Hibernate

Hibernate Component

The **hibernate:** component allows you to work with databases using Hibernate as the object relational mapping technology to map POJOs to database tables. The **camel-hibernate** library is provided by the Camel Extra project which hosts all *GPL related components for Camel.

> Note that Camel also ships with a JPA component. The JPA component abstracts from the underlying persistence provider and allows you to work with Hibernate, OpenJPA or EclipseLink.

Sending to the endpoint

Sending POJOs to the hibernate endpoint inserts entities into the database. The body of the message is assumed to be an entity bean that you have mapped to a relational table using the hibernate `.hbm.xml` files.

If the body does not contain an entity bean, use a Message Translator in front of the endpoint to perform the necessary conversion first.

Consuming from the endpoint

Consuming messages removes (or updates) entities in the database. This allows you to use a database table as a logical queue; consumers take messages from the queue and then delete/update them to logically remove them from the queue.

If you do not wish to delete the entity when it has been processed, you can specify `consumeDelete=false` on the URI. This will result in the entity being processed each poll.

If you would rather perform some update on the entity to mark it as processed (such as to exclude it from a future query) then you can annotate a method with @Consumed which will be invoked on your entity bean when the entity bean is consumed.

URI format

```
hibernate:[entityClassName][?options]
```

For sending to the endpoint, the **entityClassName** is optional. If specified it is used to help use the type conversion to ensure the body is of the correct type.

For consuming the **entityClassName** is mandatory.

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

Options

Name	Default Value	Description
`entityType`	*entityClassName*	Is the provided *entityClassName* from the URI.
`consumeDelete`	`true`	Option for HibernateConsumer only. Specifies whether or not the entity is deleted after it is consumed.
`consumeLockEntity`	`true`	Option for HibernateConsumer only. Specifies whether or not to use exclusive locking of each entity while processing the results from the pooling.
`flushOnSend`	`true`	Option for HibernateProducer only. Flushes the EntityManager after the entity bean has been persisted.
`maximumResults`	`-1`	Option for HibernateConsumer only. Set the maximum number of results to retrieve on the Query.
`consumer.delay`	`500`	Option for HibernateConsumer only. Delay in millis between each poll.
`consumer.initialDelay`	`1000`	Option for HibernateConsumer only. Millis before polling starts.
`consumer.userFixedDelay`	`false`	Option for HibernateConsumer only. Set to `true` to use fixed delay between polls, otherwise fixed rate is used. See ScheduledExecutorService in JDK for details.

See Also

- Configuring Camel
- Component
- Endpoint
- Getting Started

- Hibernate Example