Camel 1.5.0 Release

Camel 1.5.0 release

‹ed URL

New and Noteworthy

Welcome to the 1.5.0 release which approx 266 issues resolved (new features, improvements and bug fixes such as...)

- support for Guice for dependency injection along with a new Guice JMS Example
- major improvements in FTP and File components
- File consumers now default uses exclusive read locking when it consume files, meaning that they wont consume files that are in the progress of being written by third part. The FTP consumer has this support as well, however it is disabled by default as it requires write privileges.
- File and FTP component supports expression to set dynamic filename patterns instead of using the `FileComponent.HEADER_FILE_NAME` header. See File Language for samples and use cases.
- important changes in ProducerTemplate (see below)
- Exception Clause now supports marking exceptions as being **handled** so callers doesn't receive the caused exception, but you can set the response to return instead. This is a very important feature.
- various improvements in Flatpack and XMPP components
- minor improvements in CXF, SpringIntegration, HTTP, Mail and MINA components
- minor improvements in Aggregator supporting a better fluent builder, etc.
- Splitter now handles streaming avoiding reading entire content into memory (consuming very big 1gb files is now possible)
- introduced new JT400, HL7, LDAP and Smooks components
- introduced Delay Interceptor to slow down processing to show how things is happening nice and slow, so you are not bombarded with zillions of logging output.
- Mail can now send html mails with the new `contentType` option
- Camel Maven Archetypes now available in m2eclipse Maven project creation
- Java WebStart support
- refinements in loading resources using OSGi bundles, should now handle different OSGi platforms much better
- Type Converter now supports Exchange as 2nd parameter to allow converters access to the Exchange and thus the CamelContext as well. To be used for setting encoding in the future.
- New Loop api in DSL allows processing of a message a number of times, possibly in different ways.
- Spring DSL improve
- improved Tracer formatting and more options for configuration
- improved support for all Endpoints, not just singletons in CamelContext.
- minor tweaks to the Visualisation generator
- IBM JDK support (notable exceptions)

New Enterprise Integration Patterns

- Dynamic Router

New Components

- JT400 for integration with AS/400 dataqueues
- HL7 for working with the HL7 MLLP protocol and the HL7 model using the HAPI library
- LDAP to perform searches in LDAP servers
- Smooks for working with EDI parsing using the Smooks library

New DSL

- Loop
- Scala (work in progress, not fully feature complete)

New Annotations

- @Consume for POJO Consuming
- @Produce for POJO Producing
- @RecipientList for creating an annotation based Recipient List
- Exchange Pattern Annotations for specifing the message exchange pattern

New Data Formats

- Flatpack
- HL7
- EDI

New Languages

- Constant
- Header
- File Language really useful for the File and FTP component for setting dynamic names using patterns

New Examples

- camel-example-axis
- Guice JMS Example

API breakings

- `ProducerTemplate` now throws `RuntimeCamelException` for sendBody/requestBody methods.
- `DefaultTypeConverter.convertTo()` now throws a `NoTypeConversionAvailableException` (`RuntimeException`) when it cannot find a suitable conversion (see Type Converter).
- The `addSingletonEndpoint()` and `removeSingletonEndpoint` methods in CamelContext are now deprecated in favor of `addEndpoint` and `removeEndpoint`.

Known Issues

- Aggregator always consume directly from the endpoint, see CAMEL-393

Important changes to consider when upgrading

ProducerTemplate

The ProducerTemplate has refined its sendBody and requestBody methods to throw `RuntimeCamelException`, with the caused exception wrapper, in case the Exchange failed with an exception. Also if the exchange has set an `FAULT` message then the `FAULT` message is returned.

The old behavior in Camel 1.4 or older was just plain wrong!

JMX

Option usePlatformMBeanServer has changed its default value from **false** to **true**.
The naming convention for the JMX `ObjectName}}s has changed. Camel now uses simpler, shorter {{ObjectName`(s).

TypeConverter

An implementation of TypeConverter should now throw a NoTypeConversionAvailableException if conversion is not possible. The semantical ambiguity of null (both valid result and indication of no conversion) is now resolved, but this may impact existing code in that it should now catch the exception instead of checking for null.

setHeader element in Spring DSL changed

You can no longer use a 'value' attribute like this to set a header to a constant:

```
<route>
  <from uri="seda:a"/>
  <setHeader headerName="theHeader" value="the value">
    <expression/>
  </setHeader>
  <to uri="mock:b"/>
</route>
```

Now, you can use a Constant expression to do the same thing:

```
<route>
  <from uri="seda:a"/>
  <setHeader headerName="theHeader">
    <constant>the value</constant>
  </setHeader>
  <to uri="mock:b"/>
</route>
```

Notice that this constant expression is also possible in the Java DSL:

```
.setHeader("theHeader", constant("the value"))
```

redeliveryPolicy specification in Spring DSL changed

Instead of

```
<redeliveryPolicy>
  <maximumRedeliveries>1</maximumRedeliveries>
  ...
</redeliveryPolicy>
```

You now set redelivery policy settings with attributes

```
<redeliveryPolicy maximumRedeliveries="1" .../>
```

Data format specification in Spring DSL changed

The way you specify data formats in the Spring DSL has changed. You could do this in Camel 1.4:

```
<camelContext id="camel" xmlns="http://activemq.apache.org/camel/schema/spring">
  <jaxb id="myJaxb" prettyPrint="true" contextPath="org.apache.camel.example"/>
  <xstream id="xs" prettyPrint="true"/>

  <route>
    <from uri="direct:start"/>
    <marshal ref="myJaxb"/>
    <to uri="direct:marshalled"/>
  </route>
  ...
```

Now, you specify the same as:

```
<camelContext id="camel" xmlns="http://activemq.apache.org/camel/schema/spring">
  <dataFormats>
    <jaxb id="myJaxb" prettyPrint="true" contextPath="org.apache.camel.example"/>
    <xstream id="xs" prettyPrint="true"/>
  </dataFormats>

  <route>
    <from uri="direct:start"/>
    <marshal ref="myJaxb"/>
    <to uri="direct:marshalled"/>
  </route>
  ...
```

CXF Producer

The result of CXF producer has changed to use MessageContentsList instead of Object array to hold the response.
If you use the exchange.getOut.getBody(YourType.class), CXFMessage will check the list members and chose the most right answer for you.

CXF Producer before Camel 1.5

```
     Object[] oldResult = (Object[])oldExchange.getOut().getBody();
     BankQuote oldQuote = (BankQuote) oldResult[0];
```

CXF Produer in Camel 1.5

```
     BankQuote oldQuote = oldExchange.getOut().getBody(BankQuote.class);
```

Now the CXF Producer(in Camel 1.5) will throw the exception after the CXF client gets the exception, in this way you can leverage the Camel errorHandler for handling the exception.
If you don't want camel redeliver the message when your producer got the common SOAP fault, specially you are using CXF producer and CXF consumer as a proxy, you'd better override the Camel default errorHandler.

You can do it with DSL

```
  protected RouteBuilder createRouteBuilder() {
     return new RouteBuilder() {
         public void configure() {
             errorHandler(noErrorHandler());
             from(routerEndpointURI).to("log:org.apache.camel?level=DEBUG").to(serviceEndpointURI);
         }
     };
  }
```

Or Spring

```
<cxf:cxfEndpoint id="routerEndpoint" address="http://localhost:9003/CamelContext/RouterPort"
                 serviceClass="org.apache.hello_world_soap_http.GreeterImpl"/>

<cxf:cxfEndpoint id="serviceEndpoint" address="http://localhost:9000/SoapContext/SoapPort"
                 wsdlURL="testutils/hello_world.wsdl"
                 serviceClass="org.apache.hello_world_soap_http.Greeter"
                 endpointName="s:SoapPort"
                 serviceName="s:SOAPService"
         xmlns:s="http://apache.org/hello_world_soap_http" />

<camelContext id="camel" xmlns="http://activemq.apache.org/camel/schema/spring">
 <route errorHandlerRef="noErrorHandler">
   <from uri="cxf:bean:routerEndpoint" />
   <to uri="cxf:bean:serviceEndpoint" />
 </route>
</camelContext>

<bean id="noErrorHandler" class="org.apache.camel.builder.NoErrorHandlerBuilder"/>
```

FTP component

The option `consumer.recursive` has changed the default value from **true** to **false**.

Now throws `FtpOperationFailedException` with the FTP error code and reason. For instance error code 530 is not authorized. This applies for both consumer and producer. The consumer will now also regard failing move or rename operations as a severe error throw a `FtpOperationFailedExcepti` on instead of `WARN` logging. As a consumer you will **not** receive an Exchange, as opposed to the situation in Camel 1.4.0. Bottom line: all the FTP operations should succeed before the consumer will process the Exchange.

File Consumer

The option `consumer.recursive` has changed the default value from **true** to **false**.

Http Producer

Now the http producer will throw a `HttpOperationFailedException` if the response code is not 1xx or 2xx. You can get the status code, status line and location for the exception.
In the Camel < 1.5.0 , http producer does not check the response code and puts the response message into the out message.

Http producer has better algorithm to compute if either GET or POST should be used.

camel-mina

In Camel 1.5 the sync option has changed its default value from **false** to **true**, as we felt it was confusing for end-users when they used Mina to call remote servers and Camel wouldn't wait for the response. To remedy this you had to add the sync=true option before it worked, we want this to work out-of-the-box without having to specify this option.

camel-mail

In Camel 1.5 the following default options has changed in camel-mail:

- `deleteProcessedMessages` is now **false** as we felt Camel should not delete mails on the mail server by default.
- `processOnlyUnseenMessages` is now **true** as we felt Camel should only poll new mails by default.

removeOutHeader DSL method removed

Since any exchange coming into the removeOutHeader processor would have an IN message only, this method was useless and so was removed.

Aggregator and `AggregationCollection`

The AggregationCollection used by Aggregator is changed to an interface instead of a class. This allows end-users to provide their own collections that isn't a subclass of the default. The old AggregationCollection class is renamed to DefaultAggregationCollection.

Aggregator must be configured directly on the consumer:

- This is valid: `from("foo").aggregate("bar")`
- This is **not** valid: `from("foo").setHeader("id").aggreagate("bar")`
  See known issues.

Spring Event component

The `event` name (scheme) has been renamed to `spring-event`

Getting the Distributions

Binary Distributions

Description	Download Link	PGP Signature file of download
Windows Distribution	apache-camel-1.5.0.zip	apache-camel-1.5.0.zip.asc
Unix/Linux/Cygwin Distribution	apache-camel-1.5.0.tar.gz	apache-camel-1.5.0.tar.gz.asc

The above URLs use redirection

The above URLs use the Apache Mirror system to redirect you to a suitable mirror for your download. Some users have experienced issues with some versions of browsers (e.g. some Safari browsers). If the download doesn't seem to work for you from the above URL then try using FireFox

Source Distributions

Description	Download Link	PGP Signature file of download
Source for Windows	apache-camel-1.5.0-src.zip	apache-camel-1.5.0-src.zip.asc
Source for Unix/Linux/Cygwin	apache-camel-1.5.0-src.tar.gz	apache-camel-1.5.0-src.tar.gz.asc

Getting the Binaries using Maven 2

To use this release in your maven project, the proper dependency configuration that you should use in your Maven POM is:

```
<dependency>
  <groupId>org.apache.camel</groupId>
  <artifactId>camel-core</artifactId>
  <version>1.5.0</version>
</dependency>
```

SVN Tag Checkout

```
svn co http://svn.apache.org/repos/asf/activemq/camel/tags/camel-1.5.0
```

Changelog

For a more detailed view of new features and bug fixes, see:

- JIRA Release notes for 1.5.0