Event Processor

search

entry /
  scout.producer = PikaTopicPublisherMaker(
    ip_address=scout.ip_address,
    routing_key=scout.routing_key,
    exchange_name=scout.exchange_name,
    connection_attempts=RabbitConsumerScout.CONNECTION_ATTEMPTS,
    callback_tempo=RabbitConsumerScout.SCOUT_TEMPO_SEC)
  scout.subscribe(Event(signals.REFACTOR_SEARCH))

AMQP_CONSUMER_CHECK as e /
  if scout.live_trace or scout.spy_trace:
    pp(e.payload)

producer_thread_engaged

entry /
  scout.producer.start_thread()
  scout.post_fifo(Event(
    signal=signals.try_to_contact_consumer),
    times=1,
    period=0.2,
    deferred=True)

exit /
  scout.cancel_events(
    Event(signal=signals.try_to_contact_consumer))
  scout.producer.stop_thread()

producer_post_and_wait

entry /
  scout.producer.thread.post_fifo(..)
  scout.post_fifo(Event(
    signal=signals.consumer_test_complete),
    times=1,
    period=0.5,
    deferred=True)

try_to_connect_to_consumer

consumer_test_complete

[scout.producer.connect_error]

AMQP_CONSUMER_SERVER_FOUND

entry /
  scout.publish(
    signal=signals.AMQP_CONSUMER_CHECK,
    payload=(chart.ip, True)

AMQP_CONSUMER_CHECK(ip_address, TRUE, routing_key, exchange_name)

NO_AMQP_CONSUMER_SERVER_FOUND

entry /
  scout.publish(
    Event(signal=signals.AMQP_CONSUMER_CHECK),
    payload=(scout.ip_address,False)

AMQP_CONSUMER_CHECK(ip_address, FALSE, routing_key, exchange_name)

REFACTOR_SEARCH as e /
  if 'ip_address' in e.payload and scout.name is e.payload['ip_address']:
    for item in ['routing_key', 'exchange_name']:
      if item in payload:
        setattr(scout, item, payload['item'])