Eiffellib Documentation

Oct 01, 2021

Contents

Eiffellib is a python library for subscribing to and publishing Eiffel events to a message-broker.

CHAPTER 1

Contents

1.1 Eiffellib

Eiffellib is a python library for subscribing to and publishing Eiffel events to a message-broker.

1.1.1 Description

Eiffellib solves the problem of publishing Eiffel events and subscribing to events, removing the need of knowing how to connect to a message-broker or how to utilize the protocol it supplies.

With Eiffellib you can start subscribing to and publish valid Eiffel messages quickly and to get a feel for the event protocol.

It is designed to be fast and easy to start using while still being production quality.

Documentation: https://eiffellib.readthedocs.io/en/latest/

1.1.2 Features

- Simple subscription and publishing of Eiffel events.
- Event building assistance with event validation on receive and publish.
- Following a context link.

1.1.3 Installation

Install the project by running:

```
pip install eiffellib[rabbitmq]
```

If you only want to use the Eiffel message definitions leave out the optional dependency: pip install eiffellib

1.1.4 Examples

Start RabbitMQ

In order for these examples to work you need a RabbitMQ server:

Subscribing to an event

```
import time
from eiffellib.subscribers import RabbitMQSubscriber


def callback(event, context):
    print(event.pretty)

SUBSCRIBER = RabbitMQSubscriber(host="127.0.0.1", port=5672, ssl=False,
                                queue="eiffel", exchange="amq.fanout")
SUBSCRIBER.subscribe("EiffelActivityTriggeredEvent", callback)
SUBSCRIBER.start()
while True:
    time.sleep(0.1)
```

Publishing an event

```
from eiffellib.publishers import RabbitMQPublisher
from eiffellib.events import EiffelActivityTriggeredEvent

PUBLISHER = RabbitMQPublisher(host="127.0.0.1", exchange="amq.fanout", ssl=False,
                              port=5672, routing_key=None)
PUBLISHER.start()
ACTIVITY_TRIGGERED = EiffelActivityTriggeredEvent()
ACTIVITY_TRIGGERED.data.add("name", "Test activity")
PUBLISHER.send_event(ACTIVITY_TRIGGERED)
```

Deprecation of routing key

The "routing_key" argument in the RabbitMQPublisher class has been deprecated.

This deprecation also affects the default value of the "routing_key" argument and you will be getting warnings while running.

The reason for this change is due to a misunderstanding of how routing keys are supposed to be used when eiffellib was first created.

Each event will now be able to generate their own routing key every time the event is sent.

This routing key is by default "eiffel._.$event_type._._" where the different values are "eiffel.$family.$event_type.$tag.$domainid".

Please refer to https://eiffel-community.github.io/eiffel-sepia/rabbitmq-message-broker.html for more information about routing keys.

To change to the new routing key behavior (and thus removing the warning), please set "routing_key" to "None" when initializing a new RabbitMQPublisher.

```
PUBLISHER = RabbitMQPublisher(host="127.0.0.1", exchange="amq.fanout", ssl=False,
                              port=5672, routing_key=None)
```

In order to change "$family", "$tag" or "$domainid" in the routing key, they have to be set on the events.

```
PUBLISHER = RabbitMQPublisher(host="127.0.0.1", exchange="amq.fanout", ssl=False,
                              port=5672, routing_key=None)
EVENT = EiffelActivityTriggeredEvent(family="myfamily", tag="mytag", domain_id=
↪"mydomain")
PUBLISHER.send_event(EVENT)
```

1.1.5 Contribute

- Issue Tracker: https://github.com/eiffel-community/eiffel-pythonlib/issues
- Source Code: https://github.com/eiffel-community/eiffel-pythonlib

1.1.6 Support

If you are having issues, please let us know. There is a mailing list at: eiffel-pythonlib-maintainers@google-groups.com or just write an Issue.

1.2 Examples

1.2.1 Start RabbitMQ

In order for these examples to work you need a RabbitMQ server:

```
# From https://hub.docker.com/_/rabbitmq
docker run -d --hostname my-rabbit --name some-rabbit -p 8080:15672 -p 5672:5672␣
↪rabbitmq:3-management
```

1.2.2 Pub/Sub

This code snippet will subscribe to an ActivityStarted in order to publish an ActivityFinished

1. Set up a subscriber for the 'EiffelActivityStartedEvent'.
2. Publish an 'EiffelActivityTriggeredEvent' and 'EiffelActivityStartedEvent'.
3. Callback for 'EiffelActivityStartedEvent' is called.
4. Publish an 'EiffelActivityFinishedEvent'

```
import time
from eiffellib.subscribers import RabbitMQSubscriber
from eiffellib.publishers import RabbitMQPublisher
from eiffellib.events import (EiffelActivityTriggeredEvent,
                              EiffelActivityStartedEvent,
                              EiffelActivityFinishedEvent)

def callback(event, context):
```

(continues on next page)

(continued from previous page)

```
    """Fetch EiffelActivityTriggeredEvent ID from links and send␣
↪EiffelActivityFinishedEvent."""
    activity_triggered = None
    for link in event.links.links:
        if link.get("type") == "ACTIVITY_EXECUTION":
            activity_triggered = link.get("target")
            break
    else:
        print(event.pretty)
        raise Exception("No ACTIVITY_EXECUTION link on EiffelActivityStartedEvent")

    # https://github.com/eiffel-community/eiffel/blob/master/eiffel-vocabulary/
↪EiffelActivityFinishedEvent.md
    activity_finished = EiffelActivityFinishedEvent()
    activity_finished.data.add("outcome", {"conclusion": "SUCCESSFUL"})
    activity_finished.links.add("ACTIVITY_EXECUTION", activity_triggered)
    PUBLISHER.send_event(activity_finished)

SUBSCRIBER = RabbitMQSubscriber(host="127.0.0.1", queue="pubsub", exchange="amq.fanout
↪",
                                ssl=False, port=5672)
PUBLISHER = RabbitMQPublisher(host="127.0.0.1", exchange="amq.fanout", port=5672,␣
↪ssl=False,
                              routing_key=None)

SUBSCRIBER.subscribe("EiffelActivityStartedEvent", callback)
SUBSCRIBER.start()
PUBLISHER.start()

# https://github.com/eiffel-community/eiffel/blob/master/eiffel-vocabulary/
↪EiffelActivityTriggeredEvent.md
ACTIVITY_TRIGGERED = EiffelActivityTriggeredEvent()
ACTIVITY_TRIGGERED.data.add("name", "Pubsub activity")
PUBLISHER.send_event(ACTIVITY_TRIGGERED)

# https://github.com/eiffel-community/eiffel/blob/master/eiffel-vocabulary/
↪EiffelActivityStartedEvent.md
ACTIVITY_STARTED = EiffelActivityStartedEvent()
ACTIVITY_STARTED.links.add("ACTIVITY_EXECUTION", ACTIVITY_TRIGGERED)
PUBLISHER.send_event(ACTIVITY_STARTED)

# Wait for event to be received by 'callback'.
time.sleep(1)
```

1.2.3 Activity

How to utilize an `eiffellib.activity.Activity`

An activity is just a callable which will send ActivityTriggered, Started and Finished.

```
import os
import time
from eiffellib.subscribers import RabbitMQSubscriber
from eiffellib.publishers import RabbitMQPublisher
from eiffellib.events import EiffelAnnouncementPublishedEvent
```

(continues on next page)

(continued from previous page)

```
from eiffellib.activity import Activity

class MyActivity(Activity):

    def pre_call(self, event, context):
        print("Activity has triggered.")

    def call(self, event, context):
        print("Activity has started. Let's do stuff.")

    def post_call(self, event, context):
        print("Activity has finished.")

SUBSCRIBER = RabbitMQSubscriber(host="127.0.0.1", queue="activity", exchange="amq.
↪fanout",
                                ssl=False, port=5672)
PUBLISHER = RabbitMQPublisher(host="127.0.0.1", exchange="amq.fanout", port=5672,␣
↪ssl=False,
                              routing_key=None)

SOURCE = {"host": os.getenv("HOSTNAME", "hostname"), "name": "MyActivity",
          "domainId": "example"}
MY_ACTIVITY = MyActivity("Name of activity", PUBLISHER, SOURCE)
SUBSCRIBER.subscribe("EiffelAnnouncementPublishedEvent", MY_ACTIVITY)
SUBSCRIBER.start()
PUBLISHER.start()

# https://github.com/eiffel-community/eiffel/blob/master/eiffel-vocabulary/
↪EiffelAnnouncementPublishedEvent.md
ANNOUNCEMENT = EiffelAnnouncementPublishedEvent()
ANNOUNCEMENT.data.add("heading", "My activity will now trigger")
ANNOUNCEMENT.data.add("body", "This is just a quick trigger for my activity")
ANNOUNCEMENT.data.add("severity", "MINOR")
PUBLISHER.send_event(ANNOUNCEMENT)

# Wait for event to be received by 'callback'.
time.sleep(1)
```

1.3 License

```
                              Apache License
                        Version 2.0, January 2004
                     http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

   "License" shall mean the terms and conditions for use, reproduction,
   and distribution as defined by Sections 1 through 9 of this document.

   "Licensor" shall mean the copyright owner or entity authorized by
   the copyright owner that is granting the License.
```

(continues on next page)

(continued from previous page)

```
   "Legal Entity" shall mean the union of the acting entity and all
   other entities that control, are controlled by, or are under common
   control with that entity. For the purposes of this definition,
   "control" means (i) the power, direct or indirect, to cause the
   direction or management of such entity, whether by contract or
   otherwise, or (ii) ownership of fifty percent (50%) or more of the
   outstanding shares, or (iii) beneficial ownership of such entity.

   "You" (or "Your") shall mean an individual or Legal Entity
   exercising permissions granted by this License.

   "Source" form shall mean the preferred form for making modifications,
   including but not limited to software source code, documentation
   source, and configuration files.

   "Object" form shall mean any form resulting from mechanical
   transformation or translation of a Source form, including but
   not limited to compiled object code, generated documentation,
   and conversions to other media types.

   "Work" shall mean the work of authorship, whether in Source or
   Object form, made available under the License, as indicated by a
   copyright notice that is included in or attached to the work
   (an example is provided in the Appendix below).

   "Derivative Works" shall mean any work, whether in Source or Object
   form, that is based on (or derived from) the Work and for which the
   editorial revisions, annotations, elaborations, or other modifications
   represent, as a whole, an original work of authorship. For the purposes
   of this License, Derivative Works shall not include works that remain
   separable from, or merely link (or bind by name) to the interfaces of,
   the Work and Derivative Works thereof.

   "Contribution" shall mean any work of authorship, including
   the original version of the Work and any modifications or additions
   to that Work or Derivative Works thereof, that is intentionally
   submitted to Licensor for inclusion in the Work by the copyright owner
   or by an individual or Legal Entity authorized to submit on behalf of
   the copyright owner. For the purposes of this definition, "submitted"
   means any form of electronic, verbal, or written communication sent
   to the Licensor or its representatives, including but not limited to
   communication on electronic mailing lists, source code control systems,
   and issue tracking systems that are managed by, or on behalf of, the
   Licensor for the purpose of discussing and improving the Work, but
   excluding communication that is conspicuously marked or otherwise
   designated in writing by the copyright owner as "Not a Contribution."

   "Contributor" shall mean Licensor and any individual or Legal Entity
   on behalf of whom a Contribution has been received by Licensor and
   subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of
   this License, each Contributor hereby grants to You a perpetual,
   worldwide, non-exclusive, no-charge, royalty-free, irrevocable
   copyright license to reproduce, prepare Derivative Works of,
   publicly display, publicly perform, sublicense, and distribute the
   Work and such Derivative Works in Source or Object form.
```

(continues on next page)

(continued from previous page)

```
3. Grant of Patent License. Subject to the terms and conditions of
   this License, each Contributor hereby grants to You a perpetual,
   worldwide, non-exclusive, no-charge, royalty-free, irrevocable
   (except as stated in this section) patent license to make, have made,
   use, offer to sell, sell, import, and otherwise transfer the Work,
   where such license applies only to those patent claims licensable
   by such Contributor that are necessarily infringed by their
   Contribution(s) alone or by combination of their Contribution(s)
   with the Work to which such Contribution(s) was submitted. If You
   institute patent litigation against any entity (including a
   cross-claim or counterclaim in a lawsuit) alleging that the Work
   or a Contribution incorporated within the Work constitutes direct
   or contributory patent infringement, then any patent licenses
   granted to You under this License for that Work shall terminate
   as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the
   Work or Derivative Works thereof in any medium, with or without
   modifications, and in Source or Object form, provided that You
   meet the following conditions:

   (a) You must give any other recipients of the Work or
       Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices
       stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works
       that You distribute, all copyright, patent, trademark, and
       attribution notices from the Source form of the Work,
       excluding those notices that do not pertain to any part of
       the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its
       distribution, then any Derivative Works that You distribute must
       include a readable copy of the attribution notices contained
       within such NOTICE file, excluding those notices that do not
       pertain to any part of the Derivative Works, in at least one
       of the following places: within a NOTICE text file distributed
       as part of the Derivative Works; within the Source form or
       documentation, if provided along with the Derivative Works; or,
       within a display generated by the Derivative Works, if and
       wherever such third-party notices normally appear. The contents
       of the NOTICE file are for informational purposes only and
       do not modify the License. You may add Your own attribution
       notices within Derivative Works that You distribute, alongside
       or as an addendum to the NOTICE text from the Work, provided
       that such additional attribution notices cannot be construed
       as modifying the License.

   You may add Your own copyright statement to Your modifications and
   may provide additional or different license terms and conditions
   for use, reproduction, or distribution of Your modifications, or
   for any such Derivative Works as a whole, provided Your use,
   reproduction, and distribution of the Work otherwise complies with
   the conditions stated in this License.
```

(continues on next page)

(continued from previous page)

```
5. Submission of Contributions. Unless You explicitly state otherwise,
   any Contribution intentionally submitted for inclusion in the Work
   by You to the Licensor shall be under the terms and conditions of
   this License, without any additional terms or conditions.
   Notwithstanding the above, nothing herein shall supersede or modify
   the terms of any separate license agreement you may have executed
   with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade
   names, trademarks, service marks, or product names of the Licensor,
   except as required for reasonable and customary use in describing the
   origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or
   agreed to in writing, Licensor provides the Work (and each
   Contributor provides its Contributions) on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
   implied, including, without limitation, any warranties or conditions
   of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A
   PARTICULAR PURPOSE. You are solely responsible for determining the
   appropriateness of using or redistributing the Work and assume any
   risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,
   whether in tort (including negligence), contract, or otherwise,
   unless required by applicable law (such as deliberate and grossly
   negligent acts) or agreed to in writing, shall any Contributor be
   liable to You for damages, including any direct, indirect, special,
   incidental, or consequential damages of any character arising as a
   result of this License or out of the use or inability to use the
   Work (including but not limited to damages for loss of goodwill,
   work stoppage, computer failure or malfunction, or any and all
   other commercial damages or losses), even if such Contributor
   has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing
   the Work or Derivative Works thereof, You may choose to offer,
   and charge a fee for, acceptance of support, warranty, indemnity,
   or other liability obligations and/or rights consistent with this
   License. However, in accepting such obligations, You may act only
   on Your own behalf and on Your sole responsibility, not on behalf
   of any other Contributor, and only if You agree to indemnify,
   defend, and hold each Contributor harmless for any liability
   incurred by, or claims asserted against, such Contributor by reason
   of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

   To apply the Apache License to your work, attach the following
   boilerplate notice, with the fields enclosed by brackets "[]"
   replaced with your own identifying information. (Don't include
   the brackets!)  The text should be enclosed in the appropriate
   comment syntax for the file format. We also recommend that a
   file or class name and description of purpose be included on the
```

(continues on next page)

(continued from previous page)

```
   same "printed page" as the copyright notice for easier
   identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```

1.4 Authors

1.4.1 Maintainers

- Tobias Persson <tobiaspn@axis.com>
- Fredrik Fristedt <fredjn@axis.com>

1.4.2 Contributors

- Tobias Persson <tobiaspn@axis.com>
- Fredrik Fristedt <fredjn@axis.com>

1.5 Changelog

1.5.1 Version 2.2.0

- Add sepia routing keys to eiffellib

1.5.2 Version 2.1.0

- Specify open dependencies

1.5.3 Version 2.0.1

- Version 2.0.0 did not work as expected. This fixes the problem with 2.0.0

1.5.4 Version 2.0.0

- Make the pika dependency optional

1.5.5 Version 1.2.0

- Update event schemas
- Add badge to eiffel-pythonlib

1.5.6 Version 1.1.1

- Add wildcard support to nackable subscriptions

1.5.7 Version 1.1.0

- If Context is None, call a subscriber with None on Context.
- We should use the python standard library's logging module to log messages, not print or traceback.print_exc()
- Reconnect the connection thread if broker shuts down

1.5.8 Version 1.0.2

- Fix examples
- Add Fredrik as contributor
- Correct library name for the package when building wheel
- Re-added examples without unsupported rst options

1.5.9 Version 1.0.1

- 1.0.0 did not upload correctly to pypi.

1.5.10 Version 1.0.0

- Eiffel library for sending and retrieving events.

1.6 eiffellib

1.6.1 eiffellib package

Subpackages

eiffellib.events package

Submodules

eiffellib.events.eiffel_activity_canceled_event module

eiffellib.events.eiffel_activity_finished_event module

eiffellib.events.eiffel_activity_started_event module

eiffellib.events.eiffel_activity_triggered_event module

eiffellib.events.eiffel_announcement_published_event module

eiffellib.events.eiffel_artifact_created_event module

eiffellib.events.eiffel_artifact_published_event module

eiffellib.events.eiffel_artifact_reused_event module

eiffellib.events.eiffel_base_event module

eiffellib.events.eiffel_composition_defined_event module

eiffellib.events.eiffel_confidence_level_modified_event module

eiffellib.events.eiffel_environment_defined_event module

eiffellib.events.eiffel_flow_context_defined_event module

eiffellib.events.eiffel_issue_defined_event module

eiffellib.events.eiffel_issue_verified_event module

eiffellib.events.eiffel_source_change_created_event module

eiffellib.events.eiffel_source_change_submitted_event module

eiffellib.events.eiffel_test_case_canceled_event module

eiffellib.events.eiffel_test_case_finished_event module

eiffellib.events.eiffel_test_case_started_event module

eiffellib.events.eiffel_test_case_triggered_event module

eiffellib.events.eiffel_test_execution_recipe_collection_created_event module

Python Module Index

e

Index

E