Hazelcast Idempotent Repository Tutorial

Hazelcast Idempotent Repository Tutorial

Overview

Apache Camel enables you to integrate the Idempotent Consumer Pattern in a really simple way. In this tutorial we will setup a cluster consisting of two OSGi containers (Apache Karaf). Inside this both nodes we will deploy two equal Camel bundles. This two bundles will receive a message from a REST interface, send them via a Hazelcast SEDA queue and finally store it into a distributed Hazelcast map. To avoid that a (equal) message will be consumed twice we're using the idempotent repository implementation based on Hazelcast. This implementation guarantees cluster wide that no message will be processed twice.

When to Use

If you use camel in a clustered environment (2 nodes or more) and you have to be sure that two equal messages never will be processed twice.

Why to use

Because you can create with very few effort a Idempotent Repository over n nodes inside your Camel application.

How to Use

We will create a camel route with a restlet consumer and a hazelcast producer. These routes will be deployed on the servers and form our cluster.

Preconsideration

The restlet consumer will get XML strings that represent an invoice:

```
<Invoice id="${number}">
        <Article name="foo"/>
</Invoice>
```

The Invoice id will be our Unique message ID.
We will then send multiple invoice messages to the restlet consumer. Hazelcast will spread these files throughout the cluster and write them to files. Because of the Idempotent Consumer Pattern no duplicate files should occur on our cluster.

Setup

You will need two servers with an OSGI runtime environment (we will use Karaf in this tutorial).

In the Karaf console add the camel features and install the needed packages:

```
features:addUrl mvn:org.apache.camel.karaf/apache-camel/2.8-SNAPSHOT/xml/features
features:install camel-core camel-spring camel-hazelcast
```

Our routes (similar on both nodes):

```
package org.apache.camel.tutorial.routes;

import org.apache.camel.builder.RouteBuilder;
import org.apache.camel.component.hazelcast.HazelcastConstants;
import org.apache.camel.processor.idempotent.hazelcast.*;

public class RoutesHazelcast extends RouteBuilder {

        public void configure() throws Exception {
                // create an instance with repo 'my-repo'
                HazelcastIdempotentRepository hazelcastIdempotentRepo = new HazelcastIdempotentRepository("my-
repo");

                // receiving a message from REST
                from("restlet:http://localhost:9080/init?restletMethod=post")
                .routeId("from_rest_to_seda")
                .toF("hazelcast:%sfoo", HazelcastConstants.SEDA_PREFIX);

                // receiving a message from Hazelcast SEDA
                fromF("hazelcast:%sfoo", HazelcastConstants.SEDA_PREFIX)
                .routeId("from_seda_to_map")
                // check for uniqueness
                .idempotentConsumer(xpath("/Invoice/@id"), hazelcastIdempotentRepo)
                // sending the message to a distributed map
                .setHeader(HazelcastConstants.OBJECT_ID, simple("${exchangeId}"))
                .setHeader(HazelcastConstants.OPERATION, constant(HazelcastConstants.PUT_OPERATION))
                .toF("hazelcast:%sbar", HazelcastConstants.MAP_PREFIX);
        }
}
```

Testing the routes 😉:

```
package org.apache.camel.tutorial.idempotent;

import org.apache.camel.test.junit4.CamelSpringTestSupport;
import org.junit.Test;
import org.springframework.context.support.AbstractApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.hazelcast.core.Hazelcast;
import com.hazelcast.core.IMap;

public class HazelcastIdempotenRepositoryTest extends CamelSpringTestSupport {

        private IMap<String, Object> bar = Hazelcast.getMap("bar");

        public void setUp()throws Exception {
                super.setUp();
                this.bar.clear();
        }

        public void tearDown() throws Exception {
                super.tearDown();
                this.bar.clear();
        }

        @Test
        public void testRoute() {

                assertEquals(0, this.bar.size());

                template.sendBody("restlet:http://localhost:9080/init?restletMethod=post", this.createXML(5));
                template.sendBody("restlet:http://localhost:9080/init?restletMethod=post", this.createXML(2));
                template.sendBody("restlet:http://localhost:9080/init?restletMethod=post", this.createXML(3));
                template.sendBody("restlet:http://localhost:9080/init?restletMethod=post", this.createXML(5));
                template.sendBody("restlet:http://localhost:9080/init?restletMethod=post", this.createXML(3));

                assertEquals(3, this.bar.size());
        }

        @Override
        protected AbstractApplicationContext createApplicationContext() {
                return new ClassPathXmlApplicationContext(
                                "/META-INF/spring/camel-context.xml");
        }

        private String createXML(int number) {
                return " <Invoice id=\"" + number + "\">\n"
                                + "  <Article name=\"foo\"/>\n" + "</Invoice>";
        }

}
```

Our camel-context.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="
       http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
       http://camel.apache.org/schema/spring http://camel.apache.org/schema/spring/camel-spring.xsd">

        <camelContext xmlns="http://camel.apache.org/schema/spring"
                trace="true">
                <package>org.apache.camel.tutorial.idempotent</package>
        </camelContext>
</beans>
```

The pom.xml:

```
<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">

        <modelVersion>4.0.0</modelVersion>

        <groupId>org.apache.camel.tutorial.routes</groupId>
        <artifactId>HazelcastIdempotenRepositoryDemo</artifactId>
        <packaging>bundle</packaging>
        <version>1.0.0</version>

        <name>Apache Hazelcast Idempotent Repository</name>
        <url>catify</url>

        <properties>
                <camel-version>2.8-SNAPSHOT</camel-version>
        </properties>

        <dependencies>
                <dependency>
                        <groupId>org.apache.camel</groupId>
                        <artifactId>camel-core</artifactId>
                        <version>${camel-version}</version>
                </dependency>
                <dependency>
                        <groupId>org.apache.camel</groupId>
                        <artifactId>camel-spring</artifactId>
                        <version>${camel-version}</version>
                </dependency>
                <dependency>
                        <groupId>org.apache.camel</groupId>
                        <artifactId>camel-hazelcast</artifactId>
                        <version>${camel-version}</version>
                </dependency>
                <dependency>
                        <groupId>org.apache.camel</groupId>
                        <artifactId>camel-restlet</artifactId>
                        <version>${camel-version}</version>
                </dependency>
                <dependency>
                        <groupId>org.apache.camel</groupId>
                        <artifactId>camel-test</artifactId>
                        <version>${camel-version}</version>
                        <scope>test</scope>
                </dependency>
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-context</artifactId>
                        <version>3.0.5.RELEASE</version>
                </dependency>
                <dependency>
                        <groupId>org.slf4j</groupId>
                        <artifactId>slf4j-log4j12</artifactId>
                        <version>1.6.1</version>
                </dependency>
                <dependency>
                        <groupId>log4j</groupId>
                        <artifactId>log4j</artifactId>
                        <version>1.2.16</version>
                </dependency>
        </dependencies>

        <build>
                <defaultGoal>install</defaultGoal>
                <plugins>
                        <plugin>
                                <groupId>org.apache.maven.plugins</groupId>
```

```
                                <artifactId>maven-compiler-plugin</artifactId>
                                <version>2.3.2</version>
                                <configuration>
                                        <source>1.6</source>
                                        <target>1.6</target>
                                </configuration>
                        </plugin>
                        <plugin>
                                <groupId>org.apache.felix</groupId>
                                <artifactId>maven-bundle-plugin</artifactId>
                                <version>2.1.0</version>
                                <extensions>true</extensions>
                                <configuration>
                                        <instructions>
                                                <Bundle-Name>${project.artifactId}</Bundle-Name>
                                                <Bundle-SymbolicName>HazelcastIdempotenRepositoryDemo-osgi<
/Bundle-SymbolicName>
                                                <Export-Package>
                                                        org.apache.camel.tutorial.routes
                                        </Export-Package>
                                                <Import-Package>*</Import-Package>
                                                <Include-Resource>src/main/resources</Include-Resource>
                                                <Spring-Context>*;publish-context:=false;create-asynchronously:
=true</Spring-Context>
                                                <DynamicImport-Package>*</DynamicImport-Package>
                                                <Implementation-Title>HazelcastIdempotenRepositoryDemo<
/Implementation-Title>
                                                <Implementation-Version>${project.version}</Implementation-
Version>
                                        </instructions>
                                </configuration>
                        </plugin>
                </plugins>
        </build>

</project>
```

Switch to your project directory and build the jar (with maven):

```
mvn clean install
```

Copy the jar to your servers and install them in Karaf:

```
osgi:install file:/home/username/HazelcastIdempotenRepository-1.0.0.jar
```

Validate

In the java project create a class with the following code:

```
package org.apache.camel.tutorial;

import java.util.Random;

import org.apache.camel.Exchange;
import org.apache.camel.Main;
import org.apache.camel.Processor;
import org.apache.camel.ProducerTemplate;
import org.apache.camel.builder.RouteBuilder;
import org.apache.camel.component.hazelcast.HazelcastConstants;
import org.apache.camel.processor.idempotent.hazelcast.HazelcastIdempotentRepository;

import com.hazelcast.core.Hazelcast;
import com.hazelcast.core.IMap;

public class HazelcastIdempotenRepositoryRunner {

        private Main main;
```

```
	public static void main(String[] args) throws Exception {
		HazelcastIdempotenRepositoryRunner example = new HazelcastIdempotenRepositoryRunner();
		example.boot();
	}

	public void boot() throws Exception {

		// clear repo-Map for seeing effect on re-running this script
		IMap<String, Object> repo = Hazelcast.getMap("my-repo");

		if (!repo.isEmpty()) {
			repo.clear();
		}

		// create a Main instance
		main = new Main();
		// enable hangup support so you can press ctrl + c to terminate the JVM
		main.enableHangupSupport();

		main.addRouteBuilder(this.createRouteBuilder());

		// run until you terminate the JVM
		System.out.println("Starting Camel. Use ctrl + c to terminate the JVM.\n");
		main.run();
	}

	protected RouteBuilder createRouteBuilder() {

		return new RouteBuilder() {

			@Override
			public void configure() throws Exception {

				from("timer://foo?fixedRate=true&period=500")
				.process(new Processor() {

					private Random random = new Random();

					@Override
					public void process(Exchange ex) throws Exception {

						// create random invoice number between 1 and 20
						int n1 = random.nextInt(20) + 1;
						int n2 = random.nextInt(2) + 1;

						String body = " <Invoice id=\"" + n1  + "\">"
						+ "<Article name=\"foo\"/>" + "</Invoice>";

						// put xml into body
						ex.getOut().setBody(body);

						// set header for 'node' decision
						ex.getOut().setHeader("server", n2);
					}
				})
				.choice()
					.when(header("server").isEqualTo(1))
						.log("sending message to 'host1'  --> ${body}")
						.toF("restlet:http://host1:9080/init?restletMethod=post")
					.otherwise()
						.log("sending message to 'host2'  --> ${body}")
						.toF("restlet:http://host2:9080/init?restletMethod=post");

				fromF("hazelcast:%sbar", HazelcastConstants.MAP_PREFIX)
				.log("--- new message received ---");
			}
		};
	}
}
```

While executing the project the console will output the send and received messages:

```
[        foo] route1 INFO  sending message to 'host1'  -->  <Invoice id="20"><Article name="foo"/></Invoice>
[...thread-3] route2 INFO  --- new message received ---
[        foo] route1 INFO  sending message to 'host1'  -->  <Invoice id="1"><Article name="foo"/></Invoice>
[        foo] route1 INFO  sending message to 'host2'  -->  <Invoice id="16"><Article name="foo"/></Invoice>
```

If the message hasn't been sent before, you will get a `"new message received"`, if the message is already in the idempotent repository it won't be processed. If you let run the script for a minute there should be almost no `"new message received"` log out printed.