- org.infinispan.Cache
- org.infinispan.commons.api.CacheContainerAdmin
- org.infinispan.configuration.cache.CacheMode
- org.infinispan.configuration.cache.Configuration
- org.infinispan.configuration.cache.ConfigurationBuilder
- org.infinispan.configuration.global.GlobalConfigurationBuilder
- org.infinispan.manager.DefaultCacheManager
- org.infinispan.manager.EmbeddedCacheManager