
kombu-stomp Documentation

Release 0.0.0dev1

NTTE-AM OSS

October 03, 2014

1	Introduction	1
1.1	Limitations	1
2	Indices and tables	3
	Python Module Index	5

Introduction

This project is an effort for adding STOMP protocol support to Kombu, mostly Celery oriented. You can find documentation at [Read the docs](#).

1.1 Limitations

Currently we offer very limited support:

- ActiveMQ is the only one broker supported.
- We support only STOMP 1.0 protocol.
- No PyPy, Jython support.
- There is no transport options support but the host, port and credentials.

Contents:

1.1.1 API reference

`kombu_stomp`

`kombu_stomp.register_transport()`

Register STOMP transport with Kombu.

You will have to manually call this function before being able to use STOMP protocol as Kombu transport.

`kombu_stomp.stomp`

class `kombu_stomp.stomp.Connection(*args, **kwargs)`

Connection object used by kombu-stomp

class `kombu_stomp.stomp.MessageListener(q=None)`

stomp.py listener used by kombu-stomp

iterator (*timeout*)

Return a Python generator consuming received messages.

If we try to consume a message and there is no messages remaining, then an exception will be raised.

Parameters *timeout* (*int*) – Time to wait for message in seconds, a falsy value if we shouldn't block for incoming messages.

Yields dict A dictionary representing the message in a Kombu compatible format.

Raises `Queue.Empty` When there is no message to be consumed.

on_message (*headers, body*)

Received message hook.

Parameters

- **headers** – message headers.
- **body** – message body.

queue_from_destination (*destination*)

Get the queue name from a destination header value.

to_kombu_message (*headers, body*)

Get STOMP headers and body message and return a Kombu message dict.

Parameters

- **headers** – message headers.
- **body** – message body.

Return dict A dictionary that Kombu can use for creating a new message object.

kombu_stomp.transport

class kombu_stomp.transport.**Channel** (**args, **kwargs*)

kombu-stomp channel class.

conn_or_acquire (**args, **kws*)

Use current connection or create a new one.

stomp_conn

Property over the stomp.py connection object.

It will create the connection object at first use.

class kombu_stomp.transport.**Message** (*channel, raw_message*)

Kombu virtual transport message class for kombu-stomp.

This class extends `kombu.transport.virtual.Message`, so it keeps STOMP message ID for later use.

class kombu_stomp.transport.**QoS** (**args, **kwargs*)

Kombu quality of service class for kombu-stomp.

class kombu_stomp.transport.**Transport** (*client, **kwargs*)

Transport class for kombu-stomp.

Indices and tables

- *genindex*
- *modindex*
- *search*

k

`kombu_stomp`, [1](#)

`kombu_stomp.stomp`, [1](#)

`kombu_stomp.transport`, [2](#)

C

Channel (class in kombu_stomp.transport), 2
conn_or_acquire() (kombu_stomp.transport.Channel
method), 2
Connection (class in kombu_stomp.stomp), 1

I

iterator() (kombu_stomp.stomp.MessageListener
method), 1

K

kombu_stomp (module), 1
kombu_stomp.stomp (module), 1
kombu_stomp.transport (module), 2

M

Message (class in kombu_stomp.transport), 2
MessageListener (class in kombu_stomp.stomp), 1

O

on_message() (kombu_stomp.stomp.MessageListener
method), 2

Q

QoS (class in kombu_stomp.transport), 2
queue_from_destination()
(kombu_stomp.stomp.MessageListener
method), 2

R

register_transport() (in module kombu_stomp), 1

S

stomp_conn (kombu_stomp.transport.Channel attribute),
2

T

to_kombu_message() (kombu_stomp.stomp.MessageListener
method), 2
Transport (class in kombu_stomp.transport), 2