External Services integration with the Reviews Platform (Release: 13-04-2015)

Objective

Various Systems can subscribe to the RnR event bus to listen to events specific to the Reviews and Ratings System. The listening system, can then take various actions based on the events received.

Overview of the RnR Event bus

The RnR event bus composes of Apache Kafka - "Kafka is a distributed, partitioned, replicated commit log service. It provides the functionality of a messaging system". Events are generated by the RnR Event Client and put into one or more Kafka queues. Listeners who have subscribed to these queues will get notifications. To know more on Apache Kafka refer http://kafka.apache.org/documentation.html#introduction

Events that the Listeners can listen

S.no	Event Type	Event Type Code	Scenario	Event data
1	ReviewCreated	0	whenever a review posted by the customer	reviewid, ownerid,pogid
2	ReviewVoted	3	User voted for a existing review	reviewid,ownerid,userid,voteType
3	ReviewApproved	1	Whenever moderator approved review	reviewid,moderatorid,ownerid,pogid
4	ReviewRejected	2	Whenever ever moderator rejected review	reviewid,moderatorid,ownerid,pogid
5	RatingCreated	4	Whenever a rating is created	userId, ownerId, type, reviewId, rating
6	RatingDeleted	6	Whenever a rating is deleted	userId, ownerId, type, reviewId, rating
7	UpdateRecommendation	7	Whenever recommendation is updated	userId, ownerId, type, reviewId, rating, recommended
8	RemoveRecommendation	8	Whenever a recommendation is removed	userId, ownerId, type, reviewId, rating, recommended
9	CreateRecommendation	9	Whenever a recommendation is created	userId, ownerId, type, reviewId, rating, recommended
10	ReviewUpvoted	10	Whenever a review is voted UP	userId, ownerId, type, reviewId, rating, voteType
11	ReviewDownvoted	11	Whenever a review is voted DOWN	userId, ownerId, type, reviewId, rating, voteType
12	ReviewCheckedOut	12	Whenever a review is checkedout	userId, ownerId, type, reviewId, rating, autoCheckinPeriod
13	ReviewFlagged	13	Whenever a review is flagged (SPAM/ABUSE)	userId, ownerId, type, reviewId, rating
14	ReviewUnflagged	14	Whenever a review is unflagged	userId, ownerId, type, reviewId, rating

Starting Event Listener

1. ***Importing Event Consumer dependency through Maven***

```
<dependency>
 <groupId>com.snapdeal.reviews</groupId>
 <artifactId>event-consumer-core</artifactId>
 <version>0.0.2</version>
</dependency>
```

2. ***Writing the Handler for handling Review Events. A sample Event Handler is as below.***

```
public class SampleEventHandler implements EventHandler {
 @Override
 public void handle(BaseReviewEvent event) {
  // Handle various Review events.
  switch (event.getEventType()) {
   case REVIEWCREATED:
    // TODO when a new review is created
    break;
   case REVIEWAPPROVED:
    // TODO when a review is approved
    break;
   default:
    break;
   }
 }
}
```

3. ***Starting the Event Listener consumers. You can load and instantiate all the classes you need to as part of your listener system. A sample class is as below.***

```
public class ThirdPartyEventListenerProcess {
 public static void main(final String[] args) {
  try {
   // Create a new Event Listener(This will internally start the kafka
consumer)
   // Associate an Event Handler with the Event Listener. The Handler
will have the logic for handling the events.
   // This will start the Kafka consumers.
   new EventListenerGroup(new SampleEventHandler());
  } catch (EventException e) {
   // Handler for event exception
  }
  // Replace this with an Object.wait()
  while (true) {
  }
 }
}
```

4. ***Adding and Modifying Kafka consumer properties. All the Listener properties are defined in kafka_consumer.properties file.***

```
# Licensed to the Apache Software Foundation (ASF) under one or more
```

```
# contributor license agreements.  See the NOTICE file distributed
with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You 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.
# see kafka.consumer.ConsumerConfig for more details

# Zookeeper connection string
# comma separated host:port pairs, each corresponding to a zk
# server. e.g. "127.0.0.1:3000,127.0.0.1:3001,127.0.0.1:3002"
zookeeper.connect=10.1.28.18:2181

# timeout in ms for connecting to zookeeper
zookeeper.connection.timeout.ms=1000000

#consumer group id. Make sure the group id is unique for your
listener.
group.id=review_group_service

#consumer timeout
#consumer.timeout.ms=5000

#-----------------------------------------------------------------
---------
#Snapdeal specific properties

#Refer with the Reviews API team on how many partitions are for this
topic/queue.
sd.total.num.of.partitions=1

#For Distribution of Consumer threads based on number of listener
processes.
sd.total.num.of.consumer.processes=1

#The name of the topic this consumer is listening
sd.topic.name=review_events_ext

#Review events that this consumer must listen to. Refer
com.snapdeal.reviews.data.event.ReviewEventType (or the "Events that
the Listeners can listen" section in this page) for values.
#Comma separated int values. Leave it blank to listen to all the
events
```

```
sd.listening.events=0,1

#-------------------------------------------------------------------
---------
```

5. **Starting the Event Listener.**

```
//Start the ThirdPartyEventListenerProcess class with the VM argument:
-Dconfig.location=<path to the folder where
kafka_consumer.properties>. Alternatively the argumentcan be
set programatically: System.setProperty("config.location", "<path to
the folder where kafka_consumer.properties>").
```

Troubleshooting:

1. Make sure that the system hosts file has a mapping for the host name at where Kafka and zookeeper are running.
    a. The hosts file is located at Linux: /etc/hosts. Windows: C:\Windows\System32\drivers\etc\hosts
    b. Eg. entry: 10.1.28.18 poc-review4.snapdeal.com
2. Make sure that the "group.id" property in kafka_consumer.properties is unique. Kafka ensures that if two consumers have the same "group.id", only one of them will receive the event.