Item 298
git_comments:

1. * * Creates an empty array if missing * @param node object to create the array within * @param field location to create the array * @return the Map representing the extensions property
2. * * JSON utilities.
3. * 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 * * 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.
4. since 2.1 use mapper.getFactory() instead
5. * * Creates an empty array if missing * @param node objectnode to create the object within * @param field location to create the object * @return the Map representing the extensions property
6. Assume classpath
7. We are only going to use the Hosebird processor to manage the extraction of the tweets from the Stream
8. **comment:** Deserializing to an ObjectNode can take time. Parallelize the task to improve throughput
   label: code-design
9. * * TwitterStreamHelper helps with hosebird twitter stream.
10. * 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 * * 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.
11. * * TwitterStreamHelper constructor. * @param provider TwitterStreamProvider * @param poolSize poolSize
12. 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 * 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.
13. * * BasicTokenManager constructor. * @param tokens Collection of tokens
14. * * Manages a pool of tokens the most basic possible way. * If all tokens are added to the manager before {@link BasicTokenManager#getNextAvailableToken() getNextAvailableToken} * is called tokens are issued in the order they were added to the manager, FIFO. The BasicTokenManager acts as a circular queue * of tokens. Once the manager issues all available tokens it will cycle back to the first token and start issuing tokens again. * * </p> * When adding tokens to the pool of available tokens, the manager will not add tokens that are already in the pool. * * <p/> * The manager class is thread safe.
15. * * Persist montoring messages to SLF4J.
16. * * resolve a remote schema safely. * @param absolute root URI * @param relativePart relative to root * @return URI if resolvable, or Optional.absent()
17. <configLocation>https://raw.githubusercontent.com/databricks/sbt-databricks/master/scalastyle-config.xml</configLocation>
18. * * configure ActivityConverterUtil.
19. * * convert document to activity. * * @param document document to convert * @return result
20. **comment:** Use getInstance to get a globally shared thread-safe ActivityConverterUtil, rather than call this constructor. Reflection-based resolution of converters across all modules can be slow and should only happen once per JVM.
    label: code-design
21. **comment:** Use getInstance to get a globally shared thread-safe ActivityConverterUtil, rather than call this constructor. Reflection-based resolution of converters across all modules can be slow and should only happen once per JVM.
    label: code-design
22. * * configure ActivityObjectConverterUtil.
23. * * convert document to ActivityObject. * * @param document document to convert * @return result
24. * * HoconConverterProcessor. * * @param outClass outClass * @param hocon hocon * @param inPath inPath * @param outPath outPath
25. * * convert. * @param object object * @param outClass outClass * @param hocon hocon * @param inPath inPath * @param outPath outPath * @return result
26. github.com/typesafehub/config/blob/master/HOCON.md) scripts.
27. * * convertResultToString * @param entry * @return result
28. * * parseMap * @param field * @return result

29. * * processLine * @param line * @return result
30. * * parseTs * @param field * @return
31. * * getInstance. * @param configuration * @return result
32. * * convert * @param object * @param outClass * @param mapper * @return
33. * Override this to add parameters to the request.
34. * Override this to alter json payload on to the request.
35. * Override this to alter request URI.
36. * Override this to add headers to the request.
37. * Override this to add parameters to the request.
38. * Override this to place result in non-standard location on document.
39. * Override this to alter request URI.
40. * Override this to set a payload on the request.
41. * Override this to store a result other than exact json representation of response.
42. * * Override this to set the URI for the request or modify headers. * @param uri uri * @return result
43. * * SimpleHTTPGetProcessor constructor - uses provided HttpProcessorConfiguration.
44. * * SimpleHTTPGetProcessor constructor - resolves HttpProcessorConfiguration from JVM 'http'.
45. * Override this to add parameters to the request.
46. * * SimpleHTTPPostProcessor constructor - resolves HttpProcessorConfiguration from JVM 'http'.
47. * Override this to place result in non-standard location on document.
48. * Override this to store a result other than exact json representation of response.
49. * * Override this to set the URI / entity for the request or modify headers. * @param uri uri * @param entity entity * @return result
50. * * SimpleHTTPPostProcessor constructor - uses provided HttpProcessorConfiguration.
51. * Override this to add parameters to the request.
52. * * SimpleHttpProvider constructor - uses provided HttpProviderConfiguration.
53. * Override this to change how metadata is derived from object.
54. * * SimpleHttpProvider constructor - resolves HttpProcessorConfiguration from JVM 'http'.
55. * * prepareHttpRequest * @param uri uri * @return result
56. * Override this to change how entity gets converted to objects.
57. * * resolve a serializable configuration pojo from a portion of the JVM config object. * @param subConfig subConfig * @return result
58. * * resolve a serializable configuration pojo from a given typesafe config object. * @param typesafeConfig typesafeConfig * @return result
59. * * resolve a serializable configuration pojo from a portion of a given typesafe config object. * @param typesafeConfig typesafeConfig * @param subConfig subConfig * @return result
60. Pull all configuration files from the classpath, system properties, and environment variables
61. * * KinesisPersistReader reads documents from kinesis.
62. * * KinesisPersistReader constructor - uses provided KinesisReaderConfiguration.
63. * * KinesisPersistReader constructor - resolves KinesisReaderConfiguration from JVM 'kinesis'.
64. * * KinesisPersistReaderTask constructor.
65. * * KinesisPersistReaderTask reads documents from kinesis on behalf of * @see {@link KinesisPersistReader}.
66. * * KinesisPersistWriter constructor - resolves KinesisWriterConfiguration from JVM 'kinesis'.
67. * * KinesisPersistWriter constructor - uses provided KinesisWriterConfiguration.
68. * * ensureEverythingIsReleased as part of close process.
69. * * Create an input stream safely. * @param s3Object s3Object
70. * * Whenever the output stream is closed we are going to kick the ByteArrayOutputStream off to Amazon S3. * @throws IOException * Exception thrown from the FileOutputStream
71. * * Create an OutputStream Wrapper * @param amazonS3Client * The Amazon S3 Client which will be handling the file * @param bucketName * The Bucket Name you are wishing to write to. * @param path * The path where the object will live * @param fileName * The fileName you ware wishing to write. * @param metaData * Any meta data that is to be written along with the object * @throws IOException * If there is an issue creating the stream, this
72. * * If you can list files that are in this path, then you must be dealing with a directory * if you cannot list files that are in this path, then you are most likely dealing with * a simple file.
73. * * S3PersistReader reads documents from s3.
74. * * S3PersistReaderTask reads documents from s3 on behalf of * @see org.apache.streams.s3.S3PersistReader
75. * * Instantiator with a pre-existing amazonS3Client, this is used to help with re-use. * @param amazonS3Client * If you have an existing amazonS3Client, it wont' bother to create another one * @param s3WriterConfiguration * Configuration of the write paths and instructions are still required.
76. * * Reset File when it's time to create a new file. * @return OutputStreamWriter * @throws Exception Exception
77. generate a file name
78. * * S3PersistWriter writes documents to s3.
79. * * ConsolePersistReader reads documents from stdin.
80. * * ConsolePersistWriter writes documents to stdout.
81. * * ConsolePersistWriterTask writes documents to stdout on behalf of * @see org.apache.streams.console.ConsolePersistWriter

82. * * Wrapper class for a client with a known version.
83. * * Returns Client with clusterName. * @param clusterName clusterName
84. * * Wrapper class for multiple * @see org.apache.streams.elasticsearch.ElasticsearchClient
85. Check to see if we have a client.
86. * * Get the Client for this return, it is actually a transport client, but it is much * easier to work with the generic object as this interface likely won't change from * elasticsearch. This method is synchronized to block threads from creating * too many of these at any given time. * * @return Client for elasticsearch
87. * * Terminate the elasticsearch clients.
88. * * get Index to use based on supplied parameters. * * @param metadata metadata * @param config config * @return result
89. * * get id to use based on supplied parameters. * * @param datum datum * @return result
90. * * get Type to use based on supplied parameters. * * @param metadata metadata * @param config config * @return result
91. * * get id to use based on supplied parameters. * * @param metadata metadata * @return result
92. * * get routing id to use based on supplied parameters. * * @param datum datum * @return result
93. * * Utility class for handling Elasticsearch Metadata maps.
94. * * get JsonNode as Map. * @param node node * @return result
95. **comment:** TODO: move this to a utility package
    label: code-design
96. * * get parent id to use based on supplied parameters. * * @param datum datum * @return result
97. * * Enqueue DeleteRequest. * @param request request
98. * * Prepare and en-queue @see org.elasticsearch.action.delete.DeleteRequest * @param index index * @param type type * @param id id
99. * * ElasticsearchPersistDeleter deletes documents from elasticsearch.
100. * * ElasticsearchPersistReader reads documents from elasticsearch.
101. * * ElasticsearchPersistUpdater updates documents to elasticsearch.
102. * * Prepare and en-queue. * @see org.elasticsearch.action.update.UpdateRequest * @param indexName indexName * @param type type * @param id id * @param parent parent * @param routing routing * @param json json
103. * * Enqueue UpdateRequest. * @param request request
104. * * ElasticsearchPersistWriter constructor. * @param config config * @param manager manager
105. * * add based on supplied parameters. * @param indexName indexName * @param type type * @param id id * @param routing routing * @param ts ts * @param json json
106. * * add based on supplied parameters. * @param indexName indexName * @param type type * @param id id * @param ts ts * @param json json
107. * * ElasticsearchPersistUpdater updates documents to elasticsearch.
108. * Author: * Smashew * * Date: * 2013-10-20 * * Note: * With the information that we have on hand. We need to develop a heuristic * that will determine when the cluster is having a problem indexing records * by telling it to pause and wait for it to catch back up. A * * There is an impact to us, the caller, whenever this happens as well. Items * that are not yet fully indexed by the server sit in a queue, on the client * that can cause the heap to overflow. This has been seen when re-indexing * large amounts of data to a small cluster. The "deletes" + "indexes" can * cause the server to have many 'outstandingItems" in queue. Running this * software with large amounts of data, on a small cluster, can re-create * this problem. * * DO NOT DELETE THESE LINES
    ****
109. * * createIndexIfMissing * @param indexName indexName
110. * * shift to next page of scroll.
111. * * Helper for building, querying, and paging an elasticsearch query.
112. * * ElasticsearchQuery constructor - uses provided ElasticsearchReaderConfiguration.
113. * * ElasticsearchQuery constructor - resolves ElasticsearchReaderConfiguration from JVM 'elasticsearch'.
114. * * execute ElasticsearchQuery. * @param obj deprecated
115. * * PercolateQueryBuilder constructor. * @param id * @param query * @param defaultPercolateField
116. * * get active percolate tags. * @param index index * @return result
117. * * * @return returns true if all rules were addded. False indicates one or more rules have failed.
118. * * Attempt to removeOldTags. * @param ids ids * @param index index * @return Returns true if all of the old tags were removed. False indicates one or more tags were not removed.
119. * * createIndexIfMissing. * @param indexName indexName
120. * * delete old queries. * @param index index * @return result
121. * * Unit Test for * @see org.apache.streams.elasticsearch.processor.PercolateTagProcessor
122. * * Integration Test Suite for * @see org.apache.streams.elasticsearch
123. * * GraphHttpPersistWriter constructor - use supplied GraphHttpConfiguration. * @param configuration GraphHttpConfiguration
124. * * GraphHttpPersistWriter constructor - resolve GraphHttpConfiguration from JVM 'graph'.
125. * * Neo API query returns something like this: * { "columns": ["v"], "data": [[{ "data": { props }, etc... }], [{ "data": { props }, etc... }]] } * * @param jsonNode jsonNode * @return result
126. * * GraphVertexReader constructor - use supplied GraphReaderConfiguration. * @param configuration GraphReaderConfiguration

127. * * GraphVertexReader constructor - resolve GraphReaderConfiguration from JVM 'graph'.
128. * * createEdgeRequest. * @param activity activity * @return pair (query, parameterMap)
129. * * getPropertyCreater. * @param map paramMap * @return PropertyCreater string
130. * * mergeVertexRequest. * @param activityObject activityObject * @return pair (query, parameterMap)
131. * * getVertexRequest. * @param streamsId streamsId * @return pair (streamsId, parameterMap)
132. * * getVertexRequest. * @param vertexId numericId * @return pair (streamsId, parameterMap)
133. * * createVertexRequest. * @param activityObject activityObject * @return pair (query, parameterMap)
134. * * createHttpRequest neo4j rest json payload. * * @param queryPlusParameters (query, parameter map) * @return ObjectNode
135. * * HbasePersistWriter writes to hbase.
136. **comment:** TODO: refactor this to resolve this stuff from typesafe
     label: code-design
137. **comment:** TODO: refactor this to use HbaseConfiguration
     label: code-design
138. * * HbasePersistWriter constructor - resolve HbaseConfiguration from JVM 'hbase'.
139. * * HbasePersistWriter constructor - use supplied persistQueue. * @param persistQueue persistQueue
140. * * HbasePersistWriterTask writes to hbase on behalf of * @see org.apache.streams.hbase.HbasePersistWriter
141. *
     ****
     * This code is an example of how you would work with HDFS and you weren't going over * the webHDFS protocol.
     * * Smashew: 2013-10-01 *
     ****
     conf.set("fs.defaultFS", "hdfs://hadoop.mdigitallife.com:8020/user/" + userName);
     conf.set("namenode.host","0.0.0.0"); conf.set("hadoop.job.ugi", userName);
     conf.set(DFSConfigKeys.DFS_NAMENODE_USER_NAME_KEY, "runner"); fileSystem.createNewFile(new Path("/user/"+ userName + "/test")); FileStatus[] status = fs.listStatus(new Path("/user/" + userName)); for(int i=0;i<status.length;i++) { LOGGER.info("Directory: {}", status[i].getPath()); }
142. * * isConnected. * @return true if connected, false otherwise
143. * * getFileSystem. * @return FileSystem
144. * * getURI from hdfsConfiguration. * @return URI * @throws URISyntaxException URISyntaxException
145. **comment:** TODO: combine with WebHdfsPersistReader.connectToWebHDFS
     label: code-design
146. * * WebHdfsPersistReader constructor - uses supplied HdfsReaderConfiguration. * @param hdfsConfiguration hdfsConfiguration
147. * * WebHdfsPersistReader constructor - resolves HdfsReaderConfiguration from JVM 'hdfs'.
148. * * WebHdfsPersistReaderTask reads from hdfs on behalf of * @see org.apache.streams.hdfs.WebHdfsPersistReader
149. *
     ****
     * This code is an example of how you would work with HDFS and you weren't going over * the webHDFS protocol.
     * * Smashew: 2013-10-01 *
     ****
     conf.set("fs.defaultFS", "hdfs://hadoop.mdigitallife.com:8020/user/" + userName);
     conf.set("namenode.host","0.0.0.0"); conf.set("hadoop.job.ugi", userName);
     conf.set(DFSConfigKeys.DFS_NAMENODE_USER_NAME_KEY, "runner"); fileSystem.createNewFile(new Path("/user/"+ userName + "/test")); FileStatus[] status = fs.listStatus(new Path("/user/" + userName)); for(int i=0;i<status.length;i++) { LOGGER.info("Directory: {}", status[i].getPath()); }
150. * * isConnected. * @return true if connected, false otherwise
151. * * getFileSystem. * @return FileSystem
152. * * getURI from hdfsConfiguration. * @return URI * @throws URISyntaxException URISyntaxException
153. **comment:** TODO: combine with WebHdfsPersistReader.getFileSystem
     label: code-design
154. * * WebHdfsPersistWriter writes to hdfs.
155. **comment:** TODO: combine with WebHdfsPersistReader.getURI
     label: code-design
156. **comment:** TODO: combine with WebHdfsPersistReader.isConnected
     label: code-design
157. * * WebHdfsPersistReaderTask writes to hdfs on behalf of * @see org.apache.streams.hdfs.WebHdfsPersistWriter
158. * * KafkaPersistReader reads documents from kafka.
159. * * KafkaPersistReader constructor - resolves KafkaConfiguration from JVM 'kafka'.
160. * * KafkaPersistReader constructor - uses supplied persistQueue.
161. * * KafkaPersistReaderTask reads documents from kafka on behalf of * @see org.apache.streams.kafka.KafkaPersistReader
162. * * KafkaPersistWriter constructor - resolves KafkaConfiguration from JVM 'kafka'.
163. * * KafkaPersistWriter writes documents to kafka.
164. * * KafkaPersistWriter constructor - uses supplied persistQueue.

165. * * run persist writer thread
166. * * KafkaPersistWriterTask writes documents to kafka on behalf of * @see org.apache.streams.kafka.KafkaPersistWriter
167. * * KafkaPersistReader constructor - uses supplied MongoConfiguration. * @param config config
168. * * MongoPersistReader reads documents from mongo.
169. * * KafkaPersistReader constructor - resolves KafkaConfiguration from JVM 'mongo'.
170. * * KafkaPersistReader constructor - uses supplied persistQueue. * @param persistQueue persistQueue
171. * * start write thread.
172. * * stop.
173. * * Recursively removes all additionalProperties maps. * @param node ObjectNode
174. * * TypeConverterProcessor constructor. * @param inClass inClass * @param outClass outClass
175. * * TypeConverterProcessor constructor. * @param inClass inClass * @param outClass outClass * @param formats formats
176. * * TypeConverterProcessor constructor. * @param inClass inClass * @param outClass outClass * @param mapper mapper
177. * Override this to add parameters to the request.
178. * * AccountTypeProcessor constructor - uses supplied HttpProcessorConfiguration. * @param peoplePatternConfiguration peoplePatternConfiguration
179. * * AccountTypeProcessor constructor - resolves HttpProcessorConfiguration from JVM 'peoplepattern'.
180. * Override this to add parameters to the request.
181. * * AccountTypeProcessor constructor - uses supplied HttpProcessorConfiguration. * @param peoplePatternConfiguration peoplePatternConfiguration
182. * * DemographicsProcessor constructor - resolves HttpProcessorConfiguration from JVM 'peoplepattern'.
183. * * Configures the value to be persisted to the extensions object. * @param extracted the value extracted by the regex * @return an object representing the appropriate extension
184. * * Extracts matches of the given pattern that are bounded by separation characters and returns them as a list. * @param pattern the pattern for the substring to match. For example, [0-9]* matches 911 in Emergency number is 911. * @param content the complete content to find matches in. * @return a non-null list of matches.
185. * * Extracts matches of the given pattern in the content and returns them as a list. * @param pattern the pattern for the substring to match. For example, [0-9]* matches 911 in Emergency number is 911. * @param content the complete content to find matches in. * @return a non-null list of matches.
186. * * FacebookTypeConverter converts facebook data to activity streams types.
187. **comment:** TODO: use standard validation
     label: code-design
188. * * convert. * @param event event * @param inClass inClass * @param outClass outClass * @return Object * @throws ActivitySerializerException ActivitySerializerException * @throws JsonProcessingException JsonProcessingException
189. **comment:** TODO: replace with standard validation
     label: code-design
190. * * Queues facebook data. * @param data data * @param id id
191. * * Queries facebook and queues the resulting data. * @param id id * @throws Exception Exception
192. * * Gets a Facebook client. If multiple authenticated users for this app are available * it will rotate through the users oauth credentials * @return client
193. * * Returns true when the collector has finished querying facebook and has queued all data * for the provider. * @return isComplete
194. * * FacebookDataCollector constructor. * @param config config * @param queue queue
195. * * detectClass from json string. * @param json json string * @return detected Class
196. * * FacebookEventClassifier classifies facebook events.
197. * * FacebookFriendFeedProvider constructor - resolves FacebookUserInformationConfiguration from JVM 'facebook'.
198. * * FacebookFriendFeedProvider constructor - output supplied Class. * @param klass Class
199. * * FacebookFriendFeedProvider constructor - uses supplied FacebookUserInformationConfiguration.
200. * * FacebookFriendUpdatesProvider constructor. * uses supplied FacebookUserstreamConfiguration. * uses supplied output Class.
201. * * FacebookFriendUpdatesProvider constructor - uses supplied FacebookUserstreamConfiguration.
202. **comment:** TODO: factor this out.
     label: code-design
203. client.rawAPI().callPostAPI(); add a subscription
204. * * FacebookFriendUpdatesProvider constructor - resolves FacebookUserInformationConfiguration from JVM 'facebook'.
205. * * FacebookFriendUpdatesProvider constructor. * uses supplied output Class. * resolves FacebookUserInformationConfiguration from JVM 'facebook.
206. * * FacebookFriendUpdatesProvider provides updates from friend feed.
207. * * FacebookProvider constructor - resolves FacebookConfiguration from JVM 'facebook'.
208. * * FacebookProvider constructor - uses supplied FacebookConfiguration.

209. * * Overrides the ids and addedAfter time in the configuration. * @param idsToAfterDate idsToAfterDate
210. * * FacebookUserInformationProvider constructor - resolves FacebookUserInformationConfiguration from JVM 'facebook'.
211. * * FacebookUserInformationProvider constructor - uses supplie FacebookUserInformationConfiguration. * @param config
212. * * FacebookUserstreamProvider constructor.
213. * * FacebookUserstreamProvider constructor. * @param config config
214. * * FacebookUserstreamProvider constructor. * @param klass output Class
215. * * Queries facebook. Attempts requests up to 5 times and backs off on each facebook exception. * @param pageId pageId * @param paging paging * @return ResponseList of $link{facebook4j.Post} * @throws Exception Exception
216. * * Run FacebookPageFeedProvider from command line. * @param args configfile outfile * @throws Exception Exception
217. * * Builds an {@link org.apache.streams.pojo.json.ActivityObject} object from the {@link Post}. * @param post post * @return {@link org.apache.streams.pojo.json.ActivityObject}
218. * * Builds the activity {@link org.apache.streams.pojo.json.ActivityObject} actor from the Page. * @param page the object to use as the source * @return a valid Actor populated from the Page
219. * * Builds out the {@link org.apache.streams.pojo.json.ActivityObject} from the given {@link Post}. * @param post * @return {@link org.apache.streams.pojo.json.ActivityObject}
220. * * FacebookActivityUtil helps convert facebook data to activity formats.
221. * * Updates the given Activity object with the values from the Post. * @param post post * @param activity activity * @throws ActivitySerializerException
222. * * Builds the actor extensions given the page object. * @param actor actor * @param page page
223. * * Gets the common facebook {@link org.apache.streams.pojo.json.Provider} object. * @return a provider object representing Facebook
224. * * Updates the given Activity object with the values from the Page. * @param page the object to use as the source * @param activity the target of the updates. Will receive all values from the Page. * @throws ActivitySerializerException
225. * * Provider. * @param post post * @return Provider
226. * * createActor. * @param post post * @return ActivityObject
227. * * GMailMessageActivitySerializer converts a GMail message to Activity.
228. * * GooglePlusCommentProcessor collects comments about a google plus activity.
229. * * GooglePlusTypeConverter is a StreamsProcessor that converts gplus activities to activitystreams activities.
230. * * Set and overwrite the default before date that was read from the configuration file. * @param defaultBeforeDate defaultBeforeDate
231. * * Set and overwrite the default after date that was read from teh configuration file. * @param defaultAfterDate defaultAfterDate
232. * * Sets and overwrite the user info from the configuaration file. Uses the defaults before and after dates. * @param userIds userIds
233. * * Set and overwrite user into from the configuration file. Only sets after date. * @param usersAndAfterDates usersAndAfterDates
234. * * GPlusActivitySerializer converts gplus activities to as1 activities.
235. * * Looks at the status code of the exception. If the code indicates that the request should be retried, * it executes the back off strategy and returns true. * @param gjre GoogleJsonResponseException * @param backOff BackOffStrategy * @return returns true if the error code of the exception indicates the request should be retried.
236. * * Key for all public activities * https://developers.google.com/+/api/latest/activities/list
237. * * GPlusUserActivityCollector constructor. * @param plus Plus * @param datumQueue BlockingQueue<StreamsDatum> * @param backOff BackOffStrategy * @param userInfo UserInfo
238. * * Max results allowed per request * https://developers.google.com/+/api/latest/activities/list
239. * * Retrieve recent activity from a list of accounts. * @param args args * @throws Exception Exception
240. * * GPlusUserDataCollector constructor. * @param plus Plus * @param backOffStrategy BackOffStrategy * @param datumQueue BlockingQueue of StreamsDatum * @param userInfo UserInfo
241. * * Retrieve current profile status for a list of accounts. * @param args args * @throws Exception Exception
242. * * Given a raw JsonNode representation of an Activity's attachments, build out that * list of {@link com.google.api.services.plus.model.Activity.PlusObject.Attachments} objects * * @param objectNode objectNode * @return list of {@link com.google.api.services.plus.model.Activity.PlusObject.Attachments} objects
243. * * Because the GooglePlus Activity object {@link com.google.api.services.plus.model.Activity} contains complex objects * within its hierarchy, we have to use a custom deserializer * * @param jsonParser jsonParser * @param deserializationContext deserializationContext * @return The deserialized {@link com.google.api.services.plus.model.Activity} object * @throws IOException IOException * @throws JsonProcessingException JsonProcessingException
244. * * Given a raw JsonNode, build out the G+ {@link com.google.api.services.plus.model.Activity.Actor} object * * @param node node * @return {@link com.google.api.services.plus.model.Activity.Actor} object
245. * * Given a JsonNode, build out all aspects of the {@link com.google.api.services.plus.model.Activity.PlusObject} object * * @param node node * @return {@link com.google.api.services.plus.model.Activity.PlusObject} object
246. * * GPlusCommentDeserializer converts gplus comments to as1 comments.

247. * * Because the GooglePlus Comment object {@link com.google.api.services.plus.model.Comment} contains complex objects * within its hierarchy, we have to use a custom deserializer * * @param jsonParser jsonParser * @param deserializationContext deserializationContext * @return The deserialized {@link com.google.api.services.plus.model.Comment} object * @throws java.io.IOException IOException * @throws com.fasterxml.jackson.core.JsonProcessingException JsonProcessingException
248. * * Detect likely class of String json. * @param json String json * @return likely class
249. * * GPlusEventClassifier classifies GPlus Events.
250. * * Because the GooglePlus Person object contains complex objects within its hierarchy, we have to use * a custom deserializer * * @param jsonParser jsonParser * @param deserializationContext deserializationContext * @return The deserialized {@link com.google.api.services.plus.model.Person} object * @throws IOException IOException * @throws JsonProcessingException JsonProcessingException
251. * * Given a {@link com.google.api.services.plus.model.Person} object and an * {@link org.apache.streams.pojo.json.Activity} object, fill out the appropriate details. * * @param item Person * @param activity Activity * @throws ActivitySerializerException ActivitySerializerException
252. * * Given a {@link List} of {@link com.google.api.services.plus.model.Comment} objects and an * {@link org.apache.streams.pojo.json.Activity}, update that Activity to contain all comments * * @param comments input List of Comment * @param activity output Activity
253. * * Given a {@link com.google.api.services.plus.model.Activity.Actor} object, return a fully fleshed * out {@link org.apache.streams.pojo.json.ActivityObject} actor * * @param gPlusActor input c.g.a.s.p.m.Activity.Actor * @return {@link ActivityObject} output $.actor as o.a.s.p.j.ActivityObject
254. * * Given a Google Plus {@link com.google.api.services.plus.model.Activity}, * convert that into an Activity streams formatted {@link org.apache.streams.pojo.json.Activity} * * @param gPlusActivity input c.g.a.s.p.m.Activity * @param activity output o.a.s.p.j.Activity
255. * * GooglePlusActivityUtil helps convert c.g.Person and c.g.Activity into o.a.s.p.j.o.Page and o.a.s.p.j.Activity.
256. * * Gets the common googleplus {@link org.apache.streams.pojo.json.Provider} object * @return a provider object representing GooglePlus
257. * * Add in necessary extensions from the passed in {@link com.google.api.services.plus.model.Activity} to the * {@link org.apache.streams.pojo.json.Activity} object * * @param activity output o.a.s.p.j.Activity * @param gPlusActivity input c.g.a.s.p.m.Activity
258. * * Formats the ID to conform with the Apache Streams activity ID convention * @param idparts the parts of the ID to join * @return a valid Activity ID in format "id:googleplus:part1:part2:...partN"
259. * * Set the {@link org.apache.streams.pojo.json.ActivityObject} field given the passed in * {@link com.google.api.services.plus.model.Activity.PlusObject} * * @param activity output $.object as o.a.s.p.j.ActivityObject * @param plusObject input c.g.a.s.p.m.Activity.PlusObject
260. * * Extract the relevant details from the passed in {@link com.google.api.services.plus.model.Person} object and build * an actor with them * * @param person Person * @return Actor constructed with relevant Person details
261. * * Adds a single {@link com.google.api.services.plus.model.Comment} to the Object.Attachments * section of the passed in {@link org.apache.streams.pojo.json.Activity} * * @param activity output o.a.s.p.j.Activity * @param comment input c.g.a.s.p.m.Comment
262. * * setup.
263. * * setup.
264. * * setup.
265. * * setup.
266. * * Test that every collector will be run and that data queued from the collectors will be processed.
267. * * Creates a randomized activity and randomized date range. * * <p/> * The activity feed is separated into three chunks, * |. . . data too recent to be in date range . . .||. . . data in date range. . .||. . . data too old to be in date range| * [index 0, ..., index length-1] * * <p/> * Inside of those chunks data has no order, but the list is ordered by those three chunks. * * <p/> * The test will check to see if the num of data in the date range make onto the output queue.
268. * * Test that on success a datum will be added to the queue. * @throws Exception Exception
269. * * Test that on failure, no datums are output. * @throws Exception Exception
270. * * GPlusEventClassifierTest tests GPlusEventClassifier.
271. * * Overrides the users in the configuration and sets the after date for each user. A NULL DateTime implies * pull data from as early as possible. If default before or after DateTimes are set, they will applied to all * NULL DateTimes. * @param usersWithAfterDate instagram user id mapped to BeforeDate time
272. * * Add default start and stop points if necessary.
273. * * Overrides authroized user tokens in the configuration. * @param tokenStrings tokenStrings
274. * * Return the data collector to use to connect to instagram. * @return {@link InstagramDataCollector}
275. * * Overrides the default after date in the configuration. * @param afterDate afterDate
276. * * Overrides the default before date in the configuration. * @param beforeDate beforeDate
277. * * Overrides the client id in the configuration. * @param clientId client id to use
278. * * InstagramDataCollector constructor. * @param queue Queue of StreamsDatum * @param config InstagramConfiguration
279. * * Queues the Instagram data to be output by the provider. * @param userData data to queue * @param userId user id who the data came from

280. * * Takes an Instagram Object and sets it as the document of a streams datum and sets the id of the streams datum. * @param item * @return
281. * * If there are authorized tokens available, it sets a new token for the client and returns * the client. If there are no available tokens, it simply returns the client that was * initialized in the constructor with client id. * @return result
282. * * Return the number of available tokens for this data collector. * @return numbeer of available tokens
283. * * Pull instagram data for a user and queues the resulting data. * @param user * @throws Exception
284. * * @return true when the collector has queued all of the available Instagram data for the provided users.
285. * * Pull Recement Media for a user and queues the resulting data. Will try a single call 5 times before failing and * moving on to the next call or returning. * @param user user * @throws Exception Exception
286. * * To use from command line: * * <p/> * Supply (at least) the following required configuration in application.conf: * * <p/> * instagram.clientKey * instagram.usersInfo.authorizedTokens * instagram.usersInfo.users * * <p/> * Launch using: * * <p/> * mvn exec:java \ * -Dexec.mainClass=org.apache.streams.instagram.provider.recentmedia.InstagramRecentMediaProvider \ * -Dexec.args="application.conf media.json" * * @param args args * @throws Exception Exception
287. * * To use from command line: * * <p/> * Supply (at least) the following required configuration in application.conf: * * <p/> * instagram.clientKey * instagram.usersInfo.authorizedTokens * instagram.usersInfo.users * * <p/> * Launch using: * * <p/> * mvn exec:java \ * -Dexec.mainClass=org.apache.streams.instagram.provider.userinfo.InstagramUserInfoProvider \ * -Dexec.args="application.conf userinfo.json" * * @param args args * @throws Exception Exception
288. * * Builds the object. * @param item the item * @return a valid Activity Object
289. * * Builds an Actor object given a UserInfoData object. * @param item UserInfoData item * @return Actor object
290. * * Adds the location extension and populates with the instagram data. * @param activity the Activity object to update * @param item the object to use as the source
291. * * Gets the common instagram {@link org.apache.streams.pojo.json.Provider} object. * @return a provider object representing Instagram
292. * * Updates the given Activity object with the values from the item * @param item the object to use as the source * @param activity the target of the updates. Will receive all values from the tweet. * @throws ActivitySerializerException ActivitySerializerException
293. * * Adds any video objects to the attachment field. * @param attachments attachments * @param item item
294. * * Formats the ID to conform with the Apache Streams activity ID convention * @param idparts the parts of the ID to join * @return a valid Activity ID in format "id:instagram:part1:part2:...partN"
295. * * Builds the actor. * @param item MediaFeedData item * @return a valid ActivityObject
296. * * Updates the given Activity object with the values from the item * @param item the object to use as the source * @param activity the target of the updates. Will receive all values from the tweet. * @throws ActivityConversionException ActivityConversionException
297. * * Adds any image objects to the attachment field. * @param attachments attachments * @param item item
298. * * Gets the links from the Instagram event. * @param item the object to use as the source * @return a list of links corresponding to the expanded URL
299. * * Takes various parameters from the instagram object that are currently not part of the * activity schema and stores them in a generic extensions attribute. * @param activity Activity activity * @param item MediaFeedData item
300. * * Builds all of the attachments associated with a MediaFeedData object. * * @param item item * @return result
301. * * get limit ArticlesAfter sequenceId. * @param sequenceId sequenceId * @param limit limit * @return MoreoverResult * @throws IOException IOException
302. * * MoreoverClient constructor. * @param id id * @param apiKey apiKey * @param sequence sequence
303. * * MoreoverProvider constructor. * @param moreoverConfiguration MoreoverConfiguration
304. * * To use from command line: * * <p/> * Supply configuration similar to src/test/resources/rss.conf * * <p/> * Launch using: * * <p/> * mvn exec:java -Dexec.mainClass=org.apache.streams.moreover.MoreoverProvider -Dexec.args="rss.conf articles.json" * * @param args args * @throws Exception Exception
305. * * MoreoverProviderTask constructor. * @param apiId apiId * @param apiKey apiKey * @param results results * @param lastSequence lastSequence
306. * * Process batch and * @return max sequenceId.
307. * * convert Author and platformName to Actor. * @param author Author * @param platformName platformName * @return $.actor
308. * * convertObject. * @param article article * @return ActivityObject $.object
309. * * convert article into Activity. * @param article article * @return Activity
310. * * addLanguageExtension. * @param activity Activity * @param article Article
311. * * convertTarget. * @param source source * @return ActivityObject $.target
312. * * addLocationExtension. * @param activity Activity * @param source Source
313. * * convert Source to Provider. * @param source Source * @return Provider
314. * * MoreoverTestUtil.
315. * * * @param activity
316. * * Before. * @throws Exception Exception
317. * * RssEventProcessor processes Rss Events.
318. * * RssEventProcessor constructor. * @param inQueue inQueue * @param outQueue outQueue * @param inClass inClass * @param outClass outClass

319. * * RssEventProcessor constructor. * @param inQueue inQueue * @param outQueue outQueue * @param outClass outClass
320. * * To use from command line: * * <p/> * Supply configuration similar to src/test/resources/rss.conf * * <p/> * Launch using: * * <p/> * mvn exec:java -Dexec.mainClass=org.apache.streams.rss.provider.RssStreamProvider -Dexec.args="rss.conf articles.json" * @param args args * @throws Exception Exception
321. * * Map that contains the Set of previously seen articles by an rss feed.
322. * * Non-perpetual mode, time out of 10 sec * @see {@link org.apache.streams.rss.provider.RssStreamProviderTask * #RssStreamProviderTask(java.util.concurrent.BlockingQueue, String, org.joda.time.DateTime, int, boolean)} * @param queue queue * @param rssFeed rssFeed * @param publishedSince publishedSince
323. * * Reads the url and queues the data * @param feedUrl rss feed url * @return set of all article urls that were read from the feed * @throws IOException when it cannot connect to the url or the url is malformed * @throws FeedException when it cannot reed the feed.
324. * * Returns false if the artile was previously seen in another task for this feed. * @param id id * @param rssFeed rssFeed * @return boolean seenBefore
325. * * Non-perpetual mode, no date filter. * @see {@link org.apache.streams.rss.provider.RssStreamProviderTask * #RssStreamProviderTask(java.util.concurrent.BlockingQueue, String, org.joda.time.DateTime, int, boolean)} * @param queue queue * @param rssFeed rssFeed * @param timeOut timeOut
326. * * Non-perpetual mode, no date filter, time out of 10 sec * @see {@link org.apache.streams.rss.provider.RssStreamProviderTask * #RssStreamProviderTask(java.util.concurrent.BlockingQueue, String, org.joda.time.DateTime, int, boolean)} * @param queue queue * @param rssFeed rssFeed
327. * * The rss feed url that this task is responsible for reading. * @return rss feed url
328. * * RssStreamProviderTask that reads an rss feed url and queues the resulting articles as StreamsDatums with the documents * being object nodes. * @param queue Queue to push data to * @param rssFeed url of rss feed to read * @param publishedSince DateTime to filter articles by, will queue articles with published times after this * @param timeOut url connection timeout in milliseconds * @param perpetual true, if you want to run in perpetual mode. NOT RECOMMENDED
329. * * Returns link to the article to use as the id. * @param node node * @return String
330. * * Schedule Feeds.
331. * * RssFeedScheduler constructor. * @param service service * @param feedDetailsList feedDetailsList * @param dataQueue dataQueue * @param peroid peroid
332. * * deserializeWithRomeExtension ObjectNode entry withExtension. * @param entry ObjectNode * @param withExtension whether to add Rome Extension * @return Activity
333. * * Given an RSS entry, extra out the author and actor information and return it * in an actor object * * @param entry entry * @return $.actor
334. * Order of precedence for resourceLocation selection * 1. Valid URI * 2. Valid Link * 3. Non-null URI * 4. Non-null Link if (isValidResource(uri)) { resourceLocation = uri; } else if (isValidResource(link)) { resourceLocation = link; } else if (uri != null || link != null) { resourceLocation = (uri != null) ? uri : link; provider.setId("id:providers:rss"); provider.setUrl(resourceLocation); provider.setDisplayName("RSS"); return provider; } /** * Tests whether or not the passed in resource is a valid URI. * @param resource resource * @return boolean of whether or not the resource is valid
335. * * Given an RSS object, build and return the Provider object. * * @param entry ObjectNode * @return $.provider
336. * * Given an RSS object and an existing activity, * add the Rome extension to that activity and return it. * * @param activity Activity * @param entry ObjectNode * @return Activity
337. * * Given an RSS object, build the ActivityObject. * * @param entry ObjectNode * @return $.object
338. * * Test that a task can read a valid rss from a url and queue the data. * @throws Exception Exception
339. * * Test that you can task will only output aritcles after a certain published time. * @throws Exception Exception
340. * * Test that task will only output articles after a certain published time that it has not seen before. * @throws Exception Exception
341. * * Test that perpetual streams will not output previously seen articles. * @throws Exception Exception
342. * * clearPreviouslySeen.
343. * * Test that feeds are scheduled based on elapsed time correctly. * Takes 1 minute to run.
344. * * validate Provider Url. * @param provider Provider
345. * * convert BeatApi.BeatResponse.Beat to Activity * @param beat BeatApi.BeatResponse.Beat * @return Activity
346. * * HeartbeatInfo constructor. * @param xmlString xmlString * @throws Exception Exception
347. * * hasTagName. * @param tagName tagName * @return hasTagName
348. * * getTagWithTagName. * @param tagName tagName * @return SysomosTagDefinition
349. * * getQueries. * @return Queries
350. * * SysomosTagDefinition constructor. * @param tagName tagName * @param displayName displayName
351. * * Executes the request to the Sysomos Heartbeat API and returns a valid response.
352. * * The max number of items you are allowed to get per request.
353. * * Gets the Request URL based on the local fields. * @return a valid URL for the Sysomos API or an exception
354. * * Constructs a new ContentRequestBuilder for the specified API key and Sysomos URL. * @param baseUrl the base URL for the sysomos API * @param apiKey the API key generated by Sysomos for authorization

355. * * Sets the starting offset for the number of documents given the other parameters. * @param offset the starting offset * * @return The RequestBuilder for continued Chaining
356. * * Sets the size of the expected response. * @param size the number of documents * * @return The RequestBuilder for continued Chaining
357. * * Sets the date after which documents should be returned from Sysomos. * @param afterDate the {@link org.joda.time.DateTime} instance representing the after date * * @return The RequestBuilder for continued Chaining
358. * * Sets the date before which documents should be returned from Sysomos. * @param beforeDate the {@link org.joda.time.DateTime} instance representing the before date * * @return The RequestBuilder for continued Chaining
359. * * Returns the full url need to execute a request. * * <p/> * Example: * http://api.sysomos.com/dev/v1/heartbeat/content?apiKey=YOUR * -APIKEY&hid=YOUR-HEARTBEAT-ID&offset=0&size=10& * addedAfter=2010-10-15T13:00:00Z&addedBefore=2010-10-18T13:00:00Z * * @return the URL to use when making requests of Sysomos Heartbeat
360. * * Executes the request to the Sysomos Heartbeat API and returns a valid response
361. * * Sets the Sysomos Heartbeat ID. * @param hid Heartbeat ID * * @return The RequestBuilder for continued Chaining
362. * * Sets the Sysomos Heartbeat ID as a String. * @param hid Heartbeat ID string * * @return The RequestBuilder for continued Chaining
363. * * SysomosHeartbeatStream constructor. * @param provider SysomosProvider * @param heartbeatId heartbeatId * @param mode OperatingMode
364. * * SysomosHeartbeatStream constructor. * @param provider SysomosProvider * @param heartbeatId heartbeatId * @param beforeTime DateTime * @param afterTime DateTime
365. * * SysomosHeartbeatStream constructor. * @param provider SysomosProvider * @param heartbeatId heartbeatId * @param documentId last documentId
366. Ensure that we are only assigning lastId to the latest ID, even if there is backfill query. Since offset is calcuated at the end of the run, if we detect the need to backfill, it will increment to 1
367. * * Wait for the queue size to be below threshold before allowing execution to continue on this thread.
368. * * SysomosProvider constructor. * @param sysomosConfiguration SysomosConfiguration
369. * * To use from command line: * * <p/> * Supply configuration similar to src/test/resources/rss.conf * * <p/> * Launch using: * * <p/> * mvn exec:java -Dexec.mainClass=org.apache.streams.rss.provider.RssStreamProvider -Dexec.args="rss.conf articles.json" * * @param args args * @throws Exception Exception
370. * * signalComplete. * @param heartbeatId heartbeatId
371. * * Queries the sysomos URL and provides the response as a String. * * @param url the Sysomos URL to query * @return valid XML String
372. * * before.
373. * * Convert to millis with TWITTER_FORMATTER. * @param dateTime dateTime as String * @return millis as Long
374. * * StreamsTwitterMapper constructor.
375. * * TwitterDateTimeFormat.
376. **comment:** TODO: use class explicitly somewhere
     label: code-design
377. * * build ActivityObject from UserstreamEvent * @param event UserstreamEvent * @return $.actor
378. * * build ActivityObject from UserstreamEvent * @param event UserstreamEvent * @return $.object
379. * * convert UserstreamEvent to Activity. * @param event UserstreamEvent * @return Activity * @throws ActivityConversionException ActivityConversionException
380. **comment:** TODO: Use this class explicitely somewhere
     label: code-design
381. * * Adds the location extension and populates with teh twitter data. * @param activity the Activity object to update * @param tweet the object to use as the source
382. * * Takes various parameters from the twitter object that are currently not part of the * activity schema and stores them in a generic extensions attribute. * @param activity Activity * @param tweet Tweet
383. * * Adds the given Twitter event to the activity as an extension. * @param activity the Activity object to update * @param event the Twitter event to add as the extension
384. * * Gets the common twitter {@link org.apache.streams.pojo.json.Provider} object * @return a provider object representing Twitter
385. * * Builds the ActivityObject for the delete event. * @param delete the delete event * @return a valid Activity Object
386. * * Gets the links from the Twitter event * @param tweet the object to use as the source * @return a list of links corresponding to the expanded URL (no t.co)
387. * * Updates the given Activity object with the values from the User * @param user the object to use as the source * @param activity the target of the updates. Will receive all values from the tweet.
388. * * Builds the activity {@link org.apache.streams.pojo.json.ActivityObject} actor from the tweet * @param tweet the object to use as the source * @return a valid Actor populated from the Tweet
389. * * Formats the ID to conform with the Apache Streams activity ID convention. * @param idparts the parts of the ID to join * @return a valid Activity ID in format "id:twitter:part1:part2:...partN"
390. * * Compute central coordinates from bounding box. * @param place the bounding box to use as the source

391. * * Updates the activity for a delete event. * @param delete the delete event * @param activity the Activity object to update * @throws ActivityConversionException ActivityConversionException
392. * * Updates the given Activity object with the values from the Tweet. * @param tweet the object to use as the source * @param activity the target of the updates. Will receive all values from the tweet. * @throws ActivityConversionException ActivityConversionException
393. * * Creates an {@link org.apache.streams.pojo.json.ActivityObject} for the tweet * @param tweet the object to use as the source * @return a valid ActivityObject
394. * * Builds the actor for a delete event. * @param delete the delete event * @return a valid Actor
395. * * Updates the content, and associated fields, with those from the given tweet * @param activity the target of the updates. Will receive all values from the tweet. * @param tweet the object to use as the source * @param verb the verb for the given activity's type
396. * * Builds the activity {@link org.apache.streams.pojo.json.ActivityObject} actor from the User * @param user the object to use as the source * @return a valid Actor populated from the Tweet
397. * * Builds the {@link org.apache.streams.twitter.pojo.TargetObject} from the tweet. * @param tweet the object to use as the source * @return currently returns null for all activities
398. * * TwitterUrlApiProcessor constructor.
399. * * handleTwitterError. * @param twitter Twitter * @param id id * @param exception exception * @return
400. * * To use from command line: * * <p/> * Supply (at least) the following required configuration in application.conf: * * <p/> * twitter.oauth.consumerKey * twitter.oauth.consumerSecret * twitter.oauth.accessToken * twitter.oauth.accessTokenSecret * twitter.info * * <p/> * Launch using: * * <p/> * mvn exec:java -Dexec.mainClass=org.apache.streams.twitter.provider.TwitterFollowingProvider -Dexec.args="application.conf tweets.json" * * @param args args * @throws Exception Exception
401. * * TwitterFollowingProviderTask constructor. * @param provider TwitterFollowingProvider * @param twitter Twitter * @param id numeric id
402. * * TwitterFollowingProviderTask constructor. * @param provider TwitterFollowingProvider * @param twitter Twitter * @param screenName screenName
403. * * baseUrl from TwitterConfiguration. * @param config TwitterConfiguration * @return baseUrl
404. * * To use from command line: * * <p/> * Supply (at least) the following required configuration in application.conf: * * <p/> * twitter.oauth.consumerKey * twitter.oauth.consumerSecret * twitter.oauth.accessToken * twitter.oauth.accessTokenSecret * * <p/> * Launch using: * * <p/> * mvn exec:java -Dexec.mainClass=org.apache.streams.twitter.provider.TwitterStreamProvider -Dexec.args="application.conf tweets.json" * * @param args
405. * * get Twitter Client from TwitterUserInformationConfiguration. * @return result
406. * * To use from command line: * * <p/> * Supply (at least) the following required configuration in application.conf: * * <p/> * twitter.oauth.consumerKey * twitter.oauth.consumerSecret * twitter.oauth.accessToken * twitter.oauth.accessTokenSecret * twitter.info * * <p/> * Launch using: * * <p/> * mvn exec:java -Dexec.mainClass=org.apache.streams.twitter.provider.TwitterTimelineProvider -Dexec.args="application.conf tweets.json" * * @param args args * @throws Exception Exception
407. * * Using the "info" list that is contained in the configuration, ensure that all * account identifiers are converted to IDs (Longs) instead of screenNames (Strings).
408. * * TwitterTimelineProviderTask constructor. * @param provider TwitterTimelineProvider * @param twitter Twitter * @param id Long
409. **comment:** TODO: abstract out, also appears in TwitterTimelineProvider
     label: code-design
410. **comment:** TODO: this should be abstracted out
     label: code-design
411. * * To use from command line: * * <p/> * Supply (at least) the following required configuration in application.conf: * * <p/> * twitter.oauth.consumerKey * twitter.oauth.consumerSecret * twitter.oauth.accessToken * twitter.oauth.accessTokenSecret * twitter.info * * <p/> * Launch using: * * <p/> * mvn exec:java -Dexec.mainClass=org.apache.streams.twitter.provider.TwitterUserInformationProvider -Dexec.args="application.conf tweets.json" * * @param args args * @throws Exception Exception
412. * * Retrieve current profile status from a list of user ids or names.
413. * * TwitterUserInformationProvider constructor. * Resolves config from JVM properties 'twitter'.
414. * * YoutubeChannelDataCollector constructor. * @param youTube YouTube * @param queue BlockingQueue of StreamsDatum * @param strategy BackOffStrategy * @param userInfo UserInfo * @param youtubeConfig YoutubeConfiguration
415. * * To use from command line: * * <p/> * Supply (at least) the following required configuration in application.conf: * * <p/> * youtube.oauth.pathToP12KeyFile * youtube.oauth.serviceAccountEmailAddress * youtube.apiKey * youtube.youtubeUsers * * <p/> * Launch using: * * <p/> * mvn exec:java -Dexec.mainClass=org.apache.streams.youtube.provider.YoutubeUserActivityProvider -Dexec.args="application.conf tweets.json" * * @param args args * @throws Exception Exception
416. * * Looks at the status code of the expception. If the code indicates that the request should be retried, * it executes the back off strategy and returns true. * @param gjre * @param backOff * @return returns true if the error code of the exception indicates the request should be retried.
417. * * Base Collector for Youtube Data.

418. * * Define a global instance of the HTTP transport.
419. * * Set and overwrite user into from teh configuration file. Only sets after dater. * @param usersAndAfterDates usersAndAfterDates
420. * * Set and overwrite the default before date that was read from the configuration file. * @param defaultBeforeDate defaultBeforeDate
421. * * Set and overwrite the default after date that was read from teh configuration file. * @param defaultAfterDate defaultAfterDate
422. * * Sets and overwrite the user info from the configuaration file. Uses the defaults before and after dates. * @param userIds Set of String userIds
423. * * YoutubeProvider constructor - uses supplied YoutubeConfiguration. * @param config YoutubeConfiguration
424. * * YoutubeProvider constructor. * Resolves config from JVM 'youtube'.
425. * * Define a global instance of the JSON factory.
426. * * Given a feed and an after and before date, fetch all relevant user videos * and place them into the datumQueue for post-processing. * @param feed ActivityListResponse * @param afterDate DateTime * @param beforeDate DateTime * @throws IOException IOException * @throws InterruptedException InterruptedException
427. * * Given a Youtube videoId, return the relevant Youtube Video object. * @param videoId videoId * @return List of Video * @throws IOException
428. * * Process a list of Video objects. * @param videos List of Video * @param afterDate afterDate * @param beforeDate beforeDate * @param activity com.google.api.services.youtube.model.Activity * @param feed ActivityListResponse
429. * * Max results allowed per request * https://developers.google.com/+/api/latest/activities/list
430. * * Iterate through all users in the Youtube configuration and collect all videos * associated with their accounts.
431. * * YoutubeUserActivityCollector constructor. * @param youtube YouTube * @param datumQueue BlockingQueue of StreamsDatum * @param backOff BackOffStrategy * @param userInfo UserInfo * @param config YoutubeConfiguration
432. * * YoutubeDataCollector for YoutubeUserActivityProvider.
433. * * To use from command line: * * <p/> * Supply (at least) the following required configuration in application.conf: * * <p/> * youtube.oauth.pathToP12KeyFile * youtube.oauth.serviceAccountEmailAddress * youtube.apiKey * youtube.youtubeUsers * * <p/> * Launch using: * * <p/> * mvn exec:java -Dexec.mainClass=org.apache.streams.youtube.provider.YoutubeUserActivityProvider -Dexec.args="application.conf tweets.json" * * @param args args * @throws Exception Exception
434. **comment:** TODO: use generic provider id concatenator
label: code-design
435. * * createActorForChannel. * @param channel Channel * @return $.actor
436. * * Given a {@link com.google.api.services.youtube.model.Channel} object and an * {@link org.apache.streams.pojo.json.Activity} object, fill out the appropriate details * * @param channel Channel * @param activity Activity * @throws ActivitySerializerException ActivitySerializerException
437. * * Add the Youtube extensions to the Activity object that we're building. * @param activity Activity * @param video Video
438. * * Formats the ID to conform with the Apache Streams activity ID convention * @param idparts the parts of the ID to join * @return a valid Activity ID in format "id:youtube:part1:part2:...partN"
439. * * Build an {@link org.apache.streams.pojo.json.ActivityObject} actor given the video object * @param video Video * @param id id * @return Actor object
440. * * Gets the common youtube {@link org.apache.streams.pojo.json.Provider} object * @return a provider object representing YouTube
441. * * Given a video object, create the appropriate activity object with a valid image * (thumbnail) and video URL. * @param video Video * @return Activity Object with Video URL and a thumbnail image
442. * * Given a {@link com.google.api.services.youtube.YouTube.Videos} object and an * {@link org.apache.streams.pojo.json.Activity} object, fill out the appropriate details * * @param video Video * @param activity Activity * @throws ActivitySerializerException ActivitySerializerException
443. * * detect probable Class of a json String from YouTube. * @param json json * @return Class
444. * * Given the raw JsonNode, construct a video snippet object. * @param node JsonNode * @return VideoSnippet
445. * * Given the raw JsonNode, construct a statistics object. * @param node JsonNode * @return VideoStatistics
446. * * Because the Youtube Video object contains complex objects within its hierarchy, we have to use * a custom deserializer * * @param jsonParser jsonParser * @param deserializationContext deserializationContext * @return The deserialized {@link com.google.api.services.youtube.YouTube.Videos} object * @throws java.io.IOException IOException * @throws com.fasterxml.jackson.core.JsonProcessingException JsonProcessingException
447. * * setup for test.
448. * * Test for YoutubeTypeConverter.
449. * * Test for YoutubeProvider.
450. * * Test that every collector will be run and that data queued from the collectors will be processed.
451. * * Test for YoutubeUserActivityCollector.
452. * * setup for test.
453. * * Test for YoutubeVideoSerDe.
454. * * Status of StreamsDatum.

455. * * increment specific DatumStatus by count. * @param workStatus DatumStatus * @param counter counter
456. * * increment specific DatumStatus by 1. * @param workStatus DatumStatus
457. * * accumulate partial DatumStatusCounter. * @param datumStatusCounter DatumStatusCounter
458. * * Builds the stream, and starts it or submits it based on implementation.
459. * * Add a {@link org.apache.streams.core.StreamsProcessor} to the data processing stream. * @param processorId unique id for this processor - must be unique across the entire stream * @param processor the processor to execute * @param numTasks the number of instances of this processor to run concurrently * @param connectToIds the ids of the {@link org.apache.streams.core.StreamsOperation} that this process will * receive data from. * @return this
460. * * Add a {@link org.apache.streams.core.StreamsProvider} to the data processing stream. The provider will execute * {@link org.apache.streams.core.StreamsProvider:readCurrent()} to produce data. * @param streamId unique if for this provider - must be unique across the entire stream. * @param provider provider to execute * @return this
461. * * Add a {@link org.apache.streams.core.StreamsProvider} to the data processing stream. The provider will execute * {@link org.apache.streams.core.StreamsProvider:readNext(BigInteger)} to produce data. * @param streamId unique if for this provider - must be unique across the entire stream. * @param provider provider to execute * @param sequence sequence to pass to {@link org.apache.streams.core.StreamsProvider:readNext(BigInteger)} method * @return this
462. * * Add a {@link org.apache.streams.core.StreamsPersistWriter} to the data processing stream. * @param persistWriterId unique id for this processor - must be unique across the entire stream * @param writer the writer to execute * @param numTasks the number of instances of this writer to run concurrently * @param connectToIds the ids of the {@link org.apache.streams.core.StreamsOperation} that this process will * receive data from. * @return this
463. * * Add a {@link org.apache.streams.core.StreamsProvider} to the data processing stream. The provider will execute * {@link org.apache.streams.core.StreamsProvider:readRange(DateTime, DateTime)} to produce data. Whether the start * and end dates are inclusive or exclusive is up to the implementation. * @param streamId unique if for this provider - must be unique across the entire stream. * @param provider provider to execute * @param start start date * @param end end date * @return this
464. **comment:** * * Stops the streams processing. No guarantee on a smooth shutdown. Optional method, may not be implemented in * all cases.
     label: code-design
465. * * This method will be called after initialization/serialization. Initialize any non-serializable objects here. * @param configurationObject Any object to help intialize the operation. ie. Map, JobContext, Properties, etc. The type * will be based on where the operation is being run (ie. hadoop, storm, locally, etc.)
466. * * Each operation must publish an identifier.
467. * * No guarantee that this method will ever be called. But upon shutdown of the stream, an attempt to call this method * will be made. * Use this method to terminate connections, etc.
468. * * Persist the StreamsDatum to the corresponding data store. * @param entry to be stored.
469. * * StreamsOperation for writing data out of a pipeline.
470. * * Process/Analyze the {@link org.apache.streams.core.StreamsDatum} and return the the StreamsDatums that will * passed to every down stream operation that reads from this processor. * @param entry StreamsDatum to be processed * @return resulting StreamDatums from processing. Should never be null or contain null object. Empty list OK.
471. * * Flag to indicate whether the provider is still producing data. * @return true if the processor is actively awaiting or producing data. False otherwise.
472. * * Read data with sequenceId greater than sequence. * @param sequence BigInteger sequence * @return {@link StreamsResultSet}
473. * * Start the operation of the stream.
474. * * Read data with event time between start DateTime and end DateTime. * @param start start DateTime * @param end end DateTime * @return {@link StreamsResultSet}
475. * * Read the current items available from the provider. * @return a non-null {@link org.apache.streams.core.StreamsResultSet}
476. * * StreamsResultSet is a wrapper for an Iterator around a set of StreamsDatum.
477. * * Adds an error occurred during a StreamsOperation step to the StreamsDatum's metadata. By convention, errors are * placed in the metadata under the "errors" and are organized by class simple name where the failure occurred. * * @param datum the datum on which the operation step errored * @param throwable the throwable encountered * @param operationClass the class of the operation
478. * * StreamsTaskCounterDeserializer: a JsonDeserializer for StreamsTaskCounterBroadcast.
479. * * Given a list of messages, persist them out through whatever appropriate * broadcast mechanism (HTTP request, SLF4J log, etc.). * @param messages List of String messages * @return statusCode represents whether or not the persist was successful
480. * * Given a List of String messages, convert them to a JSON array. * @param messages List of String messages * @return Serialized version of this JSON array
481. * * Given a list of messages as Strings, broadcast them to the broadcastUri * (if one is defined) * @param messages * @return int status code from POST response
482. * * Given a List of String messages, convert them to a JSON array. * @param messages List of String messages * @return Serialized version of this JSON array
483. * * setup.

484. * * Given a list of messages as Strings, broadcast them to the broadcastUri * (if one is defined) * @param messages * @return int status code from POST response
485. * * Persist montoring messages to SLF4J.
486. * * Get all relevant JMX beans, convert their values to strings, and then persist them.
487. * * prepare for execution.
488. * * DEPRECATED * Please initialize logging with monitoring object via typesafe. * @param streamConfig streamConfig map.
489. * * BroadcastMonitorThread constructor - uses supplied MonitoringConfiguration. * @param configuration MonitoringConfiguration
490. * * Initialize our object mapper with all of our bean's custom deserializers. * This way we can convert them to and from Strings dictated by our * POJOs which are generated from JSON schemas.
491. * * setup.
492. * * setup.
493. * * Base Test. * @param thread BroadcastMonitorThread
494. * * get all sources. * @return Iterator of URL
495. can this be moved to streams-schemas if schemastore available in scope? maybe an interface? lot of boilerplate / reuse between plugins however treatment is way different when resolving a type symbol vs resolving and listing fields .
496. * * Run from CLI without Maven * * <p/> * java -jar streams-plugin-cassandra-jar-with-dependencies.jar StreamsCassandraResourceGenerator src/main/jsonschema target/generated-resources * * @param args [sourceDirectory, targetDirectory]
497. * * run generate using supplied StreamsCassandraGenerationConfig. * @param config StreamsCassandraGenerationConfig
498. * * generateResource String from schema and resourceId. * @param schema Schema * @param resourceId String * @return CREATE TYPE ...
499. * * Run within a module containing a src/main/jsonschema directory. * * <p/> * mvn org.apache.streams.plugins:streams-plugin-cassandra:0.4-incubating:cassandra *
500. * * execute StreamsCassandraResourceGenerator mojo. * @throws MojoExecutionException MojoExecutionException * @throws MojoFailureException MojoFailureException
501. * * Test that cassandra resources are generated. * * @throws Exception Exception
502. * * get all sources. * @return Iterator of URL
503. * * run generate using supplied StreamsElasticsearchGenerationConfig. * @param config StreamsElasticsearchGenerationConfig
504. * * Run from CLI without Maven * * <p/> * java -jar streams-plugin-elasticsearch-jar-with-dependencies.jar StreamsElasticsearchResourceGenerator src/main/jsonschema target/generated-resources * * @param args [sourceDirectory, targetDirectory]
505. * * generateResource String from schema and resourceId. * @param schema Schema * @param resourceId String * @return mapping
506. can this be moved to streams-schemas if schemastore available in scope? maybe an interface? lot of boilerplate / reuse between plugins however treatment is way different when resolving a type symbol vs resolving and listing fields .
507. * * Test that Elasticsearch resources are generated. * * @throws Exception Exception
508. * * get all sources. * @return Iterator of URL
509. * * run generate using supplied StreamsHbaseGenerationConfig. * @param config StreamsHbaseGenerationConfig
510. * * Run from CLI without Maven * * <p/> * java -jar streams-plugin-hbase-jar-with-dependencies.jar StreamsHbaseResourceGenerator src/main/jsonschema target/generated-resources * * @param args [sourceDirectory, targetDirectory] *
511. * * generateResource String from schema and resourceId. * @param schema Schema * @param resourceId String * @return mapping
512. * * execute StreamsHbaseResourceGenerator mojo. * @throws MojoExecutionException MojoExecutionException * @throws MojoFailureException MojoFailureException
513. * * Tests that all example activities can be loaded into Activity beans. * * @throws Exception Exception
514. * * generateResource String from schema and resourceId. * @param schema Schema * @param resourceId String * @return CREATE TABLE ...
515. * * Run from CLI without Maven * * <p/> * java -jar streams-plugin-hive-jar-with-dependencies.jar StreamsHiveResourceGenerator src/main/jsonschema target/generated-resources * * @param args [sourceDirectory, targetDirectory] *
516. * * run generate using supplied StreamsHiveGenerationConfig. * @param config StreamsHiveGenerationConfig
517. * * execute StreamsHiveResourceGeneratorMojo. * @throws MojoExecutionException MojoExecutionException * @throws MojoFailureException MojoFailureException
518. TODO: figure out how to compare without AL header interfering String expectedDirectory = "target/test-classes/expected"; File testExpected = new File(expectedDirectory); Iterable<File> expectedIterator = Files.fileTreeTraverser().breadthFirstTraversal(testExpected) .filter(hqlFilter); Collection<File> expectedCollection = Lists.newArrayList(expectedIterator); int fails = 0; Iterator<File> iterator = expectedCollection.iterator(); while(iterator.hasNext()) { File objectExpected = iterator.next(); String expectedEnd = dropSourcePathPrefix(objectExpected.getAbsolutePath(), expectedDirectory); File objectActual = new File(config.getTargetDirectory() + "/" + expectedEnd); LOGGER.info("Comparing: {} and {}",

objectExpected.getAbsolutePath(), objectActual.getAbsolutePath()); assert(objectActual.exists()); if(FileUtils.contentEquals(objectActual, objectExpected) == true) { LOGGER.info("Exact Match!"); } else { LOGGER.info("No Match!"); fails++; } if(fails > 0) { LOGGER.info("Fails: {}", fails); Assert.fail(); }

519. * * Tests that all example activities can be loaded into Activity beans. * * @throws Exception Exception
520. * * get all sources. * @return Iterator of URL
521. * * run generate using supplied StreamsPigGenerationConfig. * @param config StreamsPigGenerationConfig
522. * * Run from CLI without Maven * * <p/> * java -jar streams-plugin-pig-jar-with-dependencies.jar StreamsPigResourceGenerator src/main/jsonschema target/generated-resources * * @param args [sourceDirectory, targetDirectory] *
523. * * generateResource String from schema and resourceId. * @param schema Schema * @param resourceId String * @return mapping
524. * * Embed within your own java code * * <p/> * StreamsPigGenerationConfig config = new StreamsPigGenerationConfig(); * config.setSourceDirectory("src/main/jsonschema"); * config.setTargetDirectory("target/generated-resources"); * StreamsPigResourceGenerator generator = new StreamsPigResourceGenerator(config); * generator.run(); *
525. * * execute StreamsPigResourceGeneratorMojo. * @throws MojoExecutionException MojoExecutionException * @throws MojoFailureException MojoFailureException
526. * * Tests that StreamsPigResourceGenerator via SDK generates pig resources. * * @throws Exception Exception
527. * * Run from CLI without Maven * * <p/> * java -jar streams-plugin-pojo-jar-with-dependencies.jar StreamsPojoSourceGenerator src/main/jsonschema target/generated-sources * * @param args [sourceDirectory, targetDirectory, targetPackage] *
528. * * execute StreamsPojoSourceGenerator. * @throws MojoExecutionException MojoExecutionException * @throws MojoFailureException MojoFailureException
529. * * Tests that StreamsPojoSourceGenerator via SDK generates pig resources. * * @throws Exception Exception
530. * * Run from CLI without Maven * * <p/> * java -jar streams-plugin-scala-jar-with-dependencies.jar StreamsScalaSourceGenerator target/generated-sources * * @param args [targetDirectory, targetPackage] *
531. * * StreamsScalaSourceGenerator constructor. * @param config StreamsScalaGenerationConfig
532. * * detect which Classes are Pojo Classes. * @param classes List of candidate Pojo Classes * @return List of actual Pojo Classes
533. * * detectSerializableClasses. * @return List of Serializable Classes
534. * * execute StreamsScalaSourceGeneratorMojo. * @throws MojoExecutionException MojoExecutionException * @throws MojoFailureException MojoFailureException
535. * * Tests that StreamsScalaSourceGenerator via SDK generates scala sources. * * @throws Exception Exception
536. * * Creates a standard extension property. * @param object object node to create the property in * @return {@link Map} representing the extensions property
537. * * Creates a standard extension property. * @param activity activity to create the property in * @return the Map representing the extensions property
538. * * Converts multiple Activities into a list of source documents. * @param list a typed List of documents * @return a list of source documents
539. * * Converts a POJO into one or more Activities. * @param serialized the string representation * @return a fully populated Activity object
540. * * Converts the activity to a POJO representation. * * @param deserialized the string * @return a fully populated Activity object
541. * * What class does this ActivityConverter require? * * @return The class the ActivityConverter requires. Should always return the templated class.
542. * * Gets the supported content type that can be deserialized/serialized * * @return A string representing the format name. Can be an IETF MIME type or other
543. * * Converts multiple documents into a list of Activity objects. * @param list a typed List of documents * @return a list of fully populated activities
544. * * Gets the supported content type that can be deserialized/serialized. * * @return A string representing the format name. Can be an IETF MIME type or other
545. * * Converts a POJO into an ActivityObject. * @param serialized the string representation * @return a fully populated Activity object
546. * * What class does this ActivityConverter require? * * @return The class the ActivityConverter requires. Should always return the templated class.
547. * * Converts the activity to a POJO representation. * * @param deserialized the string * @return a fully populated Activity object
548. * * Converts multiple documents into a list of Activity objects. * @param serializedList a typed List of documents * @return a list of fully populated activities
549. * * Gets the supported content type that can be deserialized/serialized. * * @return A string representing the format name. Can be an IETF MIME type or other
550. * * Converts the activity to a POJO representation. * * @param deserialized the string * @return a fully populated Activity object
551. * * Converts a POJO into an Activity. * @param serialized the string representation * @return a fully populated Activity object

552. * * Assess the structure of the document, and identify whether the provided document is * a structural match for one or more typed forms. * * @param document the document * @return a serializable pojo class this document matches
553. * * The language of the post.
554. * * Location that the post was made or the actor's residence.
555. * * Gets a formatted object ID. * @param provider name of the provider * @param objectType type of the object * @param objectId the ID of the object * @return id:{provider}:{objectType}s:{objectId}
556. * * Check validity of Activity. * @param activity Activity * @return isValid
557. * * Creates a standard extension property. * @param activity activity to create the property in * @return the Map representing the extensions property
558. * * Gets a formatted ID. * @param providerName name of the provider * @param personId ID of the person within the system * @return id:{providerName}:people:{personId}
559. * * Gets a formatted activity ID. * @param providerName name of the provider * @param activityId ID of the provider * @return id:{providerName}:activities:{activityId}
560. * * The number of retweets, shares, etc that the post has received.
561. * * Country that the post was made.
562. * * Gets a formatted provider ID. * @param providerName name of the provider * @return id:providers:{providerName}
563. * * Property on the activity object to use for extensions.
564. * * Check validity of ActivityObject. * @param activityObject ActivityObject * @return isValid
565. * * Specific JSON-geo coordinates (long,lat).
566. * * The number of +1, Like, favorites, etc that the post has received.
567. * * Contains various formats. All formats should be of international standards when comes to the ordering of the * days and month.
568. * * Parses arbitrarily formatted Strings representing dates or dates and times to a {@link org.joda.time.DateTime} * objects. It first attempts parse with international standards, assuming the dates are either dd MM yyyy or * yyyy MM dd. If that fails it will try American formats where the month precedes the days of the month. * @param dateString abitrarily formatted date or date and time string * @return {@link org.joda.time.DateTime} representation of the dateString
569. * * Contains alternative formats that will succeed after failures from the DEFAULT_FORMATTER. * i.e. 4/24/2014 will throw an exception on the default formatter because it will assume international date standards * However, the date will parse in the ALT_FORMATTER because it contains the US format of MM/dd/yyyy.
570. * * parse String to DateTime * @param toParse DateTime as UTC String * @return DateTime
571. * * Formats an arbitrarily formatted into RFC3339 Specifications. * @param dateString date string to be formatted * @return RFC3339 compliant date string
572. * * Applies each additional format in turn, until it can provide a non-null DateTime
573. * * get custom StreamsJacksonMapper. * @param configuration StreamsJacksonMapperConfiguration * @return StreamsJacksonMapper
574. * * get custom StreamsJacksonMapper. * @param formats formats * @return StreamsJacksonMapper
575. * * get custom StreamsJacksonMapper. * @param format format * @return StreamsJacksonMapper
576. * * get default StreamsJacksonMapper. * @return StreamsJacksonMapper
577. **comment:** Use getInstance to get a globally shared thread-safe ObjectMapper, rather than call this constructor. Reflection-based resolution of date-time formats across all modules can be slow and should only happen once per JVM.
     label: code-design
578. * * StdDeserializer of Period.
579. * * StdSerializer of Period.
580. * * Tests that all example activities can be loaded into Activity beans. * @throws Exception Exception
581. * * Tests that defined activity verbs have an example which can be loaded into * Activity beans and into verb-specific beans. * @throws Exception Exception
582. * * push multiple ObjectNode json datums into a stream. * @param headers HttpHeaders * @param body String json * @return Response
583. * * push a String json datum into a stream. * @param headers HttpHeaders * @param body String json * @return Response
584. * * push multiple String json datums into a stream. * @param headers HttpHeaders * @param body String json * @return Response
585. * * setup StreamBuilder. * @param streamsConfiguration StreamsConfiguration * @param resourceProviders Set of StreamsProvider * @return StreamBuilder
586. * * Run from console: * * <p/> * java -jar uber.jar server ./configuration.yml * * @param args ["server", configuration.yml] * @throws Exception Exception
587. * * Creates a local stream builder with all configuration resolved by typesafe
588. * * Shutsdown the running tasks in sudo depth first search kind of way. Checks that the upstream components have * finished running before shutting down. Waits till inbound queue is empty to shutdown. * @param comp StreamComponent to shut down. * @param streamTasks the list of non-StreamsProvider tasks for this stream. * @throws InterruptedException
589. **comment:** * * NOT IMPLEMENTED.

label: requirement

590. * * Creates a local stream builder with a config object and default maximum internal queue size of 500 * @param streamConfig * @deprecated use LocalRuntimeConfiguration constructor instead
591. * * Creates a local stream builder with a config object. If maxQueueCapacity is less than 1 the queue is * unbounded. * * @param maxQueueCapacity * @param streamConfig * * @deprecated use LocalRuntimeConfiguration constructor instead
592. * * Runs the data stream in the this JVM and blocks till completion.
593. * * Creates a local stream builder with no config object. If maxQueueCapacity is less than 1 the queue is * unbounded. * @param maxQueueCapacity * * @deprecated use LocalRuntimeConfiguration constructor instead
594. * * Add an outbound queue for this component. The queue should be an inbound queue of a downstream component. * @param component the component that this supplying their inbound queue * @param queue the queue to to put post processed/provided datums on
595. * * Add a component that supplies data through the inbound queue. * @param component that supplies data through the inbound queue
596. * * * @param id * @param processor * @param inQueue * @param numTasks
597. * * The inbound queue for this component * @return inbound queue
598. * * The number of tasks this to run this component * @return
599. * * * @param id * @param writer * @param inQueue * @param numTasks
600. * * The components that are immediately downstream of this component (aka child nodes) * @return Collection of child nodes of this component
601. * * * @param id * @param provider * @param sequence
602. * * Creates a {@link org.apache.streams.local.tasks.StreamsTask} that is running a clone of this component whose * inbound and outbound queues are appropriately connected to the parent and child nodes. * * @return StreamsTask for this component * @param timeout The timeout to use in milliseconds for any tasks that support configurable timeout
603. * * * @param id * @param provider * @param start * @param end
604. * * The components that are immediately upstream of this component (aka parent nodes) * @return Collection of parent nodes of this component
605. * * The unique of this component * @return
606. * * * @param id * @param provider
607. * * Get number of passed datums * @return number of passed datums
608. * * Get the failure rate. Calculated by num failed divided by (num passed + num failed) * @return the failure rate
609. * * Get number of failed datums * @return number of failed datums
610. * * * @param id
611. * * Increment emitted count * @param delta
612. * * Increment emitted count
613. * * Add the time it takes to process a single datum in milliseconds * @param processTime
614. * * Increment error count
615. * * Increment received count
616. * * Increment error count * @param delta
617. * * Increment received count * @param delta
618. * * Get the number of {@link org.apache.streams.core.StreamsDatum}s received by the streams process * @return number of received datums
619. * * Returns the max time in milliseconds it takes the task to readCurrent, process, or write to return. * @return
620. * * Returns the average time in milliseconds it takes the task to readCurrent, process, or write to return. * @return
621. * * Get the number of {@link org.apache.streams.core.StreamsDatum}s emitted by the streams process * @return number of emitted datums
622. * * Get the error rate of the streams process calculated by the number of errors not handled by the {@link org.apache.streams.local.tasks.StreamsTask} * divided by the number of datums received. * @return error rate
623. * * Get the number of errors that the process had to catch because the executing Provider/Processor/Writer did not * catch and handle the exception * @return number of handled errors
624. * * Creates a fixed size thread pool where corePoolSize & maximumPoolSize equal numThreads with an unbounded queue. * @param numThreads number of threads in pool * @param streamBuilder streambuilder to call {@link org.apache.streams.core.StreamBuilder#stop()} on upon receiving an unhandled throwable
625. * * Creates an unbounded, unregistered {@code ThroughputQueue}
626. * * Handle updating the stats whenever elements are removed from the queue * @param e Element removed
627. * * * @param maxSize * @param id
628. * * Creates a bounded, registered {@code ThroughputQueue} * * @param maxSize maximum capacity of queue, if maxSize < 1 then unbounded * @param id unique id for this queue to be registered with. if id == NULL then not registered
629. * * If elements have been removed from the queue or no elements have been added, it returns the average wait time * in milliseconds. If elements have been added, but none have been removed, it returns the time waited by the first * element in the queue. * * @return the average wait time in milliseconds
630. * * * @param maxSize * @param streamIdentifier * @param startedAt
631. * * Creates a bounded, unregistered {@code ThroughputQueue} * * @param maxSize maximum capacity of queue, if maxSize < 1 then unbounded

632. * * Handles updating the stats whenever elements are added to the queue
633. * * Element wrapper to measure time waiting on the queue * * @param <E>
634. * * * @param streamIdentifier * @param startedAt
635. * * * @param id * @param streamIdentifier * @param startedAt
636. * * Creates an unbounded, registered {@code ThroughputQueue} * * @param id unique id for this queue to be registered with. if id == NULL then not registered
637. * * Get the average time an item spends in queue in milliseconds * @return average time an item spends in queue in milliseconds
638. * * Returns the number of items on the queue. * @return number of items on queue
639. * * Get the maximum time an item has spent on the queue before being removed from the queue. * @return the maximum time an item has spent on the queue
640. * * Get the the throughput of the queue measured by the number of items removed from the queue * dived by the time the queue has been active. * Active time starts once the first item has been placed on the queue * @return throughput of queue. items/sec, items removed / time active
641. * * Get the number of items that have been removed from this queue * @return number of items that have been removed from the queue
642. * * Get the number of items that have been added to the queue * @return number of items that have been added to the queue
643. * * //TODO LOCAL MODE HACK. Need to fix * In order for our data streams to ported to other data flow frame works(Storm, Hadoop, Spark, etc) we need to be able to * enforce the serialization required by each framework. This needs some thought and design before a final solution is * made. * * In order to be able to copy/clone StreamDatums the orginal idea was to force all StreamsDatums to be java serializable. * This was seen as unacceptable for local mode. So until we come up with a solution to enforce serialization and be * compatiable across multiple frame works, this hack is in place. * * If datum.document is Serializable, we use serialization to clone a new copy. If it is not Serializable we attempt * different methods using an com.fasterxml.jackson.databind.ObjectMapper to copy/clone the StreamsDatum. If the object * is not clonable by these methods, an error is reported to the logging and a NULL object is returned. * * @param datum * @return
644. * * Adds a StreamDatum to the outgoing queues. If there are multiple queues, it uses serialization to create * clones of the datum and adds a new clone to each queue. * @param datum
645. * * SHOULD NOT BE NECCESARY, WILL REMOVE. * Round Robins through input queues to get the next StreamsDatum. If all input queues are empty, it will return null. * @return the next StreamsDatum or null if all input queues are empty.
646. * * * Note: * Quick class and method to let us see what is going on with the JVM. We need to make sure * that everything is running with as little memory as possible. If we are generating a heap * overflow, this will be very apparent by the information shown here.
647. * * * Note: * Quick class and method to let us see what is going on with the JVM. We need to make sure * that everything is running with as little memory as possible. If we are generating a heap * overflow, this will be very apparent by the information shown here.
648. * * Default constructor. Uses default sleep of 500ms when inbound queue is empty. * @param writer writer to execute in task
649. * * * @param writer writer to execute in task * @param streamConfig stream config
650. * * Default constructor, uses default sleep time of 500ms when inbound queue is empty * @param processor process to run in task
651. * * @param processor * @param streamConfig
652. * * Constructor for a StreamsProvider to execute {@link org.apache.streams.core.StreamsProvider:readNew(BigInteger)} * @param provider * @param sequence
653. * * Constructor for a StreamsProvider to execute {@link org.apache.streams.core.StreamsProvider:readCurrent()} * @param provider
654. * * Constructor for a StreamsProvider to execute {@link org.apache.streams.core.StreamsProvider:readRange(DateTime,DateTime)} * @param provider * @param start * @param end
655. * * Returns true when the task has not completed. Returns false otherwise * @return true when the task has not completed. Returns false otherwise
656. * * Add an input {@link java.util.Queue} for this task. * @param inputQueue
657. * * Returns the output queues that have been set for this task * @return list of output queues
658. * * Informs the task to stop. Tasks may or may not try to empty its inbound queue before halting.
659. * * Set the configuration object that will shared and passed to all instances of StreamsTask. * @param config optional configuration information
660. * * Returns true if the task is waiting on more data to process * @return true, if waiting on more data to process
661. * * Returns the input queues that have been set for this task. * @return list of input queues
662. * * Add an output {@link java.util.Queue} for this task. * @param outputQueue
663. * * Creates {@link org.apache.streams.core.StreamsProcessor} that passes any StreamsDatum it gets as an * input and counts the number of items it processes. * @param counter * @return
664. * * Tests that all datums pass through each processor and that all datums reach the writer * @param numDatums * @param numProcessors

665. * * Creates a StreamsPersistWriter that adds every datums document to a set * @param collector * @return
666. * * A simple example of how to run a stream in local mode. * @param args
667. * * Remove registered mbeans from previous tests * @throws Exception
668. * * Test that you can increment passes and it returns the correct count * @throws Exception
669. * * Test failure rate returns expected values
670. * * Test that you can increment failed and it returns the correct count * @throws Exception
671. * * Test Constructor can register the counter as an mxbean with throwing an exception.
672. * * Remove registered mbeans from previous tests * @throws Exception
673. * * Test emitted increments correctly and returns expected value * @throws Exception
674. * * Test received increments correctly and returns expected value * @throws Exception
675. * * Test constructor does not throw errors
676. * * Test errors increments correctly and returns expected value * @throws Exception
677. * * Test error rate returns expected value * @throws Exception
678. * * Test that queue will block on puts when the queue is full * @throws InterruptedException
679. * * Helper runnable for test {@link ThroughputQueueMultiThreadTest#testBlockOnFullQueue()}
680. * * Remove registered mbeans from previous tests * @throws Exception
681. * * Test that queue will block on Take when queue is empty * @throws InterruptedException
682. * * Test multiple threads putting and taking from the queue while * this thread repeatedly calls the MXBean measurement methods. * Should hammer the queue with request from multiple threads * of all request types. Purpose is to expose current modification exceptions * and/or dead locks.
683. * * Helper runnable class for test {@link ThroughputQueueMultiThreadTest#testBlockOnEmptyQueue()}
684. * * Test that take and put queue and dequeue data as expected and all * measurements form the queue are returning data. * @throws Exception
685. * * Test that the mbean registers
686. * * Test that max wait and avg wait return expected values * @throws Exception
687. * * Test that throughput returns expected values. * @throws Exception
688. * * Test that mulitple mbeans of the same type with a different name can be registered
689. * * Test that offer and poll queue and dequeue data as expected * and all measurements from the queue are returning data
690. * * Test that add and remove queue and dequeue data as expected * and all measurements from the queue are returning data
691. * * Set of instance ids that received data. Usefully for testing parrallelization is actually working.
692. * * The total count of data seen by a all instances of a processor.
693. * * Random instance to generate ids
694. * * Set of all ids that have been claimed. Ensures all instances are assigned unique ids
695. * * The total count of data seen by a all instances of a processor.
696. * * Random instance to generate ids
697. * * Set of instance ids that received data. Usefully for testing parrallelization is actually working.
698. **comment:** * * The documents received
     label: documentation
699. * * Set of all ids that have been claimed. Ensures all instances are assigned unique ids
700. **comment:** * * Test validity of documents vs schemas.
     label: documentation
701. * * Tests that activities matching core-ex* can be parsed by apache streams. * * @throws Exception Test Exception
702. * * Tests that activities expect to fail cannot be parsed by apache streams. * * @throws Exception test exception
703. * * Tests that activities matching core-ex* can be parsed by apache streams. * * @throws Exception test exception
704. * * Tests that activities matching simple* can be parsed by apache streams. * * @throws Exception test exception
705. * * Tests that activities matching core-ex* can be parsed by apache streams.
706. * * Tests that activities matching vocabulary-ex* can be parsed by apache streams. * * @throws Exception test exception
707. * * Attempts to register an object with local MBeanServer. Throws runtime exception on errors. * @param name name to register bean with * @param mbean mbean to register
708. * * Removes all mbeans registered undered a specific domain. Made specificly to clean up at unit tests * @param domain mbean domain
709. * * Certain types of queues will return null when calling {@link java.util.Queue#poll()} due to many factors depending * on the type of queue. <code>pollWhileNotEmpty</code> will poll the queue until an item from the queue is returned * or the queue is empty. If the queue is empty it will return NULL. * @param queue queue to read the entry from * @param <T> type * @return result
710. * * Attempts to safely {@link java.util.concurrent.ExecutorService#shutdown()} * and {@link java.util.concurrent.ExecutorService#awaitTermination(long, java.util.concurrent.TimeUnit)} * of an {@link java.util.concurrent.ExecutorService}. * @param stream service to be shutdown * @param initialWait time in seconds to wait for currently running threads to finish execution * @param secondaryWait time in seconds to wait for running threads that did not terminate to acknowledge their forced termination
711. * * Certain types of queues will fail to {@link java.util.Queue#offer(Object)} an item due to many factors * depending on the type of queue. <code>offerUntilSuccess</code> will not return until the item has been *

successfully queued onto the desired queue * @param entry item to queue * @param queue queue to add the entry to * @param <T> type
712. * * generateGuid from list of parts. * @param parts list of parts * @return guid
713. * * serialize Object as byte array. * * <p/> * BORROwED FROM APACHE STORM PROJECT * * @param obj Object * @return byte[]
714. * * deserialize byte array as Object. * * <p/> * BORROwED FROM APACHE STORM PROJECT * * @param serialized byte[] * @return Object
715. * * clone Object by serialization. * @param obj Object * @param <T> type * @return cloned Object
716. * * A BackOffStrategy that has a limited number of uses before it throws a * {@link org.apache.streams.util.api.requests.backoff.BackOffException}. * @param baseBackOffTime time to back off in milliseconds, must be greater than 0. * @param maximumNumberOfBackOffAttempts maximum number of attempts, must be grater than 0 or -1. * -1 indicates there is no maximum number of attempts.
717. * * A BackOffStrategy that can effectively be used endlessly. * @param baseBackOffTime amount of time back of in seconds
718. * * Calculate the amount of time in milliseconds that the strategy should back off for * @param attemptCount the number of attempts the strategy has backed off. * i.e. 1 -> this is the first attempt, 2 -> this is the second attempt, etc. * @param baseSleepTime the minimum amount of time it should back off for in milliseconds * @return the amount of time it should back off in milliseconds
719. * * Gets the longest sleep period that the strategy attempted. If the function that * initialized this exception does not set the longest sleep period, -1 will be returned. * @return longest sleep period that the strategy attempted
720. * * BackOffException constructor. * @param message message * @param attemptCount attemptCount * @param maxSleepTime maxSleepTime (in millis)
721. * * Gets the number of back off attempts that happened before the exception was thrown. If the function that * initialized this exception does not set the number of attempts, -1 will be returned. * @return number of back off attempts
722. * * Cause the current thread to sleep for an amount of time based on the implemented strategy. If limits are set * on the number of times the backOff can be called, an exception will be thrown. * @throws BackOffException BackOffException
723. * * Rests the back off strategy to its original state. * After the call the strategy will act as if {@link AbstractBackOffStrategy#backOff()} * has never been called.
724. * * A ConstantTimeBackOffStrategy that has a limited number of uses before it * throws a {@link org.apache.streams.util.api.requests.backoff.BackOffException} * @param baseBackOffTimeInMiliseconds time to back off in milliseconds, must be greater than 0. * @param maximumNumberOfBackOffAttempts maximum number of attempts, must be grater than 0 or -1. * -1 indicates there is no maximum number of attempts.
725. * * A ConstantTimeBackOffStrategy that can effectively be used endlessly. * @param baseBackOffTimeInMiliseconds amount of time back of in milliseconds
726. * * Limited use ExponentialBackOffStrategy. * @param baseBackOffTimeInSeconds baseBackOffTimeInSeconds * @param maxNumAttempts maxNumAttempts
727. * * Unlimited use ExponentialBackOffStrategy. * @param baseBackOffTimeInSeconds baseBackOffTimeInSeconds
728. * * Must create equals method for all OauthTokens. * @param object object for comparison * @return true if equal, and false otherwise
729. * * Get an available token. If no tokens are available it returns null. * @return next available token
730. * * Get the number of available tokens. * @return number of available tokens
731. * * Adds a token to the available token pool. * @param token Token to be added * @return true, if token was successfully added to the pool and false otherwise.
732. * * Adds a {@link java.util.Collection} of tokens to the available token pool. * @param tokens Tokens to be added * @return true, if the token pool size increased after adding the tokens, and false otherwise.
733. * * determine FieldType from ObjectNode. * @param fieldNode ObjectNode * @return FieldType
734. * * resolveRecursive. * @param config GenerationConfig * @param schemaFiles List of schemaFiles
735. * * dropExtension. * @param inputFile inputFile * @return extension dropped
736. * * swapExtension. * @param inputFile inputFile * @param originalExtension originalExtension * @param newExtension newExtension * @return extension swapped
737. * * drop source path prefix between inputFile and sourceDirectory. * @param inputFile inputFile * @param sourceDirectory sourceDirectory * @return without path prefix
738. * * writeFile. * @param resourceFile resourceFile * @param resourceContent resourceContent
739. * * Gets the file filter used to isolate the schema mapping files in the * source directories. * * @return the file filter use when scanning for schema files.
740. * * Gets the 'targetDirectory' configuration option. * * @return The target directory into which generated types will be written * (may or may not exist before types are written)
741. * * Gets the 'outputEncoding' configuration option. * * @return The character encoding that should be used when writing output files.
742. * * Gets the 'source' configuration option. * * @return The source file(s) or directory(ies) from which JSON Schema will * be read.
743. * * getParentUri. * @return Parent.Uri

744. * * Schema constructor. * @param uri uri * @param content JsonNode content * @param parent Schema parent * @param generate whether to generate
745. * * getParentContent. * @return Parent.Content
746. * * resolve full definition of 'items'. * @param schema Schema * @param fieldNode ObjectNode * @param resourceId resourceId * @return ObjectNode
747. * * read Schema from URL. * @param schemaUrl URL * @return ObjectNode
748. * * merge parent and child properties maps. * @param content ObjectNode * @param parent ObjectNode * @return merged ObjectNode
749. * * resolve a remote schema safely. * @param absolute root URI * @param relativePart relative to root * @return URI if resolvable, or Optional.absent()
750. * * get instance of Scanner using resource path. * @param resourcePath resourcePath * @return Scanner
751. * * Simple token for testing purposes.
752. * * Test class for thread safe check.
753. * * assert iterator of Schema contains URI items ending with in order. * @param iterator Iterator of Schema * @param items List of String
754. * * count wildcards in this ObjectCombination. * @param objectCombination ObjectCombination * @return count
755. * * is pattern a wildcard. * @param pattern String * @return true or false
756. * * whether this ActivityObject matches the corresponding ObjectCombination pattern. * @param activityObject ActivityObject * @param pattern pattern * @return true or false
757. * * count matches between this ObjectCombination and this Activity. * @param objectCombination ObjectCombination * @return count
758. * * whether this Activity matches this VerbDefinition. * @param activity Activity * @param verbDefinition VerbDefinition * @return true or false
759. * * whether this Activity matches any of a Set of VerbDefinitions. * @param activity Activity * @param verbDefinitionSet Set of VerbDefinition * @return true or false
760. * * whether this Activity matches this ObjectCombination. * @param activity Activity * @param criteria ObjectCombination * @return true or false
761. * * return all matching ObjectCombinations for an Activity. * @param activity Activity * @return List of ObjectCombination
762. * * return all matching VerbDefinitions for an Activity. * @param activity Activity * @return List of VerbDefinition
763. * * Readable display Name for ActivityObject. * @param activityObject ActivityObject * @return displayName
764. * * Transform Activity into readable string using ObjectCombination title and specified language. * @param language language * @param activity Activity * @param objectCombination ObjectCombination * @return String
765. * * Transform Activity into readable string using ObjectCombination title. * @param activity Activity * @param objectCombination ObjectCombination * @return String
766. * * Test of matchingVerbDefinitions.
767. * * Test of matchingObjectCombinations.
768. * * Test application of template with top-level fields.
769. * * Test application of template with second-level fields.
770. * * Test application of template with no field.
771. * * Test read verb definition from json.
772. * * Unit tests for VerbDefinition and utils.
773. * * Test verb definition defaults are set.

git_commits:

1. **summary:** STREAMS-440: custom checkstyle.xml, address compliance
   message: STREAMS-440: custom checkstyle.xml, address compliance Squashed commit of the following: commit bf329d31fd71a3e1fc21a76073876204ca806f88 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Wed Nov 23 09:57:31 2016 -0600 STREAMS-440: retrieve checkstyle xml from streams-master site commit 45e0edbcc7cfe755b520e04d2eab2fce3f28f0fb Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 19:57:17 2016 -0600 STREAMS-440: streams-verbs reduce wc -l target/checkstyle_result.xml from 400 to 36 commit fb911c9653108289f00b253751dce4693e77f2d9 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 19:24:43 2016 -0600 STREAMS-440: streams-util reduce wc -l target/checkstyle_result.xml from 1520 to 61 commit 887762372f97d60e2a4c1ba6ec4c55da07b81ffc Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 18:39:11 2016 -0600 STREAMS-440: streams-schema-activitystreams2 reduce wc -l target/checkstyle_result.xml from 78 to 6 commit f8dd9a935f063305994a75fb162af98008c92a32 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 18:36:44 2016 -0600 STREAMS-440: streams-schema-activitystreams reduce wc -l target/checkstyle_result.xml from 35 to 5 commit ff50402e9e049b973f8db2eb947a892ada59fa73 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 18:34:00 2016 -0600 STREAMS-440: streams-runtime-storm commit bdcea2d43d04e595e15905ae604916f3b984bbf9 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 18:33:18 2016 -0600 STREAMS-440: streams-runtime-pig reduce wc -l target/checkstyle_result.xml from 452 to 75 commit 010a2b8407594b156e2f94c472ebd8fe6b3e3f1f Author: Steve Blackmon @steveblackmon <sblackmon@apache.org>

Date: Tue Nov 22 18:30:32 2016 -0600 STREAMS-440: streams-runtime-local reduce wc -l target/checkstyle_result.xml from 3997 to 908 commit 262657144cc57c1893f93b4340a8f791cd1a56c4 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 18:23:07 2016 -0600 STREAMS-440: streams-runtime-dropwizard reduce wc -l target/checkstyle_result.xml from 311 to 19 commit 86890fc0f82106093010f1971175ed54179a0c58 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 18:13:35 2016 -0600 STREAMS-440: streams-pojo-extensions reduce wc -l target/checkstyle_result.xml from 230 to 16 commit 5ff2a253b31ed774d7f9e09e7449ab7bbc19d5f6 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 18:00:37 2016 -0600 STREAMS-440: streams-pojo reduce wc -l target/checkstyle_result.xml from 822 to 70 commit 10d0b8d25c469865a995448cf3433b382cf59ad3 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 17:23:36 2016 -0600 STREAMS-440: streams-plugin-scala reduce wc -l target/checkstyle_result.xml from 517 to 22 commit c505110874ab097acbee5638690caed4fb353668 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 17:08:35 2016 -0600 STREAMS-440: streams-plugin-pojo reduce wc -l target/checkstyle_result.xml to 19 commit f2757328f2d0db9b196ac9eb7baaecebcd9db918 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 16:58:06 2016 -0600 STREAMS-440: streams-plugin-pig reduce wc -l target/checkstyle_result.xml from 422 to 20 commit 54232cd7a272f72a72f5f1ea27ffb429d0d16e8c Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 15:30:08 2016 -0600 STREAMS-440: streams-plugin-hive reduce wc -l target/checkstyle_result.xml from 442 to 20 commit 89e6dbf402bd61a7628d4d760bab126b2a75cd30 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 15:14:53 2016 -0600 STREAMS-440: streams-plugin-hbase reduce wc -l target/checkstyle_result.xml from 346 to 18 commit 6226f79ca23fe6ee99a8ceb9866fbc2219241c5e Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 14:57:15 2016 -0600 STREAMS-440: streams-plugin-elasticsearch reduce wc -l target/checkstyle_result.xml from 522 to 23 commit 4dc32f145c3d88bcb879cc95e2d53a51badde02a Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 14:44:11 2016 -0600 STREAMS-440: streams-plugin-cassandra reduce wc -l target/checkstyle_result.xml from 512 to 20 commit bebccb51b7bc90767dd768cc05453f596b032ad8 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 14:21:18 2016 -0600 STREAMS-440: streams-monitoring reduce wc -l target/checkstyle_result.xml to 38 commit 9556e9840de1cfb2862fe2ba8e270cb23f268068 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 10:32:21 2016 -0600 STREAMS-440: streams-core reduce wc -l target/checkstyle_result.xml from 317 to 33 commit f92579033eefe1f12fdb71cb231b5df82ab79d0b Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 10:12:49 2016 -0600 STREAMS-440: streams-provider-youtube reduce wc -l target/checkstyle_result.xml from 1442 to 162 commit 29e32f7f24d0e4479e162de74bf76da55c9d6dc0 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 09:38:48 2016 -0600 STREAMS-440: streams-provider-twitter reduce wc -l target/checkstyle_result.xml from 2715 to 85 commit 66bba5ced09f982d8e8c37d0f27dc5a3702e7197 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 09:37:55 2016 -0600 STREAMS-440: streams-provider-sysomos reduce wc -l target/checkstyle_result.xml from 1087 to 36 commit 56f5ce34a96eb0743d81e6894984ee90d8d204e3 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Tue Nov 22 09:37:08 2016 -0600 STREAMS-440: streams-provider-rss reduce wc -l target/checkstyle_result.xml from 1261 to 34 commit ea4ab54ce118da5e46462b1e0867b18fb6440088 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Mon Nov 21 18:27:40 2016 -0600 STREAMS-440: streams-provider-moreover reduce wc -l target/checkstyle_result.xml from 695 to 32 commit 91573f2cccc8a44ef9efd1e3856055727b5eecb4 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sun Nov 20 17:03:13 2016 -0600 STREAMS-440: streams-provider-instagram reduce wc -l target/checkstyle-result.xml from 411 to 51 commit e05113a13994eb68b56ea8a87f0e09a7f6279c65 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sun Nov 20 14:05:03 2016 -0600 STREAMS-440: google-gplus reduce wc -l target/checkstyle-result.xml to 199 commit fa6704eb9887283bdeb3b0f36544db32ba920a08 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sun Nov 20 14:04:15 2016 -0600 STREAMS-440: google-gmail reduce wc -l target/checkstyle-result.xml to 61 commit 7b779df2b9d08de3d0ba6c92ec023d2fdd78dab0 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:56:05 2016 +0100 STREAMS-440: streams-provider-facebook reduce wc -l target/checkstyle-result.xml from 1842 to 780 commit 8d861005797b13093ce4b39eb94dcbdad0124c07 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:43:13 2016 +0100 STREAMS-440: streams-processor-regex reduce wc -l target/checkstyle-result.xml from 222 to 36 commit 29ee86a7db4444a47adb2dfba5e9bc6ffaa4d0c1 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:40:19 2016 +0100 STREAMS-440: streams-processor-peoplepattern reduce wc -l target/checkstyle-result.xml from 67 to 9 commit 5e96ff4e686a4e206d31976b626d1d35cd88e251 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:39:08 2016 +0100 STREAMS-440: streams-processor-json reduce wc -l target/checkstyle-result.xml from 242 to 31 commit 5adb1495ea54ccd69e24ffbe5cbc191ac60d75a3 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:38:30 2016 +0100 STREAMS-440: streams-processor-jackson reduce wc -l target/checkstyle-result.xml from 127 to 17 commit 43a9a7dcb660488b07a19090afedfffd5f529416 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:37:39 2016 +0100 STREAMS-440: streams-persist-mongo reduce wc -l target/checkstyle-result.xml to 37 commit 50e1390047edfa4daaa2089325cf408bf3d3872a

Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:37:07 2016 +0100 STREAMS-440: streams-persist-kafka reduces wc -l target/checkstyle-result.xml from 233 to 31 commit b4e59d2f68742679f3ccbb65dcda5de2feb03db7 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:36:26 2016 +0100 STREAMS-440: streams-persist-hdfs decrease wc -l target/checkstyle-result.xml from 508 to 58 commit 7e2b49f32ff11d0b6b5f06b1e22385a8c4fedf22 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:34:41 2016 +0100 STREAMS-440: streams-persist-hbase reduce wc -l target/checkstyle-result.xml from 204 to 20 commit 676be795634a5ef712362c35df915468d6732d8d Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:33:51 2016 +0100 STREAMS-440: streams-persist-graph decrease wc -l checkstyle-result.xml from 664 to 65 commit 9afe2db595736f8f4497a36870f2dd757e946835 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:31:00 2016 +0100 STREAMS-440: streams-persist-filebuffer commit 953f4cbb609b9c6399691989b84690cb79afc43a Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:30:28 2016 +0100 STREAMS-440: streams-persist-elasticsearch decrease wc -l checkstyle-result.xml from 1572 to 131 commit fa0d73e7569e02742f0be0bdcd4871d7c0f30931 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:28:55 2016 +0100 STREAMS-440: streams-persist-console decrease wc -l checkstyle-result.xml from 114 to 12 commit f8210dddbd3de79065e1ed494a69df21a2e0ff13 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:28:04 2016 +0100 STREAMS-440: streams-persist-s3 decrease wc -l checkstyle-result.xml from 580 to 61 commit db47e801b2901d7b1720cf6dc45646bcf5373dae Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:26:49 2016 +0100 STREAMS-440: streams-persist-kinesis decrease wc -l checkstyle-result.xml from 222 to 64 commit b3b75d4e7f84b7753309652d59fa46f134977c8b Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:25:12 2016 +0100 STREAMS-440: streams-config reduce wc -l target/checkstyle-result.xml from 85 to 60 commit cfa4f706f9d184aaf5e26b359b35068e2b57fc0f Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:23:16 2016 +0100 STREAMS-440: streams-http decrease wc -l checkstyle-result.xml from 742 to 65 commit fe649be82ab3d81cae61e0451858d6372d3a8780 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:22:25 2016 +0100 STREAMS-440: streams-filters decrease wc -l checkstyle-result.xml from 84 to 13 commit 2048f43dcff52621e16a1969efce92ee1bb7545f Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: Sat Nov 19 18:14:06 2016 +0100 STREAMS-440: streams-converters decrease wc -l checkstyle-result.xml from 928 to 122
label: code-design

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** Bring codebase into compliance with requirement for sensible javadoc headers
   description:

jira_issues_comments:

1. **body:** find . -name checkstyle-result.xml -print | while read filename; do echo $filename; cat "$filename" | grep javadoc; done > checkstyle.txt wc -l checkstyle.txt 995 checkstyle.txt
   label: code-design
2. https://raw.githubusercontent.com/google/styleguide/gh-pages/intellij-java-google-style.xml https://raw.githubusercontent.com/databricks/sbt-databricks/master/scalastyle-config.xml
3. GitHub user steveblackmon opened a pull request: https://github.com/apache/incubator-streams/pull/328 STREAMS-440: custom checkstyle.xml, address compliance one pass through entire project to increase compliance with checkstyle. certain modules (streams-runtime-*) held to lower standard because discussions on whether to delete them are on-going. You can merge this pull request into a Git repository by running: $ git pull https://github.com/steveblackmon/incubator-streams STREAMS-440 Alternatively you can review and apply these changes as the patch at: https://github.com/apache/incubator-streams/pull/328.patch To close this pull request, make a commit to your master/trunk branch with (at least) the following in the commit message: This closes #328 ---- commit 2048f43dcff52621e16a1969efce92ee1bb7545f Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:14:06Z STREAMS-440: streams-converters decrease wc -l checkstyle-result.xml from 928 to 122 commit fe649be82ab3d81cae61e0451858d6372d3a8780 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:22:25Z STREAMS-440: streams-filters decrease wc -l checkstyle-

result.xml from 84 to 13 commit cfa4f706f9d184aaf5e26b359b35068e2b57fc0f Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:23:16Z STREAMS-440: streams-http decrease wc -l checkstyle-result.xml from 742 to 65 commit b3b75d4e7f84b7753309652d59fa46f134977c8b Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:25:12Z STREAMS-440: streams-config reduce wc -l target/checkstyle-result.xml from 85 to 60 commit db47e801b2901d7b1720cf6dc45646bcf5373dae Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:26:49Z STREAMS-440: streams-persist-kinesis decrease wc -l checkstyle-result.xml from 222 to 64 commit f8210dddbd3de79065e1ed494a69df21a2e0ff13 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:28:04Z STREAMS-440: streams-persist-s3 decrease wc -l checkstyle-result.xml from 580 to 61 commit fa0d73e7569e02742f0be0bdcd4871d7c0f30931 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:28:55Z STREAMS-440: streams-persist-console decrease wc -l checkstyle-result.xml from 114 to 12 commit 953f4cbb609b9c6399691989b84690cb79afc43a Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:30:28Z STREAMS-440: streams-persist-elasticsearch decrease wc -l checkstyle-result.xml from 1572 to 131 commit 9afe2db595736f8f4497a36870f2dd757e946835 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:31:00Z STREAMS-440: streams-persist-filebuffer commit 676be795634a5ef712362c35df915468d6732d8d Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:33:51Z STREAMS-440: streams-persist-graph decrease wc -l checkstyle-result.xml from 664 to 65 commit 7e2b49f32ff11d0b6b5f06b1e22385a8c4fedf22 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:34:41Z STREAMS-440: streams-persist-hbase reduce wc -l target/checkstyle-result.xml from 204 to 20 commit b4e59d2f68742679f3ccbb65dcda5de2feb03db7 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:36:26Z STREAMS-440: streams-persist-hdfs decrease wc -l target/checkstyle-result.xml from 508 to 58 commit 50e1390047edfa4daaa2089325cf408bf3d3872a Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:37:07Z STREAMS-440: streams-persist-kafka reduces wc -l target/checkstyle-result.xml from 233 to 31 commit 43a9a7dcb660488b07a19090afedfffd5f529416 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:37:39Z STREAMS-440: streams-persist-mongo reduce wc -l target/checkstyle-result.xml to 37 commit 5adb1495ea54ccd69e24ffbe5cbc191ac60d75a3 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:38:30Z STREAMS-440: streams-processor-jackson reduce wc -l target/checkstyle-result.xml from 127 to 17 commit 5e96ff4e686a4e206d31976b626d1d35cd88e251 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:39:08Z STREAMS-440: streams-processor-json reduce wc -l target/checkstyle-result.xml from 242 to 31 commit 29ee86a7db4444a47adb2dfba5e9bc6ffaa4d0c1 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:40:19Z STREAMS-440: streams-processor-peoplepattern reduce wc -l target/checkstyle-result.xml from 67 to 9 commit 8d861005797b13093ce4b39eb94dcbdad0124c07 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:43:13Z STREAMS-440: streams-processor-regex reduce wc -l target/checkstyle-result.xml from 222 to 36 commit 7b779df2b9d08de3d0ba6c92ec023d2fdd78dab0 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-19T17:56:05Z STREAMS-440: streams-provider-facebook reduce wc -l target/checkstyle-result.xml from 1842 to 780 commit fa6704eb9887283bdeb3b0f36544db32ba920a08 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-20T20:04:15Z STREAMS-440: google-gmail reduce wc -l target/checkstyle-result.xml to 61 commit e05113a13994eb68b56ea8a87f0e09a7f6279c65 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-20T20:05:03Z STREAMS-440: google-gplus reduce wc -l target/checkstyle-result.xml to 199 commit 91573f2cccc8a44ef9efd1e3856055727b5eecb4 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-20T23:03:13Z STREAMS-440: streams-provider-instagram reduce wc -l target/checkstyle-result.xml from 411 to 51 commit ea4ab54ce118da5e46462b1e0867b18fb6440088 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-22T00:27:40Z STREAMS-440: streams-provider-moreover reduce wc -l target/checkstyle_result.xml from 695 to 32 commit 56f5ce34a96eb0743d81e6894984ee90d8d204e3 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-22T15:37:08Z STREAMS-440: streams-provider-rss reduce wc -l target/checkstyle_result.xml from 1261 to 34 commit 66bba5ced09f982d8e8c37d0f27dc5a3702e7197 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-22T15:37:55Z STREAMS-440: streams-provider-sysomos reduce wc -l target/checkstyle_result.xml from 1087 to 36 commit 29e32f7f24d0e4479e162de74bf76da55c9d6dc0 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-22T15:38:48Z STREAMS-440: streams-provider-twitter reduce wc -l target/checkstyle_result.xml from 2715 to 85 commit f92579033eefe1f12fdb71cb231b5df82ab79d0b Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-22T16:12:49Z STREAMS-440: streams-provider-youtube reduce wc -l target/checkstyle_result.xml from 1442 to 162 commit 9556e9840de1cfb2862fe2ba8e270cb23f268068 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-22T16:32:21Z STREAMS-440: streams-core reduce wc -l target/checkstyle_result.xml from 317 to 33 commit bebccb51b7bc90767dd768cc05453f596b032ad8 Author: Steve Blackmon @steveblackmon <sblackmon@apache.org> Date: 2016-11-22T20:21:18Z STREAMS-440: streams-monitoring reduce wc -l target/checkstyle_result.xml to 38 commit 4dc32f145c3d88bcb879cc95e2d53a51badde02a Author: Steve Blackmon

@steveblackmon <sblackmon@apache.org> Date: 2016-11-22T20:44:11Z STREAMS-440: streams-plugin-cassandra reduce wc -l target/checkstyle_result.xml from 512 to 20 ----

4. Github user smarthi commented on the issue: https://github.com/apache/incubator-streams/pull/328 lgtm +1 to merge
5. This merged in 5dffd5c32d0d150727a39104d428b21b52c911d4
6. Github user steveblackmon commented on the issue: https://github.com/apache/incubator-streams/pull/328 this merged with 5dffd5c32d0d150727a39104d428b21b52c911d4
7. Github user steveblackmon commented on the issue: https://github.com/apache/incubator-streams/pull/328 this merged w
8. Github user steveblackmon closed the pull request at: https://github.com/apache/incubator-streams/pull/328
9. Streams 0.4.1-incubating Release on Dec 26, 2016