URIs

URIs

Camel makes extensive use of URIs to allow you to refer to endpoints which are lazily created by a Component if you refer to them within Routes.

important

Make sure to read How do I configure endpoints to learn more about configuring endpoints. For example how to refer to beans in the Registry or how to use raw values for password options, and using property placeholders etc.

Current Supported URIs

Component / ArtifactId / URI	Description
AHC / `camel-ahc` `ahc:http[s]://hostName[:port][/resourceUri][?options]`	To call external HTTP services using Async Http Client
AHC-WS / `camel-ahc-ws` `ahc-ws[s]://hostName[:port][/resourceUri][?options]`	To exchange data with external Websocket servers using Async Http Client
AMQP / `camel-amqp` `amqp:[queue:|topic:]destinationName[?options]`	For Messaging with AMQP protocol
APNS / `camel-apns` `apns:<notify|consumer>[?options]`	For sending notifications to Apple iOS devices

Atmosphere-Websocket / `camel-atmosphere-websocket` `atmosphere-websocket:///relative path[?options]`	To exchange data with external Websocket clients using Atmosphere
Atom / `camel-atom` `atom:atomUri[?options]`	Working with Apache Abdera for atom integration, such as consuming an atom feed.
Avro / `camel-avro` `avro:[transport]:[host]:[port][/messageName][?options]`	Working with Apache Avro for data serialization.
AWS-CW / camel-aws `aws-cw://namespace[?options]`	For working with Amazon's CloudWatch (CW).
AWS-DDB / camel-aws `aws-ddb://tableName[?options]`	For working with Amazon's DynamoDB (DDB).
AWS-DDBSTREAM / camel-aws `aws-ddbstream://tableName[?options]`	For working with Amazon's DynamoDB Streams (DDB Streams).
AWS-EC2 / camel-aws `aws-ec2://label[?options]`	For working with Amazon's Elastic Compute Cloud (EC2).

AWS-SDB / camel-aws `aws-sdb://domainName[?options]`	For working with Amazon's SimpleDB (SDB).
AWS-SES / camel-aws `aws-ses://from[?options]`	For working with Amazon's Simple Email Service (SES).
AWS-SNS / camel-aws `aws-sns://topicName[?options]`	For Messaging with Amazon's Simple Notification Service (SNS).
AWS-SQS / camel-aws `aws-sqs://queueName[?options]`	For Messaging with Amazon's Simple Queue Service (SQS).
AWS-SWF / camel-aws `aws-swf://<worfklow|activity>[?options]`	For Messaging with Amazon's Simple Workflow Service (SWF).
AWS-S3 / camel-aws `aws-s3://bucketName[?options]`	For working with Amazon's Simple Storage Service (S3).
Bean / `camel-core` `bean:beanName[?options]`	Uses the Bean Binding to bind message exchanges to beans in the Registry. Is also used for exposing and invoking POJO (Plain Old Java Objects).

Beanstalk / `camel-beanstalk` `beanstalk:hostname:port/tube[?options]`	For working with Amazon's Beanstalk.
Bean Validator / `camel-bean-validator` `bean-validator:label[?options]`	Validates the payload of a message using the Java Validation API (JSR 303 and JAXP Validation) and its reference implementation Hibernate Validator
Box / `camel-box` `box://endpoint-prefix/endpoint?[options]`	For uploading, downloading and managing files, managing files, folders, groups, collaborations, etc. on Box.com.
Braintree / `camel-braintree` `braintree://endpoint-prefix/endpoint?[options]`	Component for interacting with Braintree Payments via Braintree Java SDK
Browse / `camel-core` `browse:someName`	Provides a simple BrowsableEndpoint which can be useful for testing, visualisation tools or debugging. The exchanges sent to the endpoint are all available to be browsed.
Cache / `camel-cache` `cache://cacheName[?options]`	The cache component facilitates creation of caching endpoints and processors using EHCache as the cache implementation.
Cassandra / `camel-cassandraql` `cql:localhost/keyspace`	For integrating with Apache Cassandra.

Class / `camel-core` `class:classNameName[?options]`	Uses the Bean Binding to bind message exchanges to beans in the Registry. Is also used for exposing and invoking POJO (Plain Old Java Objects).
Chronicle Engine / `camel-chronicle` `chronicle-engine:addresses/path[?options]`	Chronicle Engine is a high performance, low latency, reactive processing framework.
Chunk / `camel-chunk` `chunk:templateName[?options]`	Generates a response using a Chunk template
CMIS / `camel-cmis` `cmis://cmisServerUrl[?options]`	Uses the Apache Chemistry client API to interface with CMIS supporting CMS
Cometd / `camel-cometd` `cometd://hostName:port/channelName[?options]`	Used to deliver messages using the jetty cometd implementation of the bayeux protocol
Consul / `camel-consul` `consul:apiEndpoint[?options]`	For interfacing with an Consul.
Context / `camel-context` `context:camelContextId:localEndpointName[?options]`	Used to refer to endpoints within a separate CamelContext to provide a simple black box composition approach so that routes can be combined into a CamelContext and then used as a black box component inside other routes in other CamelContexts

ControlBus / `camel-core` `controlbus:command[?options]`	ControlBus EIP that allows to send messages to Endpoints for managing and monitoring your Camel applications.
CouchDB / `camel-couchdb` `couchdb:hostName[:port]/database[?options]`	To integrate with Apache CouchDB.
Crypto (Digital Signatures) / `camel-crypto` `crypto:<sign|verify>:name[?options]`	Used to sign and verify exchanges using the Signature Service of the Java Cryptographic Extension.
CXF / `camel-cxf` `cxf:<bean:cxfEndpoint|//someAddress>[?options]`	Working with Apache CXF for web services integration
CXF Bean / `camel-cxf` `cxfbean:serviceBeanRef[?options]`	Proceess the exchange using a JAX WS or JAX RS annotated bean from the registry. Requires less configuration than the above CXF Component
CXFRS / `camel-cxf` `cxfrs:<bean:rsEndpoint|//address>[?options]`	Working with Apache CXF for REST services integration
DataFormat / `camel-core` `dataformat:name:<marshal|unmarshal>[?options]`	for working with Data Formats as if it was a regular Component supporting Endpoints and URIs.

DataSet / `camel-core` `dataset:name [?options]`	For load & soak testing the DataSet provides a way to create huge numbers of messages for sending to Components or asserting that they are consumed correctly
Direct / `camel-core` `direct: someName[?options]`	Synchronous call to another endpoint from **same** CamelContext.
Direct-VM / `camel-core` `direct-vm: someName[?options]`	Synchronous call to another endpoint in another CamelContext running in the same JVM.
DNS / `camel-dns` `dns:operation [?options]`	To lookup domain information and run DNS queries using DNSJava

Disruptor / `camel-disruptor`
`disruptor: someName[?<option>]`
`disruptor-vm: someName[?<option>]`
	To provide the implementation of SEDA which is based on disruptor

Docker / `camel-docker` `docker:// [operation]? [options]`	To communicate with Docker
Dozer / `camel-dozer` `dozer://name? [options]`	To convert message body using the Dozer type converter library.

Dropbox / `camel-dropbox` `dropbox://[operation]?[options]`	The **dropbox:** component allows you to treat Dropbox remote folders as a producer or consumer of messages.
EJB / `camel-ejb` `ejb:ejbName[?options]`	Uses the Bean Binding to bind message exchanges to EJBs. It works like the Bean component but just for accessing EJBs. Supports EJB 3.0 onwards.
Ehcache / `camel-ehcache` `ehcache://cacheName[?options]`	The cache component facilitates creation of caching endpoints and processors using Ehcache 3 as the cache implementation.
ElasticSearch / `camel-elasticsearch` `elasticsearch://clusterName[?options]`	For interfacing with an ElasticSearch server.
Etcd / `camel-etcd` `etcd:namespace[/path][?options]`	For interfacing with an Etcd key value store.
Spring Event / `camel-spring` `spring-event://default`	Working with Spring ApplicationEvents
EventAdmin / `camel-eventadmin` `eventadmin:topic[?options]`	Receiving OSGi EventAdmin events
Exec / `camel-exec` `exec://executable[?options]`	For executing system commands

Facebook / `camel-facebook` `facebook://endpoint[?options]`	Providing access to all of the Facebook APIs accessible using Facebook4J
File / `camel-core` `file://nameOfFileOrDirectory[?options]`	Sending messages to a file or polling a file or directory.
Flatpack / `camel-flatpack` `flatpack:[fixed|delim]:configFile[?options]`	Processing fixed width or delimited files or messages using the FlatPack library

Flink / `camel-flink`
`flink:dataset[?options]`
`flink:datastream[?options]`
	Bridges Camel connectors with Apache Flink tasks.

FOP / `camel-fop` `fop:outputFormat[?options]`	Renders the message into different output formats using Apache FOP
FreeMarker / `camel-freemarker` `freemarker:templateName[?options]`	Generates a response using a FreeMarker template
FTP / `camel-ftp` `ftp:contextPath[?options]`	Sending and receiving files over FTP.

FTPS / `camel-ftp` `ftps://[username@]hostName[:port]/directoryName[?options]`	Sending and receiving files over FTP Secure (TLS and SSL).
Ganglia / `camel-ganglia` `ganglia:destination:port[?options]`	Sends values as metrics to the Ganglia performance monitoring system using gmetric4j. Can be used along with JMXetric.
GAuth / camel-gae `gauth://name[?options]`	Used by web applications to implement an OAuth consumer. See also Camel Components for Google App Engine.
GHttp / camel-gae `ghttp:contextPath[?options]`	Provides connectivity to the URL fetch service of Google App Engine but can also be used to receive messages from servlets. See also Camel Components for Google App Engine.
Git / camel-git `git:localRepositoryPath[?options]`	Supports interaction with Git repositories
Github / camel-github `github:endpoint[?options]`	Supports interaction with Github
GLogin / camel-gae `glogin://hostname[:port][?options]`	Used by Camel applications outside Google App Engine (GAE) for programmatic login to GAE applications. See also Camel Components for Google App Engine.
GTask / camel-gae `gtask://queue-name[?options]`	Supports asynchronous message processing on Google App Engine by using the task queueing service as message queue. See also Camel Components for Google App Engine.

Google Calendar / camel-google-calendar `google-calendar://endpoint-prefix/endpoint?[options]`	Supports interaction with Google Calendar's REST API.
Google Drive / camel-google-drive `google-drive://endpoint-prefix/endpoint?[options]`	Supports interaction with Google Drive's REST API.
Google Mail / camel-google-mail `google-mail://endpoint-prefix/endpoint?[options]`	Supports interaction with Google Mail's REST API.
GMail / camel-gae `gmail://user@g[oogle]mail.com[?options]`	Supports sending of emails via the mail service of Google App Engine. See also Camel Components for Google App Engine.
Gora / `camel-gora` `gora:instanceName[?options]`	Supports to work with NoSQL databases using the Apache Gora framework.
Grape/ `camel-grape` `grape:defaultMavenCoordinates`	Grape component allows you to fetch, load and manage additional jars when CamelContext is running.
Geocoder / `camel-geocoder` `geocoder:<address|latlng:latitude,longitude>[?options]`	Supports looking up geocoders for an address, or reverse lookup geocoders from an address.

Google Guava EventBus / `camel-guava-eventbus` `guava-eventbus:busName[?options]`	The Google Guava EventBus allows publish-subscribe-style communication between components without requiring the components to explicitly register with one another (and thus be aware of each other). This component provides integration bridge between Camel and Google Guava EventBus infrastructure.
Hazelcast / camel-hazelcast `hazelcast://[type]:cachename[?options]`	Hazelcast is a data grid entirely implemented in Java (single jar). This component supports map, multimap, seda, queue, set, atomic number and simple cluster support.
HBase / `camel-hbase` `hbase://table[?options]`	For reading/writing from/to an HBase store (Hadoop database)
HDFS / `camel-hdfs` `hdfs://hostName[:port][/path][?options]`	For reading/writing from/to an HDFS filesystem using Hadoop 1.x
HDFS2 / `camel-hdfs2` `hdfs2://hostName[:port][/path][?options]`	For reading/writing from/to an HDFS filesystem using Hadoop 2.x
Hipchat / `camel-hipchat` `hipchat://[host][:port]?options`	For sending/receiving messages to Hipchat using v2 API
HL7 / `camel-hl7` `mina2:tcp://hostName[:port][?options]`	For working with the HL7 MLLP protocol and the HL7 data format using the HAPI library

Infinispan / `camel-infinispan` `infinispan://cacheName[?options]`	For reading/writing from/to Infinispan distributed key/value store and data grid
HTTP / `camel-http` `http:hostName[:port][/resourceUri][?options]`	For calling out to external HTTP servers using Apache HTTP Client 3.x
HTTP4 / `camel-http4` `http4:hostName[:port][/resourceUri][?options]`	For calling out to external HTTP servers using Apache HTTP Client 4.x
iBATIS / `camel-ibatis` `ibatis://statementName[?options]`	Performs a query, poll, insert, update or delete in a relational database using Apache iBATIS
Ignite / `camel-ignite` `ignite:[cache/compute/messaging/...][?options]`	Apache Ignite In-Memory Data Fabric is a high-performance, integrated and distributed in-memory platform for computing and transacting on large-scale data sets in real-time, orders of magnitude faster than possible with traditional disk-based or flash technologies. It is designed to deliver uncompromised performance for a wide set of in-memory computing use cases from high performance computing, to the industry most advanced data grid, highly available service grid, and streaming.
IMAP / `camel-mail` `imap://[username@]hostName[:port][?options]`	Receiving email using IMAP

IMAPS / `camel-mail` `imaps://[username@]hostName[:port][?options]`	...
IRC / `camel-irc` `irc:[login@]hostName[:port]/#room[?options]`	For IRC communication
IronMQ / `camel-ironmq` `ironmq:queueName[?options]`	For working with IronMQ a elastic and durable hosted message queue as a service.
JavaSpace / `camel-javaspace` `javaspace:jini://hostName[?options]`	Sending and receiving messages through JavaSpace
jBPM / `camel-jbpm` `jbpm:hostName[:port][/resourceUri][?options]`	Sending messages through kie-remote-client API to jBPM.
jcache / `camel-jcache` `jcache:cacheName[?options]`	The JCache component facilitates creation of caching endpoints and processors using JCache / jsr107 as the cache implementation.
jclouds / `camel-jclouds` `jclouds:<blobstore|compute>:[provider id][?options]`	For interacting with cloud compute & blobstore service via jclouds

JCR / `camel-jcr` `jcr://user:password@repository/path/to/node[?options]`	Storing a message in a JCR compliant repository like Apache Jackrabbit
JDBC / `camel-jdbc` `jdbc:dataSourceName[?options]`	For performing JDBC queries and operations
Jetty / `camel-jetty` `jetty:hostName[:port][/resourceUri][?options]`	For exposing or consuming services over HTTP
JGroups / `camel-jgroups` `jgroups:clusterName[?options]`	The `jgroups:` component provides exchange of messages between Camel infrastructure and JGroups clusters.
JIRA / `camel-jira` `jira://endpoint[?options]`	For interacting with JIRA
JMS / `camel-jms` `jms:[queue:|topic:]destinationName[?options]`	Working with JMS providers
JMX / `camel-jmx` `jmx://platform[?options]`	For working with JMX notification listeners
JPA / `camel-jpa` `jpa://entityName[?options]`	For using a database as a queue via the JPA specification for working with OpenJPA, Hibernate or TopLink

JOLT / `camel-jolt` `jolt:specName[?options]`	The **jolt:** component allows you to process a JSON messages using an JOLT specification. This can be ideal when doing JSON to JSON transformation.
Jsch / `camel-jsch` `scp://hostName[:port]/destination[?options]`	Support for the scp protocol
JT/400 / `camel-jt400` `jt400://user:pwd@system/<path_to_dtaq>[?options]`	For integrating with data queues on an AS/400 (aka System i, IBM i, i5, ...) system
Kafka / `camel-kafka` `kafka://server:port[?options]`	For producing to or consuming from Apache Kafka message brokers.
Kestrel / `camel-kestrel` `kestrel://[addresslist/]queueName[?options]`	For producing to or consuming from Kestrel queues
Krati / `camel-krati` `krati://[path to datastore/][?options]`	For producing to or consuming to Krati datastores
Kubernetes / `camel-kubernetes` `kubernetes:masterUrl[?options]`	For integrating your application with Kubernetes standalone or on top of OpenShift.
Kura / `camel-kura`	For deploying Camel OSGi routes into the Eclipse Kura M2M container.

Language / `camel-core` `language://languageName[:script][?options]`	Executes Languages scripts
LDAP / `camel-ldap` `ldap:host[:port][?options]`	Performing searches on LDAP servers (<scope> must be one of object|onelevel|subtree)
LinkedIn / `camel-linkedin` `linkedin://endpoint-prefix/endpoint?[options]`	Component for retrieving LinkedIn user profiles, connections, companies, groups, posts, etc. using LinkedIn REST API.
Log / `camel-core` `log:loggingCategory[?options]`	Uses Jakarta Commons Logging to log the message exchange to some underlying logging system like log4j
Lucene / `camel-lucene` `lucene:searcherName:<insert|query>[?options]`	Uses Apache Lucene to perform Java-based indexing and full text based searches using advanced analysis/tokenization capabilities
Lumberjack / `camel-lumberjack` `lumberjack:host[:port]`	Uses the Lumberjack protocol for retrieving logs (from Filebeat for instance)
Metrics / `camel-metrics` `metrics:[meter|counter|histogram|timer]:metricname[?options]`	Uses Metrics to collect application statistics directly from Camel routes.

MINA / `camel-mina` `mina:[tcp|udp|vm]:host[:port][?options]`	Working with Apache MINA 1.x
MINA2 / `camel-mina2` `mina2:[tcp|udp|vm]:host[:port][?options]`	Working with Apache MINA 2.x
Mock / `camel-core` `mock:name[?options]`	For testing routes and mediation rules using mocks
MLLP / `camel-mllp` `mllp:host:port[?options]`	The MLLP component is specifically designed to handle the nuances of the MLLP protocol and provide the functionality required by Healthcare providers to communicate with other systems using the MLLP protocol
MongoDB / `camel-mongodb` `mongodb:connectionBean[?options]`	Interacts with MongoDB databases and collections. Offers producer endpoints to perform CRUD-style operations and more against databases and collections, as well as consumer endpoints to listen on collections and dispatch objects to Camel routes
MongoDB GridFS / `camel-mongodb-gridfs` `mongodb-gridfs:dbName[?options]`	Sending and receiving files via MongoDB's GridFS system. **Note:** for Camel < 2.19, the URI syntax is gridfs:dbName[?options]
MQTT / `camel-mqtt` `mqtt:name[?options]`	Component for communicating with MQTT M2M message brokers

MSV / `camel-msv` `msv:someLocalOrRemoteResource[?options]`	Validates the payload of a message using the MSV Library
Mustache / `camel-mustache` `mustache:templateName[?options]`	Generates a response using a Mustache template
MVEL / `camel-mvel` `mvel:templateName[?options]`	Generates a response using an MVEL template
MyBatis / `camel-mybatis` `mybatis://statementName[?options]`	Performs a query, poll, insert, update or delete in a relational database using MyBatis
Nagios / `camel-nagios` `nagios://hostName[:port][?options]`	Sending passive checks to Nagios using JSendNSCA
NATS / `camel-nats` `nats://servers[?options]`	For messaging with the NATS platform.
Netty / `camel-netty` `netty:<tcp|udp>//host[:port][?options]`	Working with TCP and UDP protocols using Java NIO based capabilities offered by the Netty project

Netty4 / `camel-netty4` `netty4:<tcp|udp>//host[:port][?options]`	Working with TCP and UDP protocols using Java NIO based capabilities offered by the Netty project
Netty HTTP / `camel-netty-http` `netty-http:http:[port]/context-path[?options]`	Netty HTTP server and client using the Netty project
Netty4 HTTP / `camel-netty4-http` `netty4-http:http:[port]/context-path[?options]`	Netty HTTP server and client using the Netty project 4.x
Olingo2 / `camel-olingo2` `olingo2:endpoint/resource-path[?options]`	Communicates with OData 2.0 services using Apache Olingo 2.0.
Openshift / `camel-openshift` `openshift:clientId[?options]`	To manage your Openshift applications.
OptaPlanner / `camel-optaplanner` `optaplanner:solverConfig[?options]`	Solves the planning problem contained in a message with OptaPlanner.
Paho / `camel-paho` `paho:topic[?options]`	Paho component provides connector for the MQTT messaging protocol using the Paho library.

Pax-Logging / `camel-paxlogging` `paxlogging:appender`	Receiving Pax-Logging events in OSGi
PDF / `camel-pdf` `pdf:operation[?options]`	Allows to work with Apache PDFBox PDF documents
PGEvent / `camel-pgevent` `pgevent:dataSource[?options]`	Allows for Producing/Consuming PostgreSQL events related to the LISTEN/NOTIFY commands added since PostgreSQL 8.3

POP3 / `camel-mail` `pop3s://[username@]hostName port][?options]`	Receiving email using POP3 and JavaMail
POP3S / `camel-mail` `pop3s://[username@]hostName port][?options]`	...
Printer / `camel-printer` `lpr://host:port/path/to/printer[?options]`	The printer component facilitates creation of printer endpoints to local, remote and wireless printers. The endpoints provide the ability to print camel directed payloads when utilized on camel routes.
Properties / `camel-core` `properties://key[?options]`	The properties component facilitates using property placeholders directly in endpoint URI definitions.
Quartz / `camel-quartz` `quartz://groupName/timerName[?options]`	Provides a scheduled delivery of messages using the Quartz 1.x scheduler
Quartz2 / `camel-quartz2` `quartz2://groupName/timerName[?options]`	Provides a scheduled delivery of messages using the Quartz 2.x scheduler

Quickfix / `camel-quickfix` `quickfix:configFile[?options]`	Implementation of the QuickFix for Java engine which allow to send/receive FIX messages
RabbitMQ / `camel-rabbitmq` `rabbitmq://hostname[:port]/exchangeName[?options]`	Component for integrating with RabbitMQ
Ref / `camel-core` `ref:name`	Component for lookup of existing endpoints bound in the Registry.
Rest / `camel-core` `rest:verb:path[?options]`	Component for consuming Restful resources supporting the Rest DSL and plugins to other Camel rest components.
Restlet / `camel-restlet` `restlet:restletUrl[?options]`	Component for consuming and producing Restful resources using Restlet
REST Swagger / **camel-rest-swagger** `rest-swagger:[specificationUri#]operationId[?options]`	Component for accessing REST resources using Swagger specification as configuration.
RMI / `camel-rmi` `rmi://hostName[:port][?options]`	Working with RMI
RNC / `camel-jing` `rnc:/relativeOrAbsoluteUri[?options]`	Validates the payload of a message using RelaxNG Compact Syntax
RNG / `camel-jing` `rng:/relativeOrAbsoluteUri[?options]`	Validates the payload of a message using RelaxNG
Routebox / `camel-routebox` `routebox:routeBoxName[?options]`	Facilitates the creation of specialized endpoints that offer encapsulation and a strategy/map based indirection service to a collection of camel routes hosted in an automatically created or user injected camel context

RSS / `camel-rss` `rss:uri[?options]`	Working with ROME for RSS integration, such as consuming an RSS feed.
Salesforce / `camel-salesforce` `salesforce:topic[?options]`	To integrate with Salesforce
SAP NetWeaver / `camel-sap-netweaver` `sap-netweaver:hostName[:port][?options]`	To integrate with SAP NetWeaver Gateway
Scheduler / `camel-core` `scheduler://name?[options]`	Used to generate message exchanges when a scheduler fires. The scheduler has more functionality than the timer component.
schematron / `camel-schematron` `schematron://path?[options]`	Camel component of Schematron which supports to validate the XML instance documents.
SEDA / `camel-core` `seda:someName[?options]`	Asynchronous call to another endpoint in the same CamelContext
ServiceNow / `camel-servicenow` `servicenow:instanceName[?options]`	Camel component for ServiceNow
SERVLET / `camel-servlet` `servlet:relativePath[?options]`	For exposing services over HTTP through the servlet which is deployed into the Web container.
SFTP / `camel-ftp` `sftp://[username@]hostName[:port]/directoryName[?options]`	Sending and receiving files over SFTP (FTP over SSH).
Sip / `camel-sip` `sip://user@hostName[:port][?options]`	Publish/Subscribe communication capability using the Telecom SIP protocol. RFC3903 - Session Initiation Protocol (SIP) Extension for Event

SIPS / `camel-sip` `sips://user@hostName[:port][?options]`	...
SJMS / `camel-sjms` `sjms:[queue:|topic:]destinationName[?options]`	A ground up implementation of a JMS client
SJMS Batch / `camel-sjms` `sjms-batch:[queue:]destinationName[?options]`	A specialized JMS component for highly-performant transactional batch consumption from a queue.
Slack / `camel-slack` `slack:#channel[?options]`	The **slack** component allows you to connect to an instance of Slack and delivers a message contained in the message body via a pre established Slack incoming webhook .
SMTP / `camel-mail` `smtps://[username@]hostName[:port][?options]`	Sending email using SMTP and JavaMail
SMTP / `camel-mail` `smtps://[username@]hostName[:port][?options]`	...
SMPP / `camel-smpp` `smpp://[username@]hostName[:port][?options]`	To send and receive SMS using Short Messaging Service Center using the JSMPP library
SMPPS / `camel-smpp` `smpps://[username@]hostName[:port][?options]`	...
SNMP / `camel-snmp` `snmp://hostName[:port][?options]`	Polling OID values and receiving traps using SNMP via SNMP4J library

Solr / `camel-solr` `solr://hostName[:port]/solr[?options]`	Uses the Solrj client API to interface with an Apache Lucene Solr server
Apache Spark / `camel-spark` `spark:{rdd|dataframe|hive}[?options]`	Bridges Apache Spark computations with Camel endpoints.
Spark-rest / `camel-spark-rest` `spark-rest://verb:path[?options]`	For easily defining REST services endpoints using Spark REST Java library.
Splunk / `camel-splunk` `splunk://[endpoint][?options]`	For working with Splunk
SpringBatch / `camel-spring-batch` `spring-batch://jobName[?options]`	To bridge Camel and Spring Batch
SpringIntegration / `camel-spring-integration` `spring-integration:defaultChannelName[?options]`	The bridge component of Camel and Spring Integration
Spring LDAP / `camel-spring-ldap` `spring-ldap:springLdapTemplateBean[?options]`	Camel wrapper for Spring LDAP
Spring Redis / `camel-spring-redis` `spring-redis://hostName:port[?options]`	Component for consuming and producing from Redis key-value store Redis
Spring Web Services / `camel-spring-ws` `spring-ws:[mapping-type:]address[?options]`	Client-side support for accessing web services, and server-side support for creating your own contract-first web services using Spring Web Services

SQL / `camel-sql` `sql:select * from table where id=#[?options]`	Performing SQL queries using JDBC
SQL Stored Procedure / `camel-sql` `sql-stored:template[?options]`	Performing SQL queries using Stored Procedure calls
SSH component / `camel-ssh` `ssh:[username[:password]@]hostName[:port][?options]`	For sending commands to a SSH server
StAX / `camel-stax` `stax:(contentHandlerClassName|#myHandler)`	Process messages through a SAX ContentHandler.
Stream / `camel-stream` `stream:[in|out|err|file|header|url][?options]`	Read or write to an input/output/error/file stream rather like unix pipes
Stomp / `camel-stomp` `stomp:queue:destinationName[?options]`	For communicating with Stomp compliant message brokers, like Apache ActiveMQ or ActiveMQ Apollo
StringTemplate / `camel-stringtemplate` `string-template:templateName[?options]`	Generates a response using a String Template
Stub / `camel-core` `stub:someOtherCamelUri[?options]`	Allows you to stub out some physical middleware endpoint for easier testing or debugging
Telegram / `camel-telegram` `telegram://bots/authToken[?options]`	Allows to exchange data with the Telegram messaging network

Test / `camel-spring` `test:expectedMessagesEndpointUri[?options]`	Creates a Mock endpoint which expects to receive all the message bodies that could be polled from the given underlying endpoint
Timer / `camel-core` `timer:timerName[?options]`	Used to generate message exchanges when a timer fires You can only consume events from this endpoint.
Twitter / `camel-twitter` `twitter://endpoint[?options]`	A twitter endpoint
Undertow / `camel-undertow` `undertow://host:port/context-path[?options]`	HTTP server and client using the light-weight Undertow server.
Validation / `camel-core` (camel-spring for Camel 2.8 or older) `validation:someLocalOrRemoteResource[?options]`	Validates the payload of a message using XML Schema and JAXP Validation
Velocity / `camel-velocity` `velocity:templateName[?options]`	Generates a response using an Apache Velocity template
Vertx / `camel-vertx` `vertx:eventBusName`	Working with the vertx event bus
VM / `camel-core` `vm:queueName[?options]`	Asynchronous call to another endpoint in the same JVM
Weather / `camel-weather` `wweather://name[?options]`	Polls the weather information from Open Weather Map
Websocket / `camel-websocket` `websocket://hostname[:port][/resourceUri][?options]`	Communicating with Websocket clients

XML Security / `camel-xmlsecurity` `xmlsecurity:<sign|verify>:name[?options]`	Used to sign and verify exchanges using the XML signature specification.
XMPP / `camel-xmpp` `xmpp://[login@]hostname[:port][/participant][?options]`	Working with XMPP and Jabber
XQuery / `camel-saxon` `xquery:someXQueryResource`	Generates a response using an XQuery template
XSLT / `camel-core` (camel-spring for Camel 2.8 or older) `xslt:templateName[?options]`	Generates a response using an XSLT template
Yammer / `camel-yammer` `yammer://function[?options]`	Allows you to interact with the Yammer enterprise social network
Zookeeper / `camel-zookeeper` `zookeeper://zookeeperServer[:port][/path][?options]`	Working with ZooKeeper cluster(s)

URI's for external components

Other projects and companies have also created Camel components to integrate additional functionality into Camel. These components may be provided under licenses that are not compatible with the Apache License, use libraries that are not compatible, etc... These components are not supported by the Camel team, but we provide links here to help users find the additional functionality.

Component / ArtifactId / URI	License	Description
ActiveMQ / `activemq-camel` `activemq:[queue|topic:]destinationName`	Apache	For JMS Messaging with Apache ActiveMQ.

ActiveMQ Broker / `activemq-camel` `broker:[queue|topic:] destinationName`	Apache	For internal message routing in the ActiveMQ broker using Camel.
Activiti / `activiti-camel` `activiti:camelProcess: serviceTask`	Apache	For working with Activiti, a light-weight workflow and Business Process Management (BPM) platform which supports BPMN 2.
Bluetooth / `camel-bluetooth` / rhiot.io `bluetooth:label`	Apache	Camel Bluetooth component can retrieve information about the Bluetooth devices available within the device range.
Couchbase / `camel-couchbase` / camel-extra `couchbase: protocol://host[:port] /bucket`	Couchbase	Working with Couchbase NoSQL document database.
Db4o / `camel-db4o` / camel-extra `db4o://className`	GPL	For using a db4o datastore as a queue via the db4o library.
Esper / `camel-esper` / camel-extra `esper:name`	GPL	Working with the Esper Library for Event Stream Processing.
Fabric AMQ / `mq-fabric-camel` / fabric8 `amq:[queue|topic:] destinationName`	Apache	The **amq:** endpoint works exactly like the **activemq:** endpoint in Apache Camel; only it uses the fabric to automatically discover the broker. So there is no configuration required; it'll just work out of the box and automatically discover whatever ActiveMQ message brokers are available; with failover and load balancing.
Fabric Fabric / `fabric-camel` / fabric8 `fabric:logicalName: camelEndpointUri`	Apache	The **fabric:** endpoint uses Fabric's discovery mechanism to expose physical sockets, HTTP endpoints, etc. into the runtime registry using a logical name so that clients can use the existing Camel Load Balancer.
Fabric Master / `fabric-camel` / fabric8 `master:clusterName: camelEndpointUri`	Apache	The **master:** endpoint provides a way to ensure only a single consumer in a cluster consumes from a given endpoint; with automatic failover if that JVM dies.

Framebuffer / `camel-framebuffer` / rhiot.io `framebuffer://name`	Apache	Camel Framebuffer component can be used to manage any Linux Framebuffer.
gpsd / `camel-gpsd` / rhiot.io `gpsd:label[?options]`	Apache	Camel GPSD component can be used to read current GPS information from GPS devices.
Hibernate / `camel-hibernate` / camel-extra `hibernate://entityName`	GPL	For using a database as a queue via the Hibernate library.
JBI / `servicemix-camel` `jbi:serviceName`	Apache	For JBI integration such as working with Apache ServiceMix.
JCIFS / `camel-jcifs` / camel-extra `smb://user@server.example.com/sharename?password=secret&localWorkDirectory=/tmp`	LGPL	This component provides access to remote file systems over the CIFS/SMB networking protocol by using the JCIFS library.
kura-cloud / `camel-kura` / rhiot.io `kura-wifi:networkInterface/ssid`	Apache	Camel Kura Cloud component interacts directly with Kura CloudService.
kura-wifi / `camel-kura` / rhiot.io `kura-wifi:networkInterface/ssid`	Apache	Camel Kura WiFi component can be used to retrieve the information about the WiFi access spots available within the device range.
NMR / `servicemix-nmr` `nmr://serviceName`	Apache	Integration with the Normalized Message Router BUS in ServiceMix 4.x.
OpenIMAJ / `camel-openimaj` / rhiot.io `pi4j-gpio://gpioId[?options]`	Apache	Camel OpenIMAJ component can be used to detect faces in images.
pi4j-gpio / `camel-pi4j` / rhiot.io `pi4j-gpio://gpioId[?options]`	Apache	GPIO Component for RaspberryPi based on pi4j lib.

pi4j-i2c / `camel-pi4j` / rhiot.io `pi4j-i2c://busId /deviceId[?options]`	Apache	i2c Component for RaspberryPi based on pi4j lib.
PubNub / `camel-pubnub` / rhiot.io `pubnub://pubnubEndpoint Type:channel[?options]`	Apache	Camel PubNub component. More information rhiot.io project.
RCode / `camel-rcode` / camel-extra `rcode://host[:port] /operation[?options]`	LGPL	Uses Rserve to integrate Camel with the statistics environment R.
Scalate / `scalate-camel` `scalate:templateName`	Apache	Uses the given Scalate template to transform the message.
Smooks / `camel-smooks` / camel-extra `unmarshal(edi)`	GPL	For working with EDI parsing using the Smooks library. This component is **deprecated** as Smooks now provides Camel integration out of the box.
Spring Neo4j / `camel-spring-neo4j` / camel-extra `spring-neo4j: http://hostname[:port] /database[?options]`	TBA	Component for producing to Neo4j datastore using the Spring Data Neo4j library.
Tinkerforge / `camel-tinkerforge` / rhiot.io `tinkerforge:[//hostname [:port]]/devicetype/uid/ [?options]`	Apache	The tinkerforge component allows interaction with Tinkerforge bricklets. It uses the standard Java bindings to connects to `brickd`. For more information see the rhiot.io.
VirtualBox / `camel-virtualbox` / camel-extra `virtualbox:machine[? options]`	GPL V2	The VitualBox component uses the webservice API that exposes VirtualBox functionality and consumes events generated by virtual machines.
Webcam / `camel-webcam` / rhiot.io `webcam:label[?options]`	Apache	Camel Webcam component can be used to capture still images and detect motion.

ZeroMQ / `camel-zeromq` / camel-extra `zeromq:(tcp|ipc)://hostname:port`	LGPL	The ZeroMQ component allows you to consumer or produce messages using ZeroMQ.