

Camel 2.0-M2 Release

Camel 2.0-M2 release

New and Noteworthy

Welcome to the 2.0-M2 release which approx 222 issues resolved (new features, improvements and bug fixes such as...)

- Introduced [type converter registry](#) to allow end users to programmatic get access to this registry
- major improvements to [Jetty](#) for better handling of exception/faults and easier end-user customization how response should be written
- major improvements to [File](#) for consuming files, now supporting pluggable sorting, filtering and idempotent repositories.
- major refactor and improvements to [FTP](#), aligning it with the [File](#) component.
- minor improvements to [JDBC](#), [SQL](#), [Http](#) and [Mina](#)
- major refactor and improvements to [CXF](#) component. [CxfBinding](#), [HeaderFilterStrategy](#), [Bus](#) can be looked up from registry and configured at the endpoint level.
- [File](#) and [FTP](#) producers added option to write files using a temporary name and then rename it to the real name after the write completes
- [SFTP](#) added support for knownhosts and privatekey files
- [Mail](#) now supports setting additional SUN java mail properties
- [Mina](#) now supports setting text line delimiters for textline codec.
- [Mina](#) producer now throws [CamelExchangeException](#) in case no response received from remote server when in sync mode (`sync=true`)
- [RedeliveryPolicy](#) added support for `ref` attribute to reference a existing policy in the [Registry](#)
- [RedeliveryPolicy](#) added `delayPattern` option for using a special pattern for setting different delay based on intervals
- Added `onWhen` and `retryUntil` predicates to [Exception Clause](#)
- [List](#) component renamed to [Browse](#)
- [Exception Clause](#) is much smarter as it will use caused by exception hierarchy for matching as well (will use bottom ups)
- [Dead Letter Channel](#) and [Exception Clause](#) now have `onRedeliver` to allow custom processing an [Exchange](#) **before** its being redelivered. Allowing you to work on the message being sent.
- Added support for [# syntax in Endpoint URI's](#) to allow endpoint URI's to refer to bean in the [Registry](#).
- `consumer.` prefix can be omitted for scheduled polling consumers such as [File](#), [FTP](#) consumers. Thus the URI options is more simpler and you don't have to remember which requires `consumer.` prefix and which doesn't.
- [Tracer](#) improved to allow custom routing and processing of [TraceEventMessage](#) so you can store trace logs as you like, for instance in a database.
- [Tracer](#) now also displays the previous node so you can see where the Exchange is coming from.
- [Quartz](#) has added support for stateful jobs.
- Introduced [@FallbackConverter](#) for using annotation based fallback type converters
- Extended [content enrichment](#) support via the [enrich](#) DSL element.
- [Simple](#) language now supports a basic set of operators.
- [Aggregator](#) now supports grouped exchanges out of the box, so you can combine all aggregated exchanges into a single grouped exchange going out.
- Multiple inputs can be define as input to routes: `from("activemq:queue:order", "file://order").to("bean:handleOrder");`
- [CXF](#) component supports a new "cxfbean" endpoint that allows RESTful requests to be routed to JAXRS annotated service beans.
- Setting cron expression in camel-quartz improved.
- Better handling of interrupts while shutting down.
- The [JMSReplyTo](#) destination is available as [Exchange](#) property when consuming JMS messages.
- Better support for [InOptionalOut](#) Message Exchange Pattern.
- [Pluggable Class Resolvers](#) SPI for class resolvers allowing third party platforms such as JBoss to provide integration with Camel.
- Refined API to reduce package tangling.
- Introduced [wireTap](#) node for the [Wire Tap](#) EIP pattern, supporting the traditional tapping and sending a new message.
- Added `fromF` and `toF` in the Java DSL to build uri strings using `String.format` with arguments, e.g.: `fromF("ftp://%s@myserver?password=%s", user, password)..to`
- Improved tooling being able to retrieve more runtime information from the [CamelContext](#)
- [Scala](#) DSL is improved
- Added `JmsMessageType` option to [JMS](#) to allow you to set which `javax.jms.Message` implementation to use for sending a JMS message.
- Fixed a rare bug when JMX is disabled, a [Dead Letter Channel](#) could mistakenly be added to some routes, when a global `noErrorHandler` was configured.
- [JMS](#) is now able to preserve hyphen in JMS keys (eg `Content-Type` can be sent as a Message header). Introduced pluggable strategy to allow end users to use their custom key formatter for encode/decode.

- Added option `transferExchange` to **JMS** so you can transfer the **Exchange** over the wire. Can be used to use JMS queues for **Dead Letter Channel** to preserve all information from the failed **Exchange** including the original exception with stack trace.
- Added option `transferException` to **JMS** so when using **Request Reply** messaging with JMS, any caused exception on the server side will be returned as response to the client.
- Added `rollback` as DSL keyword to force a rollback of the given **Exchange**. Does this by throwing an `org.apache.camel.RollbackExchangeException`.
- Added SOAP Message Header filtering capability in camel-cxf component.
- ProducerTemplate API reworked for `sendBody` operations to return void, as they are InOnly. Use `requestBody` if you need InOut.
- Introduced `ConsumerTemplate` supporting the **Polling Consumer** EIP
- Custom endpoints with Spring `@ManagedResource` is now also registered in the mbean server.
- The default error handler is changed from **Dead Letter Channel** to **DefaultErrorHandler**.
- **HTTP** component added option `throwException` that can be disabled to allow failed response codes to be returned without throwing a `HttpOperationsFailedException`.
- Overhaul and improvements to the **Try Catch Finally** DSLs, so they are on pair with the **Exception Clause**.
- Overhaul and improvements to the **Intercept** DSLs. Added new `interceptSendToEndpoint` to intercept sending to a given **Endpoint**.
- Added the notion of a Channel in the routes.
- Added support to navigate the routes at runtime, so you can dynamic change or affect the processors in the routes.
- Introduced a new **Async** API for asynchronous messaging.
- **Jetty** now supports configuring Handler for e.g. security.
- **iBatis** added option to set `StatementType` for fine grained control of which `SqlMapClient` operation to invoke. Allowing **iBatis** component to be used like **Sql** or the **Jdbc** component.
- Added **OnCompletion** callback to **Exchange** so you can do custom routing when an **Exchange** is completed. You can for instance use it to send an email if an **Exchange** failed.
- Added **Failover** as load balancer.
- Added `throwException` to the DSL.
- Added `@Handler` annotation to mark a method to be invoked when using **POJO** in routes.
- **Predicate** uses type coercion for improving matching, thus allowing you for instance to compare a String with an Integer, "true" with a boolean, or enum type as string etc.
- Started to standardize consumers that supported batching as a **Batch Consumer**.
- **Connection pooling** for **FTP** and **Mina** producers. This allows thread safe concurrency usage of these components out of the box. The connection pooling is pluggable so you can use a 3rd party pool framework.
- Added option `existFile` to **File** and **FTP** component. This option allows you to configure what should happen when you write a file and an existing file with that name already exists. The **File** producer will now by default **Override** existing files. In Camel 1.x it would default append.
- **Seda** and **Vm** now supports **Request Reply** and waiting for the reply if one expected.
- Fixed issue with using 3rd party annotations and Camel annotations in same POJO class. Now all annotations is processed by Spring.
- Added **Jackson** as supported library for the **JSON** data format.
- Simplified using **Delayer** especially in Spring DSL.
- Improved **Idempotent Consumer** to eagerly detect duplicated messages for in progress exchanges.
- Added **Camel property to set a max chars limit** for `DEBUG` logs Message bodies. To avoid logging very big payloads. The default limit is 1000 chars.
- Configuration of http proxy is now possible with the **HTTP** component
- **Message Filter** EIP marks Exchanges as filtered and is now skipped for aggregation in `AggregationStrategy` for example used when doing **Spitter** or **Aggregator**
- **DefaultErrorHandler** is now just as powerful as **Dead Letter Channel** support redelivery et. all.
- **Dead Letter Channel** will by default **handle** exceptions.
- **TransactionErrorHandler** is now just as powerful as **Dead Letter Channel** support redelivery et. all.

New Enterprise Integration Patterns

- [Sort](#)

New Components

- [Cometd](#)
- [FreeMarker](#)
- [Restlet](#)
- [RSS](#)
- [Quickfix](#)

New DSL

- [Scala](#)

New Annotations

- [@FallbackConverter](#)

New Data Formats

- [Bindy](#)
- [JSON](#)
- [TidyMarkup](#)
- [GZip](#)
- [Zip](#)
- [XMLSecurity](#)

New Languages

- [Mvel](#)
- [Property](#)

New Examples

- [camel-example-pojo-messaging](#) shows how to use annotations to produce, consume or route messages to Camel endpoints without using any DSL.
- [camel-example-reportincident](#) is based on a real life use case.
- [camel-example-tracer](#) is a new example showing [Tracer](#) persisting trace events into a database using [JPA](#)

API breaking

Client API

ProducerTemplate

The `sendBody` methods now return void for InOnly messaging. Use `requestBody` if you want InOut messaging.

AggregationStrategy

The `aggregate` method is now also invoked on the very first exchange. Allowing the end users to be in full power. At this first invocation the `oldExchange` parameter is `null`.

Notable changes to DSL

- Renamed DSL operations
 - `splitter` -> `split`
 - `resequencer` -> `resequence`
 - `aggregator` -> `aggregate`
 - `delayer` -> `delay`
 - `throttler` -> `throttle`
 - `expression` -> `language`
 - `try` (Spring DSL) -> `doTry`
 - `catch` (Spring DSL) -> `doCatch`
 - `finally` (Spring DSL) -> `doFinally`
 - `tryBlock` (Java DSL) -> `doTry`
 - `handle` (Java DSL) -> `doCatch`
 - `finallyBlock` (Java DSL) -> `doFinally`
 - `intercept` -> `interceptFrom`
 - `thread` -> `threads`
 - `throwFault` has been removed, you can set a the fault using the `exchange.setFault` method
- Renamed DSL in SpringBuilder
 - `bean` -> `lookup`

Components

- The [List](#) component is renamed to [Browse](#) component
- The [Queue](#) component has been removed
- The [FTP](#) component have renamed some of its URI options
- The [File](#) component have renamed some of its URI options
- The [CXF](#) component have been refactored and APIs have been simplified

Known Issues

See known issues from previous releases.

Important changes to consider when upgrading

The default error handler is no longer [Dead Letter Channel](#) but [DefaultErrorHandler](#) without redelivery support. There are no single solution that fits all solutions, so you should tailor the error handling according to your needs.

Getting the Distributions

Binary Distributions

--	--	--

Description	Download Link	PGP Signature file of download
Windows Distribution	apache-camel-2.0-M2.zip	apache-camel-2.0-M2.zip.asc
Unix/Linux/Cygwin Distribution	apache-camel-2.0-M2.tar.gz	apache-camel-2.0-M2.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-2.0-M2-src.zip	apache-camel-2.0-M2-src.zip.asc
Source for Unix/Linux /Cygwin	apache-camel-2.0-M2-src.tar.gz	apache-camel-2.0-M2-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>2.0-M2</version>
</dependency>
```

SVN Tag Checkout

```
svn co http://svn.apache.org/repos/asf/camel/tags/camel-2.0-M2
```

Changelog

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

- [JIRA Release notes for 2.0-M2](#)