

# Timer

## Timer Component

The **timer** component is used to generate message exchanges when a timer fires. You can only consume events from this endpoint.

### URI format

```
timer:name[?options]
```

Where `name` is the name of the `Timer` object, which is created and shared across endpoints. So if you use the same name for all your timer endpoints, only one `Timer` object and thread will be used.

You can append query options to the URI in the following format, `?option=value&option=value&...`

**Note:** The `IN` body of the generated exchange is `null`. So `exchange.getIn().getBody()` returns `null`.



#### Advanced Scheduler

See also the [Quartz](#) component that supports much more advanced scheduling.



#### Specify time in human friendly format

In **Camel 2.3** onwards you can specify the time in [human friendly syntax](#).

## Options

Name	Default Value	Description
time	null	A <code>java.util.Date</code> the <b>first</b> event should be generated. If using the URI, the pattern expected is: <code>yyyy-MM-dd HH:mm:ss</code> or <code>yyyy-MM-dd'T'HH:mm:ss</code> .
pattern	null	Allows you to specify a custom <code>Date</code> pattern to use for setting the time option using URI syntax.
period	1000	If greater than 0, generate periodic events every <code>period</code> milliseconds. You can also specify time values using units, such as 60s (60 seconds), 5m30s (5 minutes and 30 seconds), and 1h (1 hour).
delay	0 / 1000	The number of milliseconds to wait before the first event is generated. Should not be used in conjunction with the <code>time</code> option. You can also specify time values using units, such as 60s (60 seconds), 5m30s (5 minutes and 30 seconds), and 1h (1 hour). Before <b>Camel 2.11</b> the default value is 0 From <b>Camel 2.11</b> the default value is 1000 From <b>Camel 2.17</b> it is possible to specify a negative delay. In this scenario the timer will generate and fire events as soon as possible.
fixedRate	false	Events take place at approximately regular intervals, separated by the specified period.
daemon	true	Specifies whether or not the thread associated with the timer endpoint runs as a daemon.
repeatCount	0	<b>Camel 2.8:</b> Specifies a maximum limit of number of fires. So if you set it to 1, the timer will only fire once. If you set it to 5, it will only fire five times. A value of zero or negative means fire forever.

## Exchange Properties

When the timer is fired, it adds the following information as properties to the `Exchange`:

Name	Type	Description
<code>Exchange.TIMER_NAME</code>	<code>String</code>	The value of the <code>name</code> option.
<code>Exchange.TIMER_TIME</code>	<code>Date</code>	The value of the <code>time</code> option.
<code>Exchange.TIMER_PERIOD</code>	<code>long</code>	The value of the <code>period</code> option.
<code>Exchange.TIMER_FIRED_TIME</code>	<code>Date</code>	The time when the consumer fired.

Exchange.TIMER_COUNTER	Long	<b>Camel 2.8:</b> The current fire counter. Starts from 1.
------------------------	------	--

## Message Headers

When the timer is fired, it adds the following information as headers to the IN message

Name	Type	Description
Exchange.TIMER_FIRED_TIME	java.util.Date	The time when the consumer fired

## Sample

To set up a route that generates an event every 60 seconds:

```
from( "timer://foo?fixedRate=true&period=60000" ).to( "bean:myBean?method=someMethodName" );
```



Instead of 60000 you can use period=60s which is more friendly to read.

The above route will generate an event and then invoke the `someMethodName` method on the bean called `myBean` in the [Registry](#) such as JNDI or [Spring](#).

And the route in Spring DSL:

```
<route>
  <from uri="timer://foo?fixedRate=true&period=60000"/>
  <to uri="bean:myBean?method=someMethodName"/>
</route>
```

## Firing as soon as possible

### Available as of Camel 2.17

You may want to fire messages in a Camel route as soon as possible you can use a negative delay:

```
<route>
  <from uri="timer://foo?delay=-1"/>
  <to uri="bean:myBean?method=someMethodName"/>
</route>
```

In this way the timer will fire messages immediately.

You can also specify a `repeatCount` parameter in conjunction with a negative delay to stop firing messages after a fixed number has been reached.

If you don't specify a `repeatCount` then the timer will continue firing messages until the route will be stopped.

## Firing only once

### Available as of Camel 2.8

You may want to fire a message in a Camel route only once, such as when starting the route. To do that you use the `repeatCount` option as shown:

```
<route>
  <from uri="timer://foo?repeatCount=1"/>
  <to uri="bean:myBean?method=someMethodName"/>
</route>
```

## See Also

- [Configuring Camel](#)
- [Component](#)
- [Endpoint](#)
- [Getting Started](#)

- Quartz