Item 230
git_comments:

1. Create, initialize, and register the state store.
2. send a record to the processor
3. now, we trigger the punctuator, which iterates over the state store and forwards the contents.
4. * Licensed to the Apache Software Foundation (ASF) under one or more * contributor license agreements. See the NOTICE file distributed with * this work for additional information regarding copyright ownership. * The ASF licenses this file to You under the Apache License, Version 2.0 * (the "License"); you may not use this file except in compliance with * the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License.
5. finally, we can verify the output.
6. Changelog is not supported by MockProcessorContext.
7. note that the processor commits, but does not forward, during process()
8. Create and initialize the processor under test
9. * * Demonstrate the use of {@link MockProcessorContext} for testing the {@link Processor} in the {@link WordCountProcessorDemo}.
10. Immutable fields ==
11. * * Whether {@link ProcessorContext#commit()} has been called in this context. * * @return {@code true} iff {@link ProcessorContext#commit()} has been called in this context since construction or reset.
12. noinspection unchecked
13. * * Create a {@link MockProcessorContext} with dummy {@code taskId} and {@code null} {@code stateDir}. * Most unit tests using this mock won't need to know the taskId, * and most unit tests should be able to get by with the * {@link InMemoryKeyValueStore}, so the stateDir won't matter. * * @param config a Properties object, used to configure the context and the processor.
14. * * The context exposes these metadata for use in the processor. Normally, they are set by the Kafka Streams framework, * but for the purpose of driving unit tests, you can set them directly. * * @param topic A topic name * @param partition A partition number * @param offset A record offset * @param timestamp A record timestamp
15. * * The context exposes this metadata for use in the processor. Normally, they are set by the Kafka Streams framework, * but for the purpose of driving unit tests, you can set it directly. Setting this attribute doesn't affect the others. * * @param offset A record offset
16. **comment:** This interface is assumed by state stores that add change-logging. Rather than risk a mysterious ClassCastException during unit tests, throw an explanatory exception.
    label: code-design
17. mocks ==
18. * * Reset the commit capture to {@code false} (whether or not it was previously {@code true}).
19. * Licensed to the Apache Software Foundation (ASF) under one or more * contributor license agreements. See the NOTICE file distributed with * this work for additional information regarding copyright ownership. * The ASF licenses this file to You under the Apache License, Version 2.0 * (the "License"); you may not use this file except in compliance with * the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License.
20. * * Clear the captured forwarded data.
21. * * The context exposes this metadata for use in the processor. Normally, they are set by the Kafka Streams framework, * but for the purpose of driving unit tests, you can set it directly. Setting this attribute doesn't affect the others. * * @param partition A partition number
22. * * Create a {@link MockProcessorContext} with a specified taskId and null stateDir. * * @param config a {@link Properties} object, used to configure the context and the processor. * @param taskId a {@link TaskId}, which the context makes available via {@link MockProcessorContext#taskId()}. * @param stateDir a {@link File}, which the context makes available viw {@link MockProcessorContext#stateDir()}.
23. * * {@link MockProcessorContext} is a mock of {@link ProcessorContext} for users to test their {@link Processor}, * {@link Transformer}, and {@link ValueTransformer} implementations. * <p> * The tests for this class (org.apache.kafka.streams.MockProcessorContextTest) include several behavioral * tests that serve as example usage. * <p> * Note that this class does not take any automated actions (such as firing scheduled punctuators). * It simply captures any data it witnessess. * If you require more automated tests, we recommend wrapping your {@link Processor} in a minimal source-processor-sink * {@link Topology} and using the {@link TopologyTestDriver}.
24. * * The timestamp attached to the forwarded record. * * @return A timestamp, or {@code -1} if none was forwarded.
25. * * Get all the forwarded data this context has observed. The returned list will not be * affected by subsequent interactions with the context. The data in the list is in the same order as the calls to * {@code forward(...)}. * * @return A list of key/value pairs that were previously passed to the context.
26. * * Get the punctuators scheduled so far. The returned list is not affected by subsequent calls to {@code schedule(...)}. * * @return A list of captured punctuators.
27. * * The child this data was forwarded to. * * @return The child name, or {@code null} if it was broadcasted.
28. noinspection DoubleBraceInitialization
29. settable record metadata ==
30. * * The context exposes this metadata for use in the processor. Normally, they are set by the Kafka Streams framework, * but for the purpose of driving unit tests, you can set it directly. Setting this attribute doesn't affect the others. * * @param topic A topic name
31. * * Create a {@link MockProcessorContext} with dummy {@code config} and {@code taskId} and {@code null} {@code stateDir}. * Most unit tests using this mock won't need to know the taskId, * and most unit tests should be able to get by with the * {@link InMemoryKeyValueStore}, so the stateDir won't matter.
32. contructors ==
33. * * {@link CapturedPunctuator} holds captured punctuators, along with their scheduling information.
34. * * The context exposes this metadata for use in the processor. Normally, they are set by the Kafka Streams framework, * but for the purpose of driving unit tests, you can set it directly. Setting this attribute doesn't affect the others. * * @param timestamp A record timestamp
35. * * Get all the forwarded data this context has observed for a specific child by name. * The returned list will not be affected by subsequent interactions with the context. * The data in the list is in the same order as the calls to {@code forward(...)}. * * @param childName The child name to retrieve forwards for * @return A list of key/value pairs that were previously passed to the context.

36. * * The data forwarded. * * @return A key/value pair. Not null.
37. broadcast
38. expected, since the record metadata isn't initialized
39. * Licensed to the Apache Software Foundation (ASF) under one or more * contributor license agreements. See the NOTICE file distributed with * this work for additional information regarding copyright ownership. * The ASF licenses this file to You under the Apache License, Version 2.0 * (the "License"); you may not use this file except in compliance with * the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License.
40. noinspection unchecked
41. record metadata should be "sticky"
42. noinspection deprecation
43. expected
44. **comment:** div style="height:35px"> <a href="/{{version}}/documentation/streams/">Introduction</a> <a class="active-menu-item" href="/{{version}}/documentation/streams/developer-guide">Developer Guide</a> <a href="/{{version}}/documentation/streams/core-concepts">Concepts</a> <a href="/{{version}}/documentation/streams/quickstart">Run Demo App</a> <a href="/{{version}}/documentation/streams/tutorial">Tutorial: Write App</a> </div
label: documentation
45. timestamp
46. offset
47. you can reset forwards to clear the captured data. This may be helpful in constructing longer scenarios.
48. partition
49. commit captures can also be reset.

git_commits:

1. **summary:** KAFKA-6473: Add MockProcessorContext to public test-utils (#4736)
message: KAFKA-6473: Add MockProcessorContext to public test-utils (#4736) Reviewers: Matthias J. Sax <matthias@confluent.io>, Guozhang Wang <guozhang@confluent.io>, Bill Bejeck <bill@confluent.io>

github_issues:

github_issues_comments:

github_pulls:

1. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
2. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
3. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
4. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
5. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
6. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
7. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils).

The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation

8. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: test
9. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
10. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
11. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
12. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
13. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
14. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
15. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
16. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
17. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
18. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation

19. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
20. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
21. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
22. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
23. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
24. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
25. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
26. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
27. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
28. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
29. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
30. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)

label: code-design

31. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
32. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
33. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
34. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
35. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
36. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
37. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
38. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
39. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
40. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
41. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
42. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit

message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
43. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
44. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
45. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
46. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
47. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
48. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
49. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
50. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
51. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
52. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
53. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
54. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils

body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design

55. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: test
56. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
57. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
58. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
59. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
60. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
61. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
62. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
63. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
64. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
65. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
66. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils

   body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
   label: documentation
67. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
68. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
69. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
70. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
71. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
72. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: documentation
73. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
74. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
75. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
76. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
77. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils

body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design

78. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
79. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
80. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
81. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
82. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
83. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
84. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
85. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
86. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
87. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
88. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils

body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)

89. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
90. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
91. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
92. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
93. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
94. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
95. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
96. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
97. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
98. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
    label: code-design
99. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
    body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit

message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design

100. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
101. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
102. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
103. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
104. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
105. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
106. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
107. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
108. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
109. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: test
110. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
111. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils).

The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)

112. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
     body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
113. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
     body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
114. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
     body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
115. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
     body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
116. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
     body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
117. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
     body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
118. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
     body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
119. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
     body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
120. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
     body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
121. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
     body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
     label: test
122. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
     body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
123. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
     body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)

124. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
125. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
126. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
127. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
128. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
129. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
130. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
131. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
132. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
133. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: test
134. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
135. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils).

The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation

136. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
137. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
138. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
139. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
140. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
141. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
142. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
143. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
144. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
145. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: documentation
146. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design
147. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils

body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
label: code-design

148. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
149. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
150. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)
151. **title:** KAFKA-6473: Add MockProcessorContext to public test-utils
body: We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes)

github_pulls_comments:

1. **body:** TODO: I still need to write html docs for this, but I wanted to kick off the reviews of the implementation and test.
label: documentation
2. \cc @bbejeck
3. Ah, thanks for adding reviewers @mjsax . The tests failed check-style because I have single-line methods. I'll fix it tomorrow.
4. Ok, I think I've addressed all the concerns so far. I've started on the documentation, but I'm calling it a night for now.
5. LGTM!
6. Thanks for the thorough review, @mjsax ! I've addressed your comments.
7. **body:** Ok, @mjsax I've either responded to your latest comments or made changes. Please let me know what you think now. I think the question about the structure of the docs is particularly important. I explained my thoughts here: https://github.com/apache/kafka/pull/4736#discussion_r176450178 . I've taken the liberty of adding the links I proposed. If we don't like what I did, I can take them back out. As I've mentioned elsewhere, I think one of the bis strengths of this project is ease-of-use, especially for folks getting off the ground, so it's super important for the docs to cater to them. Not saying that the approach I've taken is better by this metric, just affirming that that's what I'm going for. I'm super open to more discussion about the best way to document this feature. \cc @bbejeck @guozhangwang @dguy @miguno
label: documentation
8. **body:** Hmm... I think I'll also add a test to the WordCountProcessorDemo...
label: test
9. **body:** WordCountProcessorDemo -> maybe different PR (it's easier to review short PRs and the other class is not related to this work) PR looks good overall. Let's figure out the doc question, and I'll do one more pass after we decided on this.
label: documentation
10. **body:** Ah, I added the example before I saw your last comment @mjsax . I recognize it's easier to review smaller PRs. FWIW, writing that example led me down a different code path and exposed a shortcoming of my implementation. Namely, if folks register a state store with logging enabled, the context needs to supply a RecordCollector. I opted to supply a no-op collector. An alternative I considered: throw an exception explaining they should disable logging in state stores for testing. I think no-op will offer a better experience.
label: code-design
11. retest this please
12. Made another pass over the PR. As for the docs, I'd prefer we concentrate on the testing section and let other sections just refer to it. > FWIW, writing that example led me down a different code path and exposed a shortcoming of my implementation. Namely, if folks register a state store with logging enabled, the context needs to supply a RecordCollector. I opted to supply a no-op collector. That is a good question. And we have actually seen people encountering this in their tests. I think it is valuable to test that changelog records are successfully sent, and hence I'd suggest we add an internal mock record collector (we do not need to expose it in the public API), which will be returned in `MockProcessContext.recordCollector` that will keep track of its send calls.
13. @guozhangwang Thanks for the review. I'm not opposed to adding it, but just to play devil's advocate for a sec... As written the MockProcessorContext is for testing Processors, but a mocked RecordCollector would be for testing the state store, no? I believe it's an important distinction, since as written the mock is not fully capable of managing all kinds of state stores, since it doesn't wire in a StateManager. We decided that processor implementations should be agnostic to the state store implementation, so it should be sufficient for testing to provide an in-memory store. If anything, I think this is an argument to throw `new UnsupportedOperationException("This mock does not provide record collection. Please reconfigure your state store or processor not to expect it for testing.")` in `MockProcessorContext#recordCollector()`. WDYT? Have I missed the point?
14. **body:** I see your point about the mock record collector now, good point. To rephrase it: We should claim to users that MockProcessorContext is only for testing your `process/transform` functions, and in which even if you have a hooked state store, it should not be a fully equipped one as it should not be in your testing scope. You should, instead, only use a mocking store instead. For now we should advocate users to use some in-memory state stores as a workaround for the mock, and in the longer term we'd have

https://issues.apache.org/jira/browse/KAFKA-6460 to cover that for you. I think this is a better philosophy, and for that I think it's appropriate to throw UnsupportedException indicating that this function should never be called.
label: code-design

15. **body:** I agree that throwing `UnsupportedOperationException` is a good solution -- maybe @vvcephei can tackle 6460 next to get both into 1.2 at once? What is the status for the docs question? @bbejeck and @miguno did not comment yet I think.
    label: documentation
16. @mjsax : I'm going to move the MPC docs into the testing document and link to it from the PAPI doc. I've just pushed a changeset containing the exception instead of a no-op recordCollector.
17. Sorry for the late response, but I'm in favor of moving the MPC into testing docs and linking to it from PAPI docs.
18. **body:** Ok, I reorganized the docs to put MPC in the testing section. Here's what it looks like rendered: ![screenshot-2018-3-23 apache kafka](https://user-images.githubusercontent.com/832787/37856828-a2a87378-2ec4-11e8-87f4-c95d88b976d9.png)
    label: documentation
19. retest this please
20. resolved those last issues and rebased. I'll make sure the tests pass and then ping for final reviews.
21. Hey @guozhangwang & @dguy , I think we're ready to merge this. Do you all mind taking a look?
22. Merged to `trunk`.

github_pulls_reviews:

1. **body:** Nit: either import those cases or change to `{@link org.apache.kafka.streams.kstream.Transformer Transformer}` to get rid of the long package name in the rendered JavaDocs. Applies multiple times.
   label: code-design
2. **body:** I would omit this (not the test, just the sentence) and put examples into the web docs.
   label: documentation
3. dummy -> minimal source-processor-sink ?
4. **body:** nit: no new paragraph required
   label: code-design
5. those are all `final` -- comment can be omitted.
6. **body:** nit: `{@code null}`
   label: code-design
7. **body:** Can we simplify the param-doc? Mabye: "The MockProcessorContext allow a `Processor` to access those config during runtime?" It it required to lost all of those with the corresponding methods? nit `{@link StreamsConfig}`
   label: code-design
8. **body:** Nit: We usually have `@Override` in it's own line. Might be better to keep an uniform code style?
   label: code-design
9. **body:** Nit: we usually declare all members at the beginning of the class -- what about unique code style?
   label: code-design
10. as above
11. **body:** nit: line too long
    label: code-design
12. What about adding a `isCanceled` flag?
13. **body:** nit: break line
    label: code-design
14. Child-Index was deprecated recently -- should we remove it? Also, with KIP-251, we should capture output timestamps, too.
15. Was deprecated recently. Should we throw similar to `schedule()`?
16. Was deprecated recently. Should we throw similar to schedule()?
17. does this add much value?
18. I think we could add a constructor with zero parameters, too?
19. **body:** nit: rename test -> `shouldCaptureOutputRecords` ?
    label: code-design
20. **body:** ah, cool. I never bothered with the javadoc formatting before.
    label: documentation
21. Yeah, maybe it's not that nice to get the list of effective properties, in exchange for a tidier doc.
22. I was testing you to learn your preference ;) I'll move the fields and inner classes to the beginning of the class.
23. **body:** Roger the timestamps. I've added deprecation notices to these methods. I think as long as the corresponding methods are still on the Processor, we should let the mock capture them. ChildIndex and childName may be in the code base for a long time, for one thing. For another, I want migrating tests to the mock to be a pleasant experience. Plus, folks will already get compiler and IDE warnings about the deprecated methods. The one exception is `public void schedule(final long interval)`, but it's because I just cannot support it in a mock. The fact that it's deprecated gives me an excuse to not support it.
    label: code-design
24. **body:** Oh, haha. It did when I was actually casting ;) I'm removing it.
    label: code-design
25. **body:** nit: maybe `for users to test their Processor, Transformer, and ValueTransformer implementations`?
    label: code-design
26. I'd suggest when editing on `https://kafka.apache.org/11/documentation/streams/developer-guide/testing.html` state clearly how to switch from an in-memory store to a persistent store when move from unit test code to production code.
27. Hmm.. is this correct? If `forward(kv)` is called without childName or childIndex, it means sending to all children. So should this be `capture.childName == null || ...` ? Ditto above in line 414.
28. maybe add some data that are `forward(key + value)` directly, and verify they are also returned below in `forwarded`?
29. **body:** If we are already deprecating the API with childIndex, should we simply throw an unsupported exception in the deprecate function and remove this field here as well? WDOT?
    label: code-design
30. \cc @bbejeck @guozhangwang @dguy WDYT?
31. **body:** For this specific API, I suspect it is ever commonly used in PAPI, so I'm fine with not supporting it right away, also as a way to encourage users to change code sooner than later, if there's anyone.
    label: code-design

32. **body:** since this testing forwarding to child nodes by name and we have this test above, do we need to have it again here?
    label: test
33. same as above
34. I'm +1 on supporting the timestamps, even if it's not commonly used now, users will often look to tests for example usage (at least I do). I'm also +1 on removing `childIndex` for the same reason, but I don't have too strong an opinion on that.
35. I like that better.
36. **body:** good idea. Maybe I'll include a link to that page in the javadoc?
    label: documentation
37. **body:** ah, yeah, that makes more sense. I was thinking in terms of reflecting exactly what calls to `forward(...)` there were, but it's probably more useful to mimic the regular forwarding logic.
    label: code-design
38. I think you mean adding something like: ```java this.context().forward(key, value); ``` and verify it gets forwarded to all children?
39. Sounds good.
40. yup.
41. This link might break in the future. Do you really want to put it in?
42. We don't strictly need it, but these tests are partially intended to demonstrate example usage, so I wrote it in "scenario style".
43. **body:** I think it's nice to link to it from here. We can add the stablest link we're aware of (I'm not sure what `/11/` is for). If it does break, it will at least clue the reader in that there *is* more documentation out there, and they can search until they find it. I have had this exact thing happen to me before, and found useful information that way. But I understand it's also setting us up to have a broken link in our docs (since there's no process in place to verify it resolves), which doesn't make us look great if/when it breaks. I don't feel strongly either way.
    label: documentation
44. nice rework of this part. Where is the new section about the `MockProcessorContext`? Did I miss it?
45. **body:** nit: move to top of class ?
    label: code-design
46. **body:** nit: `kv` -> `keyValue` (thought the whole class) -- IMHO, we should avoid abbreviations to improved code readability
    label: code-design
47. **body:** nit: top of class
    label: code-design
48. **body:** nit: top of class
    label: code-design
49. **body:** I see the following problem: either we link to a stable version like 11 or the "latest" version. However, for the first case, we need to update the link each release (and I am pretty sure we will forget at some point and thus link to the wrong version). For the second case, the link "changes" dynamically, and will also link to the wrong version. Thus, I would prefer to not put links to the docs into the JavaDoc. It's a paint maintain them.
    label: documentation
50. Do we need to set both?
51. **body:** Nit: What does `FK, `FV` stand for (ie, `F`)? Why not `K` and `V` ?
    label: code-design
52. **body:** This method is also deprecated. We should throw same exception as for `childIndex`.
    label: code-design
53. **body:** nit: `capturedForwards.clear()`
    label: code-design
54. **body:** Nit: `Reset` (in the past, we agreed to use imperative always)
    label: code-design
55. **body:** nit: `This` -> `{@code MockProcessorContext}` "this" , "here" etc is bad style IMHO
    label: code-design
56. **body:** nit: `{@code CapturedPunctuator} holds captured punctuators, along with their scheduling information.`
    label: code-design
57. **body:** nit: remove obvious comment (it's not `final`) -- comments should never repeat what the code "says" (comment should explain "why" something is done -- not what is done)
    label: code-design
58. **body:** Nit: why not `private final String childName; // nullable` (would be consistent with L60)
    label: code-design
59. **body:** Nit: `A method for getting...` sounds clumsy to me
    label: code-design
60. **body:** Nit: `A method for...` (as above)
    label: code-design
61. As above
62. **body:** Nit: I don't think we need JavaDocs in tests.
    label: documentation
63. **body:** nit: remove `this` if not required (code style)
    label: code-design
64. **body:** as above: the method name documents this already -- that's why we use `shouldDoX` in method names (allows to avoid JavaDocs in tests)
    label: documentation
65. as above
66. the broadcast was tested above already -- we should remove it -- one test for one feature
67. **body:** nit: remove (was tested already)
    label: code-design
68. as above
69. **body:** nit: `shouldThrowIfForwardedWithDeprecatedChildIndex`
    label: code-design
70. **body:** not used: can be removed
    label: code-design
71. **body:** nit: `uoe` -> `expected`
    label: code-design

72. Need to add a similar test for deprecates `childName` forwarding
73. **body:** nit: `child` -> `toChild`
    label: code-design
74. **body:** nit: use static imports to get rid of `Assert.`
    label: code-design
75. **body:** nit: use static imports to get rid of `Assert.`
    label: code-design
76. **body:** nit: remove -- not used
    label: code-design
77. explain why `Integer`, `Long` is used instead of `int`, `long`
78. duplicate line
79. No, I hadn't added it yet. It's pushed now. I decided to put it alongside the PAPI docs.
80. Why do you add this here? It's already linked in L48
81. `<>` -> `lt;&gt;`
82. **body:** Not sure about this. To use the `MockProcessorContext` users need to add the dependency -- this is explained in the "Testing" section and we should not explain it twice. If `MockProcesorContext` is in PAPI section, people might be confused as the class is not available to them by default. It would be useful to link to the "Testing" section from PAPI guide though. Not sure what others think. \cc @bbejeck @guozhangwang @dguy @miguno
    label: documentation
83. I see, you just mention it here. Might be fine. Still not sure if it might be better to put both into "Testing" section.
84. **body:** Should we remove `final` ? It's out internal code style but is not required here IMHO.
    label: code-design
85. **body:** nit: wrap line ?
    label: code-design
86. **body:** guess we can omit this. It's a test of `resetForward` that people would not put into their code when testing their own processor
    label: test
87. as above; can be removed IMHO
88. Ok, I'l take out the url.
89. yes, both configs are required.
90. ah, that's left over from by brief adventure with generifying this class. I'll change to K/V
91. k.
92. It was an annotation I turned into a comment when I realized the Nullable annotation in scope is a Sun internal. Just explaining. I'll change it, since consistency is good.
93. actually, I'm just deleting both comments. It's dumb to document nullable references with comments, since it provides no findbugs/warnings value, and all object references are nullable.
94. These tests are partly intended to demonstrate the use of the mock context. If you still think it's silly, I'll take them out.
95. Here, I'm verifying that it works in conjunction with send-to-child-by-name.
96. I can't; it's required.
97. opsh!
98. **body:** so that they throw an exception when gotten before setting, which is always a test bug since the real context always sets them first. Otherwise, they'll return -1, leading to longer test debugging cycles.
    label: test
99. Did you see this comment?
100. `<>` must be updated to `&lt;&gt;` AFAIK.
101. **body:** I understand why. But other contributors might not (and I might forget why in the future and want to change it...). It's not obvious from the code and thus should be explained with a commend, IMHO.
    label: code-design
102. **body:** Where does it fail if we don't specify them? In KS they are required, because we pass the configs into a `StreamsConfig` -- but we don't use `StreamsConfig` (if I did not miss this) thus I think we can simplify the code here and just pass in empty `Properties`
    label: code-design
103. Just a personal preference. We can also leave it for this case -- in general we might not want to put JavaDocs into tests.
104. Can you point out where it would fail? Unclear to me atm.
105. Oh! My bad. I overlooked it and thought it was missing. Sorry.
106. **body:** Yeah, that's where I started to put it, but after some writing and thinking about it, I think it's better to organize it by theme, with liberal hyperlinks. By theme, I mean: "I am using the Streams DSL, and now I need to test it" or "I am using the PAPI, and now I need to test it", versus " I need to write some tests". In other words, while the boundaries are obviously blurry, I think your PAPI user is a different persona from your DSL user. It's better to structure documentation around personas, to keep users on task while they're learning the system. I think the "write a streams app" page is long enough to justify splitting testing into a separate page, but it should probably link to it at the bottom, like "Next, you'll want to test your topology: <link>". That said, I do think we are missing some links. I'd propose to add: * The testing page should mention the MockProcessorContext and link here. * The "write a streams app" page should probably link to the testing page. One final justification: the thing that got me thinking about organizing the docs this way is that unit testing a Processor is literally not "testing a streams app", since a processor is not an app, so it felt weird to put it under that heading. As with all things, I'm open to adopting your suggestion, but that was my rationale...
    label: documentation
107. **body:** It's a question of style either way. I'd prefer to set a good example.
    label: code-design
108. **body:** It won't wrap in the rendered text, and this keeps the html lines shorter and therefore more readable for us.
    label: code-design
109. huh. That was an oversight on my part, but it actually renders correctly, and IDEA didn't warn about it. Maybe since `<>` isn't valid HTML, it just interprets it as text... Anyway, I'm escaping it now.
110. **body:** Actually, I was trying to plant the seed: "ah, I can reset, and then the list is empty, so I can probably process more stuff and verify more forwards...". That was the spirit of the behavioral tests. But if you think: 1) you'd rather not have behavioral tests, 2) people won't read the tests to learn how to use it, or 3) the behavioral test doesn't express what I was trying to express, I'm happy to change it.
    label: test
111. **body:** Yep, as discussed "above", there is a link, although I'd still appreciate others' thoughts about the structure of the docs anyway.
    label: documentation

112. **body:** Alternatively, I could check for null in the getters and throw an exception explaining that the fields must be initialized before use. This would document the situation for us, as well as explain it more clearly for users.
label: documentation
113. ``` org.apache.kafka.common.config.ConfigException: Missing required configuration "application.id" which has no default value. at org.apache.kafka.common.config.ConfigDef.parseValue(ConfigDef.java:472) at org.apache.kafka.common.config.ConfigDef.parse(ConfigDef.java:462) at org.apache.kafka.common.config.AbstractConfig.<init>(AbstractConfig.java:62) at org.apache.kafka.common.config.AbstractConfig.<init>(AbstractConfig.java:75) at org.apache.kafka.streams.StreamsConfig.<init>(StreamsConfig.java:703) at org.apache.kafka.streams.processor.MockProcessorContext.<init>(MockProcessorContext.java:190) at org.apache.kafka.streams.MockProcessorContextTest.fullConstructorShouldSetAllExpectedAttributes(MockProcessorContextTest.java:415) ``` and ``` org.apache.kafka.common.config.ConfigException: Missing required configuration "bootstrap.servers" which has no default value. at org.apache.kafka.common.config.ConfigDef.parseValue(ConfigDef.java:472) at org.apache.kafka.common.config.ConfigDef.parse(ConfigDef.java:462) at org.apache.kafka.common.config.AbstractConfig.<init>(AbstractConfig.java:62) at org.apache.kafka.common.config.AbstractConfig.<init>(AbstractConfig.java:75) at org.apache.kafka.streams.StreamsConfig.<init>(StreamsConfig.java:703) at org.apache.kafka.streams.processor.MockProcessorContext.<init>(MockProcessorContext.java:190) at org.apache.kafka.streams.MockProcessorContextTest.fullConstructorShouldSetAllExpectedAttributes(MockProcessorContextTest.java:415) ```
114. Agreed. I've removed it.
115. ok, I'll remove them.
116. It also crossed my mind to put the behavioral tests in a separate file, not sure what you think about that...
117. **body:** ah, I commented on the wrong usage. See my stacktraces comment on the MockProcessorContext constructor.
label: documentation
118. Ah. Thanks. I missed the line in the constructor when a `StreamsConfig` is created -- thought there is no `StreamsConfig`. Makes sense now.
119. **body:** Sounds good. Would be helpful for users to understand. The exception message should explain what they need to do to avoid the exception.
label: code-design
120. **body:** > In other words, while the boundaries are obviously blurry, I think your PAPI user is a different persona from your DSL user. Agreed. Adding more cross links is always good, IMHO. Let's see what others think.
label: documentation
121. Fair enough.
122. **body:** About the docs, I think we concentrate most of the contents in the `documentation/streams/developer-guide/testing` and in all other sections like PAPI, we can just add a note and refer to this link when necessary
label: documentation
123. **body:** nit. Add `{ }` to block (we always use them). Same below.
label: code-design
124. **body:** `assertX` has expected value as first parameter -- we should switch both to avoid confusing error messages. Applied to whole class.
label: code-design
125. ah, auto-format must have stripped them at some point. I'll check my settings.
126. gah! I will switch them all. :(
127. sorry about that.

jira_issues:

1. **summary:** Add MockProcessorContext to public test-utils
description: With KIP-247, we added public test-utils artifact with a TopologyTestDriver class. Using the test driver for a single Processor/Transformer/ValueTransformer it's required to specify a whole topology with source and sink and plus the Processor/Transformer/ValueTransformer into it. For unit testing, it might be more convenient to have a MockProcessorContext, that can be used to test the Processor/Transformer/ValueTransformer in isolation. Ie, the test itself creates new Processor/Transformer/ValueTransformer object and calls init() manually passing in the MockProcessorContext. This is a public API change and requires a KIP: https://cwiki.apache.org/confluence/display/KAFKA/Kafka+Improvement+Proposals
2. **summary:** Add MockProcessorContext to public test-utils
description: With KIP-247, we added public test-utils artifact with a TopologyTestDriver class. Using the test driver for a single Processor/Transformer/ValueTransformer it's required to specify a whole topology with source and sink and plus the Processor/Transformer/ValueTransformer into it. For unit testing, it might be more convenient to have a MockProcessorContext, that can be used to test the Processor/Transformer/ValueTransformer in isolation. Ie, the test itself creates new Processor/Transformer/ValueTransformer object and calls init() manually passing in the MockProcessorContext. This is a public API change and requires a KIP: https://cwiki.apache.org/confluence/display/KAFKA/Kafka+Improvement+Proposals
3. **summary:** Add MockProcessorContext to public test-utils
description: With KIP-247, we added public test-utils artifact with a TopologyTestDriver class. Using the test driver for a single Processor/Transformer/ValueTransformer it's required to specify a whole topology with source and sink and plus the Processor/Transformer/ValueTransformer into it. For unit testing, it might be more convenient to have a MockProcessorContext, that can be used to test the Processor/Transformer/ValueTransformer in isolation. Ie, the test itself creates new Processor/Transformer/ValueTransformer object and calls init() manually passing in the MockProcessorContext. This is a public API change and requires a KIP: https://cwiki.apache.org/confluence/display/KAFKA/Kafka+Improvement+Proposals
4. **summary:** Add MockProcessorContext to public test-utils
description: With KIP-247, we added public test-utils artifact with a TopologyTestDriver class. Using the test driver for a single Processor/Transformer/ValueTransformer it's required to specify a whole topology with source and sink and plus the Processor/Transformer/ValueTransformer into it. For unit testing, it might be more convenient to have a MockProcessorContext, that can be used to test the Processor/Transformer/ValueTransformer in isolation. Ie, the test itself creates new Processor/Transformer/ValueTransformer object and calls init() manually passing in the MockProcessorContext. This is a public API change and requires a KIP: https://cwiki.apache.org/confluence/display/KAFKA/Kafka+Improvement+Proposals

jira_issues_comments:

1. I have created [KIP-267|https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils] to propose a solution to this.
2. vvcephei opened a new pull request #4736: KAFKA-6473: Add MockProcessorContext to public test-utils URL: https://github.com/apache/kafka/pull/4736 We are adding a public testing utility to make it easier to unit test Processor implementations. See KIP-267 (https://cwiki.apache.org/confluence/display/KAFKA/KIP-267%3A+Add+Processor+Unit+Test+Support+to+Kafka+Streams+Test+Utils). The testing for this change is in this commit. There are behavioral and unit tests. ### Committer Checklist (excluded from commit message) - [ ] Verify design and implementation - [ ] Verify test coverage and CI build status - [ ] Verify documentation (including upgrade notes) -- This is an automated message from the Apache Git Service. To respond to the message, please log on GitHub and use the URL above to go to the specific comment. For queries about this service, please contact Infrastructure at: users@infra.apache.org
3. mjsax closed pull request #4736: KAFKA-6473: Add MockProcessorContext to public test-utils URL: https://github.com/apache/kafka/pull/4736 This is a PR merged from a forked repository. As GitHub hides the original diff on merge, it is displayed below for the sake of provenance: As this is a foreign pull request (from a fork), the diff is supplied below (as it won't show otherwise due to GitHub magic): diff --git a/build.gradle b/build.gradle index 8de03efeba0..d5fd7d5fd7b 100644 --- a/build.gradle +++ b/build.gradle @@ -1002,6 +1002,10 @@ project(':streams:examples') { compile project(':streams') compile project(':connect:json') // this dependency should be removed after we unify data API compile libs.slf4jlog4j + + testCompile project(':clients').sourceSets.test.output // for org.apache.kafka.test.IntegrationTest + testCompile project(':streams:test-utils') + testCompile libs.junit } javadoc { diff --git a/docs/streams/developer-guide/dsl-api.html b/docs/streams/developer-guide/dsl-api.html index 34ac89ffe05..8552bcc8674 100644 --- a/docs/streams/developer-guide/dsl-api.html +++ b/docs/streams/developer-guide/dsl-api.html @@ -66,6 +66,7 @@ </ul> </li> <li><a class="reference internal" href="#writing-streams-back-to-kafka" id="id25">Writing streams back to Kafka</a></li> + <li><a class="reference internal" href="#testing-a-streams-app" id="id26">Testing a Streams application</a></li> </ul> </div> <div class="section" id="overview"> @@ -3154,6 +3155,10 @@ <h2><a class="toc-backref" href="#id7">Overview</a><a class="headerlink" href="# retry on delivery failure or to prevent message duplication).</p> </div> </div> + <div class="section" id="testing-a-streams-app"> + <a class="headerlink" href="#testing-a-streams-app" title="Permalink to this headline"><h2>Testing a Streams application</a></h2> + Kafka Streams comes with a <code>test-utils</code> module to help you test your application <a href="testing.html">here</a>. + </div> </div> diff --git a/docs/streams/developer-guide/processor-api.html b/docs/streams/developer-guide/processor-api.html index b51bc22cfe2..e3432b79b7c 100644 --- a/docs/streams/developer-guide/processor-api.html +++ b/docs/streams/developer-guide/processor-api.html @@ -41,13 +41,16 @@ <p class="topic-title first"><b>Table of Contents</b></p> <ul class="simple"> <li><a class="reference internal" href="#overview" id="id1">Overview</a></li> - <li><a class="reference internal" href="#defining-a-stream-processor" id="id2">Defining a Stream Processor</a></li> - <li><a class="reference internal" href="#state-stores" id="id3">State Stores</a><ul> - <li><a class="reference internal" href="#defining-and-creating-a-state-store" id="id4">Defining and creating a State Store</a></li> - <li><a class="reference internal" href="#fault-tolerant-state-stores" id="id5">Fault-tolerant State Stores</a></li> - <li><a class="reference internal" href="#enable-or-disable-fault-tolerance-of-state-stores-store-changelogs" id="id6">Enable or Disable Fault Tolerance of State Stores (Store Changelogs)</a></li> - <li><a class="reference internal" href="#implementing-custom-state-stores" id="id7">Implementing Custom State Stores</a></li> - </ul> + <li><a class="reference internal" href="#defining-a-stream-processor" id="id2">Defining a Stream + Processor</a></li> + <li><a class="reference internal" href="#unit-testing-processors" id="id9">Unit Testing Processors</a></li> + <li><a class="reference internal" href="#state-stores" id="id3">State Stores</a> + <ul> + <li><a class="reference internal" href="#defining-and-creating-a-state-store" id="id4">Defining and creating a State Store</a></li> + <li><a class="reference internal" href="#fault-tolerant-state-stores" id="id5">Fault-tolerant State Stores</a></li> + <li><a class="reference internal" href="#enable-or-disable-fault-tolerance-of-state-stores-store-changelogs" id="id6">Enable or Disable Fault Tolerance of State Stores (Store Changelogs)</a></li> + <li><a class="reference internal" href="#implementing-custom-state-stores" id="id7">Implementing Custom State Stores</a></li> + </ul> </li> <li><a class="reference internal" href="#connecting-processors-and-state-stores" id="id8">Connecting Processors and State Stores</a></li> </ul> @@ -98,11 +101,12 @@ <h2><a class="toc-backref" href="#id1">Overview</a><a class="headerlink" href="# callbacks with different <code class="docutils literal"><span class="pre">PunctuationType</span></code> types within the same processor by calling <code class="docutils literal"><span class="pre">ProcessorContext#schedule()</span></code> multiple times inside <code class="docutils literal"><span class="pre">init()</span></code> method.</p> <div class="admonition attention"> - <p class="first admonition-title">Attention</p> + <p class="first admonition-title"><b>Attention</b></p> <p class="last">Stream-time is only advanced if all input partitions over all input topics have new data (with newer timestamps) available. If at least one partition does not have any new data available, stream-time will not be advanced and thus <code class="docutils literal"><span class="pre">punctuate()</span></code> will not be triggered if <code class="docutils literal"><span class="pre">PunctuationType.STREAM_TIME</span></code> was specified. This behavior is independent of the configured timestamp extractor, i.e., using <code class="docutils literal"><span class="pre">WallclockTimestampExtractor</span></code> does not enable wall-clock triggering of <code class="docutils literal"><span class="pre">punctuate()</span></code>.</p> </div> + <p><b>Example</b></p> <p>The following example <code class="docutils literal"><span class="pre">Processor</span></code> defines a simple word-count algorithm and the following actions are performed:</p> <ul class="simple"> <li>In the <code class="docutils literal"><span class="pre">init()</span></code> method, schedule the punctuation every 1000 time units (the time unit is normally milliseconds, which in this example would translate to punctuation every 1 second) and retrieve the local state store by its name &#8220;Counts&#8221;.</li> @@ -159,6 +163,16 @@ <h2><a class="toc-backref" href="#id1">Overview</a><a class="headerlink" href="# arrived records for stateful processing needs like aggregations and joins. For more information, see the <a class="reference internal" href="#streams-developer-guide-state-store"><span class="std std-ref">state stores</span></a> documentation.</p> </div> </div> + <div class="section" id="unit-testing-processors"> + <h2> + <a class="toc-backref" href="#id9">Unit Testing Processors</a> + <a class="headerlink" href="#unit-testing-processors" title="Permalink to this headline"></a> + </h2> + <p> + Kafka Streams comes with a <code>test-utils</code> module to help you write unit tests for your + processors <a href="testing.html#unit-testing-processors">here</a>. + </p> + </div> <div class="section" id="state-stores"> <span id="streams-developer-guide-state-store"></span><h2><a class="toc-backref" href="#id3">State Stores</a><a class="headerlink" href="#state-stores" title="Permalink to this headline"></a></h2> <p>To implement a <strong>stateful</strong> <code class="docutils literal"><span class="pre">Processor</span></code> or <code class="docutils literal"><span class="pre">Transformer</span></code>, you must provide one or more state stores to the processor diff --git a/docs/streams/developer-guide/testing.html b/docs/streams/developer-guide/testing.html index e6886a1689f..ea2ae987c7e 100644 --- a/docs/streams/developer-guide/testing.html +++ b/docs/streams/developer-guide/testing.html @@ -18,26 +18,40 @@ <script><!--#include virtual="../../js/templateData.js" --></script> <script id="content-template" type="text/x-handlebars-template"> - <!-- h1>Developer Guide for Kafka Streams</h1 --> - <div class="sub-nav-sticky"> - <div class="sticky-top"> - <!-- div style="height:35px"> - <a href="/{{version}}/documentation/streams/">Introduction</a> - <a class="active-menu-item" href="/{{version}}/documentation/streams/developer-guide">Developer Guide</a> - <a href="/{{version}}/documentation/streams/core-concepts">Concepts</a> - <a href="/{{version}}/documentation/streams/quickstart">Run Demo App</a> - <a href="/{{version}}/documentation/streams/tutorial">Tutorial: Write App</a> - </div --> + <!-- h1>Developer Guide for Kafka

Streams</h1 --> + <div class="sub-nav-sticky"> + <div class="sticky-top"> + <!-- div style="height:35px"> + <a href="/{{version}}/documentation/streams/">Introduction</a> + <a class="active-menu-item" href="/{{version}}/documentation/streams/developer-guide">Developer Guide</a> + <a href="/{{version}}/documentation/streams/core-concepts">Concepts</a> + <a href="/{{version}}/documentation/streams/quickstart">Run Demo App</a> + <a href="/{{version}}/documentation/streams/tutorial">Tutorial: Write App</a> + </div --> + </div> </div> - </div> - - <div class="section" id="testing"> - <span id="streams-developer-guide-testing"></span><h1>Testing a Streams Application<a class="headerlink" href="#testing" title="Permalink to this headline"></a></h1> - <p> - To test a Kafka Streams application, Kafka provides a test-utils artifact that can be added as regular dependency to your test code base. - Example <code>pom.xml</code> snippet when using Maven: - </p> - <pre> + + <div class="section" id="testing"> + <span id="streams-developer-guide-testing"></span> + <h1>Testing Kafka Streams<a class="headerlink" href="#testing" title="Permalink to this headline"></a></h1> + <div class="contents local topic" id="table-of-contents"> + <p class="topic-title first"><b>Table of Contents</b></p> + <ul class="simple"> + <li><a class="reference internal" href="#test-utils-artifact">Importing the test utilities</a></li> + <li><a class="reference internal" href="#testing-topologytestdriver">Testing Streams applications</a> + </li> + <li><a class="reference internal" href="#unit-testing-processors">Unit testing Processors</a> + </li> + </ul> + </div> + <div class="section" id="test-utils-artifact"> + <h2><a class="toc-backref" href="#test-utils-artifact" title="Permalink to this headline">Importing the test + utilities</a></h2> + <p> + To test a Kafka Streams application, Kafka provides a test-utils artifact that can be added as regular + dependency to your test code base. Example <code>pom.xml</code> snippet when using Maven: + </p> + <pre> &lt;dependency&gt; &lt;groupId&gt;org.apache.kafka&lt;/groupId&gt; &lt;artifactId&gt;kafka-streams-test-utils&lt;/artifactId&gt; @@ -45,13 +59,21 @@ &lt;scope&gt;test&lt;/scope&gt; &lt;/dependency&gt; </pre> - <p> - The test-utils package provides a <code>TopologyTestDriver</code> that can be used pipe data through a <code>Topology</code> that is either assembled manually - using Processor API or via the DSL using <code>StreamsBuilder</code>. - The test driver simulates the library runtime that continuously fetches records from input topics and processes them by traversing the topology. - You can use the test driver to verify that your specified processor topology computes the correct result with the manually piped in data records. - The test driver captures the results records and allows to query its embedded state stores. - <pre> + </div> + <div class="section" id="testing-topologytestdriver"> + <h2><a class="toc-backref" href="#testing-topologytestdriver" title="Permalink to this headline">Testing a + Streams application</a></h2> + + <p> + The test-utils package provides a <code>TopologyTestDriver</code> that can be used pipe data through a + <code>Topology</code> that is either assembled manually + using Processor API or via the DSL using <code>StreamsBuilder</code>. + The test driver simulates the library runtime that continuously fetches records from input topics and + processes them by traversing the topology. + You can use the test driver to verify that your specified processor topology computes the correct result + with the manually piped in data records. + The test driver captures the results records and allows to query its embedded state stores. + <pre> // Processor API Topology topology = new Topology(); topology.addSource("sourceProcessor", "input-topic"); @@ -68,62 +90,66 @@ config.put(StreamsConfig.APPLICATION_ID_CONFIG, "test"); config.put(StreamsConfig.BOOTSTRAP_SERVERS_CONFIG, "dummy:1234"); TopologyTestDriver testDriver = new TopologyTestDriver(topology, config); - </pre> - <p> - The test driver accepts <code>ConsumerRecord</code>s with key and value type <code>byte[]</code>. - Because <code>byte[]</code> types can be problematic, you can use the <code>ConsumerRecordFactory</code> to generate those records - by providing regular Java types for key and values and the corresponding serializers. - </p> - <pre> + </pre> + <p> + The test driver accepts <code>ConsumerRecord</code>s with key and value type <code>byte[]</code>. + Because <code>byte[]</code> types can be problematic, you can use the <code>ConsumerRecordFactory</code> + to generate those records + by providing regular Java types for key and values and the corresponding serializers. + </p> + <pre> ConsumerRecordFactory&lt;String, Integer&gt; factory = new ConsumerRecordFactory&lt;&gt;("input-topic", new StringSerializer(), new IntegerSerializer()); testDriver.pipe(factory.create("key", 42L)); - </pre> - <p> - To verify the output, the test driver produces <code>ProducerRecord</code>s with key and value type <code>byte[]</code>. - For result verification, you can specify corresponding deserializers when reading the output record from the driver. - <pre> + </pre> + <p> + To verify the output, the test driver produces <code>ProducerRecord</code>s with key and value type + <code>byte[]</code>. + For result verification, you can specify corresponding deserializers when reading the output record from + the driver. + <pre> ProducerRecord&lt;String, Integer&gt; outputRecord = testDriver.readOutput("output-topic", new StringDeserializer(), new LongDeserializer()); - </pre> - <p> - For result verification, you can use <code>OutputVerifier</code>. - It offers helper methods to compare only certain parts of the result record: - for example, you might only care about the key and value, but not the timestamp of the result record. - </p> - <pre> + </pre> + <p> + For result verification, you can use <code>OutputVerifier</code>. + It offers helper methods to compare only certain parts of the result record: + for example, you might only care about the key and value, but not the timestamp of the result record. + </p> + <pre> OutputVerifier.compareKeyValue(outputRecord, "key", 42L); // throws AssertionError if key or value does not match - </pre> - <p> - <code>TopologyTestDriver</code> supports punctuations, too. - Event-time punctuations are triggered automatically based on the processed records' timestamps. - Wall-clock-time punctuations can also be triggered by advancing the test driver's wall-clock-time (the driver mocks wall-clock-time internally to give users control over it). - </p> - <pre> + </pre> + <p> + <code>TopologyTestDriver</code> supports punctuations, too. + Event-time punctuations are triggered automatically based on the processed records' timestamps. + Wall-clock-time punctuations can also be triggered by advancing the test driver's wall-clock-time (the + driver mocks wall-clock-time internally to give users control over it). + </p> + <pre> testDriver.advanceWallClockTime(20L); - </pre> - </div> - <p> - Additionally, you can access state stores via the test driver before or after a test. - Accessing stores before a test is useful to pre-populate a store with some initial values. - After data was processed, expected updates to the store can be verified. - </p> - <pre> + </pre> + <p> + Additionally, you can access state stores via the test driver before or after a test. + Accessing stores before a test is useful to pre-populate a store with some initial values. + After data was processed, expected updates to the store can be verified. + </p> + <pre> KeyValueStore store = testDriver.getKeyValueStore("store-name"); - </pre> - <p> - Note, that you should always close the test driver at the end to make sure all resources are release properly. - </p> - <pre> + </pre> + <p> + Note, that you should always close the test driver at the end to make sure all resources are release + properly. + </p> + <pre> testDriver.close(); - </pre> - - <h2>Example</h2> - <p> - The following example demonstrates how to use the test driver and helper classes. - The example creates a topology that computes the maximum value per key using a key-value-store. - While processing, no output is generated, but only the store is updated. - Output is only sent downstream based on event-time and wall-clock punctuations. - </p> - <pre> + </pre> + + <h3>Example</h3> + <p> + The following example demonstrates how to use the test driver and helper classes. + The example creates a topology that computes the maximum value per key using a key-value-store. + While processing, no output is generated, but only the store is updated. + Output is only sent downstream based on event-time and wall-clock punctuations. + </p> + <pre> private TopologyTestDriver testDriver; private KeyValueStore&lt;String, Long&gt; store; @@ -266,31 +292,147 @@ <h2>Example</h2> @Override public void close() {} } - </pre> - <div class="pagination"> - <div class="pagination"> - <a href="/{{version}}/documentation/streams/developer-guide/datatypes" class="pagination__btn pagination__btn__prev">Previous</a> - <a href="/{{version}}/documentation/streams/developer-guide/interactive-queries" class="pagination__btn pagination__btn__next">Next</a> - </div> + </pre> + </div> + <div class="section" id="unit-testing-processors"> + <h2> + <a class="headerlink" href="#unit-testing-processors" + title="Permalink to this headline">Unit Testing Processors</a> + </h2> + <p> + If you <a href="processor-api.html">write a Processor</a>, you will want to test it. + </p> + <p> + Because the <code>Processor</code> forwards its results to the context rather than returning them, + Unit testing requires a mocked context capable of capturing forwarded data for inspection. + For this reason, we provide a <code>MockProcessorContext</code> in <a href="#test-utils-artifact"><code>test-utils</code></a>. + </p> + <b>Construction</b> + <p> + To begin with, instantiate your processor and initialize it with the mock context: + <pre> +final Processor processorUnderTest = ...; +final MockProcessorContext context = new MockProcessorContext(); +processorUnderTest.init(context); + </pre> + If you need to pass configuration to your processor or set the default serdes, you can create the mock with + config: + <pre> +final Properties config = new Properties(); +config.put(StreamsConfig.APPLICATION_ID_CONFIG, "unit-test"); +config.put(StreamsConfig.BOOTSTRAP_SERVERS_CONFIG, ""); +config.put(StreamsConfig.DEFAULT_KEY_SERDE_CLASS_CONFIG, Serdes.String().getClass()); +config.put(StreamsConfig.DEFAULT_VALUE_SERDE_CLASS_CONFIG, Serdes.Long().getClass()); +config.put("some.other.config", "some config value"); +final MockProcessorContext context = new MockProcessorContext(config); + </pre> + </p> + <b>Captured data</b> + <p> + The mock will capture any values that your processor forwards. You can make assertions on them: + <pre> +processorUnderTest.process("key", "value"); + +final Iterator&lt;CapturedForward&gt; forwarded = context.forwarded().iterator(); +assertEquals(forwarded.next().keyValue(), new KeyValue&lt;&gt;(..., ...)); +assertFalse(forwarded.hasNext()); + +// you can reset forwards to clear the captured data. This may be helpful in constructing longer scenarios. +context.resetForwards(); + +assertEquals(context.forwarded().size(), 0); + </pre> + If your processor forwards to specific child processors, you can query the context for captured data by + child name: + <pre> +final List&lt;CapturedForward&gt; captures = context.forwarded("childProcessorName"); + </pre> + The mock also captures whether your processor has called <code>commit()</code> on the context: + <pre> +assertTrue(context.committed()); + +// commit captures can also be reset. +context.resetCommit(); + +assertFalse(context.committed()); + </pre> + </p> + <b>Setting record metadata</b> + <p> + In case your processor logic depends on the record metadata (topic, partition, offset, or timestamp), + you can set them on the context, either all together or individually: + <pre> +context.setRecordMetadata("topicName", /*partition*/ 0, /*offset*/ 0L, /*timestamp*/ 0L); +context.setTopic("topicName"); +context.setPartition(0); +context.setOffset(0L); +context.setTimestamp(0L); + </pre> + Once these are set, the context will continue returning the same values, until you set new ones. + </p> + <b>State stores</b> + <p> + In case your punctuator is stateful, the mock context allows you to register state stores. + You're encouraged to use a simple in-memory store of the appropriate type (KeyValue, Windowed, or + Session), since the mock context does <i>not</i> manage changelogs, state directories, etc. + </p> + <pre> +final KeyValueStore&lt;String, Integer&gt; store = + Stores.keyValueStoreBuilder(+ Stores.inMemoryKeyValueStore("myStore"), + Serdes.String(), + Serdes.Integer() +) + .withLoggingDisabled() // Changelog is not supported by MockProcessorContext. + .build(); +store.init(context, store); +context.register(store, /*deprecated parameter*/ false, /*parameter unused in mock*/ null); + </pre> + <b>Verifying punctuators</b> + <p> + Processors can schedule punctuators to handle periodic tasks. + The mock context does <i>not</i> automatically execute punctuators, but it does capture them to + allow you to unit test them as well: + <pre> +final MockProcessorContext.CapturedPunctuator capturedPunctuator = context.scheduledPunctuators().get(0); +final long interval = capturedPunctuator.getIntervalMs(); +final PunctuationType type = capturedPunctuator.getType(); +final boolean cancelled = capturedPunctuator.cancelled(); +final Punctuator punctuator = capturedPunctuator.getPunctuator(); +punctuator.punctuate(/*timestamp*/ 0L); + </pre> + If you need to write tests involving automatic firing of scheduled punctuators, we recommend creating a + simple topology with your processor and using the <a href="testing.html#testing-topologytestdriver"><code>TopologyTestDriver</code></a>. + </p> + </div> + </div> + <div class="pagination"> + <div class="pagination"> + <a href="/{{version}}/documentation/streams/developer-guide/datatypes" + class="pagination__btn pagination__btn__prev">Previous</a> + <a href="/{{version}}/documentation/streams/developer-guide/interactive-queries" + class="pagination__btn pagination__btn__next">Next</a> + </div> + </div> </script> <!--#include virtual="../../../includes/_header.htm" --> <!--#include virtual="../../../includes/_top.htm" --> <div class="content documentation documentation--current"> - <!--#include virtual="../../../includes/_nav.htm" --> - <div class="right"> - <!--#include virtual="../../../includes/_docs_banner.htm" --> - <ul class="breadcrumbs"> - <li><a href="/documentation">Documentation</a></li> - <li><a href="/documentation/streams">Kafka Streams</a></li> - <li><a href="/documentation/streams/developer-guide/">Developer Guide</a></li> - </ul> - <div class="p-content"> </div> - </div> + <!--#include virtual="../../../includes/_nav.htm" --> + <div class="right"> + <!--#include virtual="../../../includes/_docs_banner.htm" --> + <ul class="breadcrumbs"> + <li><a href="/documentation">Documentation</a></li> + <li><a href="/documentation/streams">Kafka Streams</a></li> + <li><a href="/documentation/streams/developer-guide/">Developer Guide</a></li> + </ul> + <div class="p-content"></div> + </div> </div> <!--#include virtual="../../../includes/_footer.htm" --> <script> - $(function() { + $(function () { // Show selected style on nav item $('.b-nav__streams').addClass('selected'); @@ -299,7 +441,7 @@ <h2>Example</h2> y_pos = $navbar.offset().top, height = $navbar.height(); - $(window).scroll(function() { + $(window).scroll(function () { var scrollTop = $(window).scrollTop(); if (scrollTop > y_pos - height) { diff --git a/docs/streams/developer-guide/write-streams.html b/docs/streams/developer-guide/write-streams.html index 1e4213d1b81..44cdb3fd9ba 100644 --- a/docs/streams/developer-guide/write-streams.html +++ b/docs/streams/developer-guide/write-streams.html @@ -37,6 +37,7 @@ <ul class="simple"> <li><a class="reference internal" href="#libraries-and-maven-artifacts" id="id1">Libraries and Maven artifacts</a></li> <li><a class="reference internal" href="#using-kafka-streams-within-your-application-code" id="id2">Using Kafka Streams within your application code</a></li> + <li><a class="reference internal" href="#testing-a-streams-app" id="id3">Testing a Streams application</a></li> </ul> <p>Any Java application that makes use of the Kafka Streams library is considered a Kafka Streams application. The computational logic of a Kafka Streams application is defined as a <a class="reference internal" href="../concepts.html#streams-concepts"><span class="std std-ref">processor topology</span></a>, @@ -196,6 +197,11 @@ <h2>Using Kafka Streams within your application code<a class="headerlink" href=" <p>After an application is stopped, Kafka Streams will migrate any tasks that had been running in this instance to available remaining instances.</p> </div> + + <div class="section" id="testing-a-streams-app"> + <a class="headerlink" href="#testing-a-streams-app" title="Permalink to this headline"><h2>Testing a Streams application</a></h2> + Kafka Streams comes with a <code>test-utils</code> module to help you test your application <a href="testing.html">here</a>. + </div> </div> diff --git a/streams/examples/src/main/java/org/apache/kafka/streams/examples/wordcount/WordCountProcessorDemo.java b/streams/examples/src/main/java/org/apache/kafka/streams/examples/wordcount/WordCountProcessorDemo.java index cfa2137c8fa..dbf2b707c3f 100644 --- a/streams/examples/src/main/java/org/apache/kafka/streams/examples/wordcount/WordCountProcessorDemo.java +++ b/streams/examples/src/main/java/org/apache/kafka/streams/examples/wordcount/WordCountProcessorDemo.java @@ -49,7 +49,7 @@ */ public class WordCountProcessorDemo { - private static class MyProcessorSupplier implements ProcessorSupplier<String, String> { + static class MyProcessorSupplier implements ProcessorSupplier<String, String> { @Override public Processor<String, String> get() { diff --git a/streams/examples/src/test/java/org/apache/kafka/streams/examples/wordcount/WordCountProcessorTest.java b/streams/examples/src/test/java/org/apache/kafka/streams/examples/wordcount/WordCountProcessorTest.java new file mode 100644 index 00000000000..566b7d430ea --- /dev/null +++ b/streams/examples/src/test/java/org/apache/kafka/streams/examples/wordcount/WordCountProcessorTest.java @@ -0,0 +1,70 @@ +/* + * Licensed to the Apache Software Foundation (ASF) under one or more + * contributor license agreements. See the NOTICE file distributed with + * this work for additional information regarding copyright ownership. + * The ASF licenses this file to You under the Apache License, Version 2.0 + * (the "License"); you may not use this file except in compliance with + * the License. You may obtain a copy of the License at + * + * http://www.apache.org/licenses/LICENSE-2.0 + * + * Unless required by applicable law or agreed to in writing, software + * distributed under the License is distributed on an "AS IS" BASIS, + * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. + * See the License for the specific language governing permissions and + * limitations under the License. + */ +package org.apache.kafka.streams.examples.wordcount; + +import org.apache.kafka.common.serialization.Serdes; +import org.apache.kafka.streams.KeyValue; +import org.apache.kafka.streams.processor.MockProcessorContext; +import org.apache.kafka.streams.processor.Processor; +import org.apache.kafka.streams.state.KeyValueStore; +import org.apache.kafka.streams.state.Stores; +import org.junit.Test; + +import java.util.Iterator; + +import static org.junit.Assert.assertEquals; +import static org.junit.Assert.assertFalse; +import static org.junit.Assert.assertTrue; + +/** + * Demonstrate the use of {@link MockProcessorContext} for testing the {@link Processor} in the {@link WordCountProcessorDemo}. + */ +public class WordCountProcessorTest { + @Test + public void test() { + final MockProcessorContext context = new MockProcessorContext(); + + // Create, initialize, and register the state store. + final KeyValueStore<String, Integer> store = + Stores.keyValueStoreBuilder(Stores.inMemoryKeyValueStore("Counts"), Serdes.String(), Serdes.Integer()) + .withLoggingDisabled() // Changelog is not supported by MockProcessorContext. + .build(); + store.init(context, store); + context.register(store, false, null); + + // Create and initialize the processor under test + final Processor<String, String> processor = new WordCountProcessorDemo.MyProcessorSupplier().get(); + processor.init(context); + + // send a record to the processor + processor.process("key", "alpha beta gamma alpha"); + + // note that the processor commits, but does not forward, during process() + assertTrue(context.committed()); + assertTrue(context.forwarded().isEmpty()); + + // now, we trigger the punctuator, which iterates over the state store and forwards the contents. + context.scheduledPunctuators().get(0).getPunctuator().punctuate(0L); + + // finally, we can verify the output. + final Iterator<MockProcessorContext.CapturedForward> capturedForwards = context.forwarded().iterator(); + assertEquals(new KeyValue<>("alpha", "2"), capturedForwards.next().keyValue()); + assertEquals(new KeyValue<>("beta", "1"), capturedForwards.next().keyValue()); + assertEquals(new KeyValue<>("gamma", "1"), capturedForwards.next().keyValue()); + assertFalse(capturedForwards.hasNext()); + } +} diff --git a/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamKStreamJoinTest.java b/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamKStreamJoinTest.java index 572c0b02df5..56000234373 100644 --- a/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamKStreamJoinTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamKStreamJoinTest.java @@ -26,7 +26,7 @@ import org.apache.kafka.streams.kstream.KStream; import org.apache.kafka.streams.processor.internals.ProcessorRecordContext; import org.apache.kafka.test.KStreamTestDriver; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.MockProcessorSupplier; import org.apache.kafka.test.MockValueJoiner; import org.apache.kafka.test.TestUtils; @@ -719,6 +719,6 @@ public void testAsymetricWindowingBefore() { } private void setRecordContext(final long time, final String topic) { - ((MockProcessorContext) driver.context()).setRecordContext(new ProcessorRecordContext(time, 0, 0, topic)); + ((InternalMockProcessorContext) driver.context()).setRecordContext(new ProcessorRecordContext(time, 0, 0, topic)); } } diff --git a/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamKStreamLeftJoinTest.java b/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamKStreamLeftJoinTest.java index 39b318fec48..465082b7aa7 100644 --- a/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamKStreamLeftJoinTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamKStreamLeftJoinTest.java @@ -26,7 +26,7 @@ import org.apache.kafka.streams.kstream.KStream; import org.apache.kafka.streams.processor.internals.ProcessorRecordContext; import org.apache.kafka.test.KStreamTestDriver; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.MockProcessorSupplier; import org.apache.kafka.test.MockValueJoiner; import org.apache.kafka.test.TestUtils; @@ -304,6 +304,6 @@ public void testWindowing() { } private void setRecordContext(final long time, final String topic) { - ((MockProcessorContext) driver.context()).setRecordContext(new ProcessorRecordContext(time, 0, 0, topic)); + ((InternalMockProcessorContext) driver.context()).setRecordContext(new ProcessorRecordContext(time, 0, 0, topic)); } } diff --git a/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamSessionWindowAggregateProcessorTest.java b/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamSessionWindowAggregateProcessorTest.java index 21dc4f037c9..212c48d43f7 100644 --- a/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamSessionWindowAggregateProcessorTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamSessionWindowAggregateProcessorTest.java @@ -31,7 +31,7 @@ import org.apache.kafka.streams.state.internals.RocksDBSessionStoreSupplier; import org.apache.kafka.streams.state.SessionStore; import org.apache.kafka.streams.state.internals.ThreadCache; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.NoOpRecordCollector; import org.apache.kafka.test.TestUtils; import org.junit.After; @@ -81,13 +81,13 @@ public Long apply(final String aggKey, final Long aggOne, final Long aggTwo) { private final List<KeyValue> results = new ArrayList<>(); private Processor<String, String> processor = sessionAggregator.get(); private SessionStore<String, Long> sessionStore; - private MockProcessorContext context; + private InternalMockProcessorContext context; @Before public void initializeStore() { final File stateDir = TestUtils.tempDirectory(); - context = new MockProcessorContext(stateDir, + context = new InternalMockProcessorContext(stateDir, Serdes.String(), Serdes.String(), new NoOpRecordCollector(), new ThreadCache(new LogContext("testCache "), 100000, new MockStreamsMetrics(new Metrics()))) { @Override public <K, V> void forward(final K key, final V value) { diff --git a/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamWindowAggregateTest.java b/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamWindowAggregateTest.java index aa660e08680..d3a74e336cc 100644 --- a/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamWindowAggregateTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/kstream/internals/KStreamWindowAggregateTest.java @@ -31,7 +31,7 @@ import org.apache.kafka.test.KStreamTestDriver; import org.apache.kafka.test.MockAggregator; import org.apache.kafka.test.MockInitializer; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.MockProcessorSupplier; import org.apache.kafka.test.TestUtils; import org.junit.Before; @@ -143,7 +143,7 @@ public void testAggBasic() { } private void setRecordContext(final long time, final String topic) { - ((MockProcessorContext) driver.context()).setRecordContext(new ProcessorRecordContext(time, 0, 0, topic)); + ((InternalMockProcessorContext) driver.context()).setRecordContext(new ProcessorRecordContext(time, 0, 0, topic)); } @Test diff --git a/streams/src/test/java/org/apache/kafka/streams/processor/internals/AbstractTaskTest.java b/streams/src/test/java/org/apache/kafka/streams/processor/internals/AbstractTaskTest.java index 4569858e357..347e9c4fd75 100644 --- a/streams/src/test/java/org/apache/kafka/streams/processor/internals/AbstractTaskTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/processor/internals/AbstractTaskTest.java @@ -32,7 +32,7 @@ import org.apache.kafka.streams.errors.ProcessorStateException; import org.apache.kafka.streams.processor.StateStore; import org.apache.kafka.streams.processor.TaskId; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.MockRestoreCallback; import org.apache.kafka.test.MockStateRestoreListener; import org.apache.kafka.test.TestUtils; @@ -194,7 +194,7 @@ public void shouldDeleteAndRecreateStoreDirectoryOnReinitialize() throws IOExcep testFile4.createNewFile(); assertTrue(testFile4.exists()); - task.processorContext = new MockProcessorContext(stateDirectory.directoryForTask(task.id), streamsConfig); + task.processorContext = new InternalMockProcessorContext(stateDirectory.directoryForTask(task.id), streamsConfig); task.stateMgr.register(store1, new MockRestoreCallback()); task.stateMgr.register(store2, new MockRestoreCallback()); diff --git a/streams/src/test/java/org/apache/kafka/streams/processor/internals/GlobalStateManagerImplTest.java b/streams/src/test/java/org/apache/kafka/streams/processor/internals/GlobalStateManagerImplTest.java index df8d2010d24..d19e63e0543 100644 --- a/streams/src/test/java/org/apache/kafka/streams/processor/internals/GlobalStateManagerImplTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/processor/internals/GlobalStateManagerImplTest.java @@ -34,7 +34,7 @@ import org.apache.kafka.streams.processor.StateRestoreCallback; import org.apache.kafka.streams.processor.StateStore; import org.apache.kafka.streams.state.internals.OffsetCheckpoint; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.MockStateRestoreListener; import org.apache.kafka.test.NoOpReadOnlyStore; import org.apache.kafka.test.TestUtils; @@ -86,7 +86,7 @@ private MockConsumer<byte[], byte[]> consumer; private File checkpointFile; private ProcessorTopology topology; - private MockProcessorContext mockProcessorContext; + private InternalMockProcessorContext processorContext; @Before public void before() throws IOException { @@ -120,8 +120,8 @@ public void before() throws IOException { stateDirectory, stateRestoreListener, streamsConfig); - mockProcessorContext = new MockProcessorContext(stateDirectory.globalStateDir(), streamsConfig); - stateManager.setGlobalProcessorContext(mockProcessorContext); + processorContext = new InternalMockProcessorContext(stateDirectory.globalStateDir(), streamsConfig); + stateManager.setGlobalProcessorContext(processorContext); checkpointFile = new File(stateManager.baseDir(), ProcessorStateManager.CHECKPOINT_FILE_NAME); } @@ -631,7 +631,7 @@ public void shouldDeleteAndRecreateStoreDirectoryOnReinitialize() throws IOExcep assertTrue(testFile4.exists()); // only delete and recreate store 1 and 3 -- 2 and 4 must be untouched - stateManager.reinitializeStateStoresForPartitions(Utils.mkList(t1, t3), mockProcessorContext); + stateManager.reinitializeStateStoresForPartitions(Utils.mkList(t1, t3), processorContext); assertFalse(testFile1.exists()); assertTrue(testFile2.exists()); diff --git a/streams/src/test/java/org/apache/kafka/streams/processor/internals/ProcessorNodeTest.java b/streams/src/test/java/org/apache/kafka/streams/processor/internals/ProcessorNodeTest.java index 90ef771ed61..0dea1930be5 100644 --- a/streams/src/test/java/org/apache/kafka/streams/processor/internals/ProcessorNodeTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/processor/internals/ProcessorNodeTest.java @@ -24,7 +24,7 @@ import org.apache.kafka.streams.processor.Processor; import org.apache.kafka.streams.processor.ProcessorContext; import org.apache.kafka.streams.state.StateSerdes; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.junit.Test; import java.util.Collections; @@ -110,7 +110,7 @@ public void testMetrics() { final StateSerdes anyStateSerde = StateSerdes.withBuiltinTypes("anyName", Bytes.class, Bytes.class); final Metrics metrics = new Metrics(); - final MockProcessorContext context = new MockProcessorContext(anyStateSerde, new RecordCollectorImpl(null, null, new LogContext("processnode-test "), new DefaultProductionExceptionHandler()), metrics); + final InternalMockProcessorContext context = new InternalMockProcessorContext(anyStateSerde, new RecordCollectorImpl(null, null, new LogContext("processnode-test "), new DefaultProductionExceptionHandler()), metrics); final ProcessorNode node = new ProcessorNode("name", new NoOpProcessor(), Collections.emptySet()); node.init(context); diff --git a/streams/src/test/java/org/apache/kafka/streams/processor/internals/RecordQueueTest.java b/streams/src/test/java/org/apache/kafka/streams/processor/internals/RecordQueueTest.java index 2fa1b591822..faf72e92d2d 100644 --- a/streams/src/test/java/org/apache/kafka/streams/processor/internals/RecordQueueTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/processor/internals/RecordQueueTest.java @@ -34,7 +34,7 @@ import org.apache.kafka.streams.processor.LogAndSkipOnInvalidTimestamp; import org.apache.kafka.streams.processor.TimestampExtractor; import org.apache.kafka.streams.state.StateSerdes; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.MockSourceNode; import org.apache.kafka.test.MockTimestampExtractor; import org.junit.After; @@ -54,7 +54,7 @@ private final TimestampExtractor timestampExtractor = new MockTimestampExtractor(); private final String[] topics = {"topic"}; - final MockProcessorContext context = new MockProcessorContext(StateSerdes.withBuiltinTypes("anyName", Bytes.class, Bytes.class), + final InternalMockProcessorContext context = new InternalMockProcessorContext(StateSerdes.withBuiltinTypes("anyName", Bytes.class, Bytes.class), new RecordCollectorImpl(null, null, new LogContext("record-queue-test "), new DefaultProductionExceptionHandler())); private final MockSourceNode mockSourceNodeWithMetrics = new MockSourceNode<>(topics, intDeserializer, intDeserializer); private final RecordQueue queue = new RecordQueue(diff --git a/streams/src/test/java/org/apache/kafka/streams/processor/internals/SinkNodeTest.java b/streams/src/test/java/org/apache/kafka/streams/processor/internals/SinkNodeTest.java index 67927403fc2..4b48a17ab7c 100644 --- a/streams/src/test/java/org/apache/kafka/streams/processor/internals/SinkNodeTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/processor/internals/SinkNodeTest.java @@ -24,7 +24,7 @@ import org.apache.kafka.streams.errors.DefaultProductionExceptionHandler; import org.apache.kafka.streams.errors.StreamsException; import org.apache.kafka.streams.state.StateSerdes; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.junit.After; import org.junit.Before; import org.junit.Test; @@ -37,7 +37,7 @@ public class SinkNodeTest { private final Serializer anySerializer = Serdes.Bytes().serializer(); private final StateSerdes anyStateSerde = StateSerdes.withBuiltinTypes("anyName", Bytes.class, Bytes.class); - private final MockProcessorContext context = new MockProcessorContext(anyStateSerde, + private final InternalMockProcessorContext context = new InternalMockProcessorContext(anyStateSerde, new RecordCollectorImpl(new MockProducer<byte[], byte[]>(true, anySerializer, anySerializer), null, new LogContext("sinknode-test "), new DefaultProductionExceptionHandler())); private final SinkNode sink = new SinkNode<>("anyNodeName", "any-output-topic", anySerializer, anySerializer, null); diff --git a/streams/src/test/java/org/apache/kafka/streams/state/KeyValueStoreTestDriver.java b/streams/src/test/java/org/apache/kafka/streams/state/KeyValueStoreTestDriver.java index a3425856b15..fc810e3bc3b 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/KeyValueStoreTestDriver.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/KeyValueStoreTestDriver.java @@ -34,7 +34,7 @@ import org.apache.kafka.streams.processor.internals.RecordCollectorImpl; import org.apache.kafka.streams.state.internals.RocksDBKeyValueStoreTest; import org.apache.kafka.streams.state.internals.ThreadCache; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.MockTimestampExtractor; import org.apache.kafka.test.TestUtils; @@ -179,7 +179,7 @@ private final Set<K> flushedRemovals = new HashSet<>(); private final List<KeyValue<byte[], byte[]>> restorableEntries = new LinkedList<>(); - private final MockProcessorContext context; + private final InternalMockProcessorContext context; private final StateSerdes<K, V> stateSerdes; private KeyValueStoreTestDriver(final StateSerdes<K, V> serdes) { @@ -227,7 +227,7 @@ private KeyValueStoreTestDriver(final StateSerdes<K, V> serdes) { props.put(StreamsConfig.DEFAULT_VALUE_SERDE_CLASS_CONFIG, serdes.valueSerde().getClass()); props.put(StreamsConfig.ROCKSDB_CONFIG_SETTER_CLASS_CONFIG, RocksDBKeyValueStoreTest.TheRocksDbConfigSetter.class); - context = new MockProcessorContext(stateDir, serdes.keySerde(), serdes.valueSerde(), recordCollector, null) { + context = new InternalMockProcessorContext(stateDir, serdes.keySerde(), serdes.valueSerde(), recordCollector, null) { ThreadCache cache = new ThreadCache(new LogContext("testCache "), 1024 * 1024L, metrics()); @Override diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/AbstractKeyValueStoreTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/AbstractKeyValueStoreTest.java index 937b1d0b0bf..51c782ad9d5 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/AbstractKeyValueStoreTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/AbstractKeyValueStoreTest.java @@ -25,7 +25,7 @@ import org.apache.kafka.streams.state.KeyValueIterator; import org.apache.kafka.streams.state.KeyValueStore; import org.apache.kafka.streams.state.KeyValueStoreTestDriver; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.junit.After; import org.junit.Before; import org.junit.Test; @@ -48,14 +48,14 @@ protected abstract <K, V> KeyValueStore<K, V> createKeyValueStore(final ProcessorContext context); - protected MockProcessorContext context; + protected InternalMockProcessorContext context; protected KeyValueStore<Integer, String> store; protected KeyValueStoreTestDriver<Integer, String> driver; @Before public void before() { driver = KeyValueStoreTestDriver.create(Integer.class, String.class); - context = (MockProcessorContext) driver.context(); + context = (InternalMockProcessorContext) driver.context(); context.setTime(10); store = createKeyValueStore(context); } diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/CachingKeyValueStoreTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/CachingKeyValueStoreTest.java index 0e3b4e8223e..8705326b782 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/CachingKeyValueStoreTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/CachingKeyValueStoreTest.java @@ -33,7 +33,7 @@ import org.apache.kafka.streams.state.KeyValueStore; import org.apache.kafka.streams.state.StoreBuilder; import org.apache.kafka.streams.state.Stores; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.junit.After; import org.junit.Before; import org.junit.Test; @@ -58,7 +58,7 @@ public class CachingKeyValueStoreTest extends AbstractKeyValueStoreTest { private final int maxCacheSizeBytes = 150; - private MockProcessorContext context; + private InternalMockProcessorContext context; private CachingKeyValueStore<String, String> store; private InMemoryKeyValueStore<Bytes, byte[]> underlyingStore; private ThreadCache cache; @@ -73,7 +73,7 @@ public void setUp() { store = new CachingKeyValueStore<>(underlyingStore, Serdes.String(), Serdes.String()); store.setFlushListener(cacheFlushListener, false); cache = new ThreadCache(new LogContext("testCache "), maxCacheSizeBytes, new MockStreamsMetrics(new Metrics())); - context = new MockProcessorContext(null, null, null, (RecordCollector) null, cache); + context = new InternalMockProcessorContext(null, null, null, (RecordCollector) null, cache); topic = "topic"; context.setRecordContext(new ProcessorRecordContext(10, 0, 0, topic)); store.init(context, null); diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/CachingSessionStoreTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/CachingSessionStoreTest.java index 16ef47c1c1b..a9a66e9a7df 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/CachingSessionStoreTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/CachingSessionStoreTest.java @@ -30,7 +30,7 @@ import org.apache.kafka.streams.processor.internals.ProcessorRecordContext; import org.apache.kafka.streams.state.KeyValueIterator; import org.apache.kafka.streams.state.StateSerdes; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.TestUtils; import org.junit.After; import org.junit.Before; @@ -52,7 +52,7 @@ public class CachingSessionStoreTest { private static final int MAX_CACHE_SIZE_BYTES = 600; - private MockProcessorContext context; + private InternalMockProcessorContext context; private RocksDBSegmentedBytesStore underlying; private CachingSessionStore<String, String> cachingStore; private ThreadCache cache; @@ -75,7 +75,7 @@ public void setUp() { Segments.segmentInterval(retention, numSegments)); cache = new ThreadCache(new LogContext("testCache "), MAX_CACHE_SIZE_BYTES, new MockStreamsMetrics(new Metrics())); - context = new MockProcessorContext(TestUtils.tempDirectory(), null, null, null, cache); + context = new InternalMockProcessorContext(TestUtils.tempDirectory(), null, null, null, cache); context.setRecordContext(new ProcessorRecordContext(DEFAULT_TIMESTAMP, 0, 0, "topic")); cachingStore.init(context, cachingStore); } diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/CachingWindowStoreTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/CachingWindowStoreTest.java index bbf9bef03f3..c25655b6b2e 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/CachingWindowStoreTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/CachingWindowStoreTest.java @@ -30,7 +30,7 @@ import org.apache.kafka.streams.processor.internals.RecordCollector; import org.apache.kafka.streams.state.KeyValueIterator; import org.apache.kafka.streams.state.WindowStoreIterator; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.TestUtils; import org.junit.After; import org.junit.Before; @@ -56,7 +56,7 @@ private static final int MAX_CACHE_SIZE_BYTES = 150; private static final long DEFAULT_TIMESTAMP = 10L; private static final Long WINDOW_SIZE = 10000L; - private MockProcessorContext context; + private InternalMockProcessorContext context; private RocksDBSegmentedBytesStore underlying; private CachingWindowStore<String, String> cachingStore; private CachingKeyValueStoreTest.CacheFlushListenerStub<Windowed<String>, String> cacheListener; @@ -80,7 +80,7 @@ public void setUp() { cachingStore.setFlushListener(cacheListener, false); cache = new ThreadCache(new LogContext("testCache "), MAX_CACHE_SIZE_BYTES, new MockStreamsMetrics(new Metrics())); topic = "topic"; - context = new MockProcessorContext(TestUtils.tempDirectory(), null, null, (RecordCollector) null, cache); + context = new InternalMockProcessorContext(TestUtils.tempDirectory(), null, null, (RecordCollector) null, cache); context.setRecordContext(new ProcessorRecordContext(DEFAULT_TIMESTAMP, 0, 0, topic)); cachingStore.init(context, cachingStore); } diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/ChangeLoggingKeyValueBytesStoreTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/ChangeLoggingKeyValueBytesStoreTest.java index 9360daef272..7342c93abcc 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/ChangeLoggingKeyValueBytesStoreTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/ChangeLoggingKeyValueBytesStoreTest.java @@ -23,7 +23,7 @@ import org.apache.kafka.common.utils.LogContext; import org.apache.kafka.streams.KeyValue; import org.apache.kafka.streams.processor.internals.MockStreamsMetrics; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.NoOpRecordCollector; import org.apache.kafka.test.TestUtils; import org.junit.After; @@ -41,7 +41,7 @@ public class ChangeLoggingKeyValueBytesStoreTest { - private MockProcessorContext context; + private InternalMockProcessorContext context; private final InMemoryKeyValueStore<Bytes, byte[]> inner = new InMemoryKeyValueStore<>("kv", Serdes.Bytes(), Serdes.ByteArray()); private final ChangeLoggingKeyValueBytesStore store = new ChangeLoggingKeyValueBytesStore(inner); private final Map sent = new HashMap<>(); @@ -64,7 +64,7 @@ public void before() { sent.put(key, value); } }; - context = new MockProcessorContext(+ context = new InternalMockProcessorContext(TestUtils.tempDirectory(), Serdes.String(), Serdes.Long(), diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/MeteredWindowStoreTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/MeteredWindowStoreTest.java index 59c7ade8f97..4fd7f30a626 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/MeteredWindowStoreTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/MeteredWindowStoreTest.java @@ -27,7 +27,7 @@ import org.apache.kafka.streams.StreamsMetrics; import org.apache.kafka.streams.kstream.Windowed; import org.apache.kafka.streams.state.WindowStore; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.NoOpRecordCollector; import org.apache.kafka.test.TestUtils; import org.easymock.EasyMock; @@ -44,7 +44,7 @@ import static org.junit.Assert.assertTrue; public class MeteredWindowStoreTest { - private MockProcessorContext context; + private InternalMockProcessorContext context; @SuppressWarnings("unchecked") private final WindowStore<Bytes, byte[]> innerStoreMock = EasyMock.createNiceMock(WindowStore.class); private final MeteredWindowStore<String, String> store = new MeteredWindowStore<>(innerStoreMock, "scope", new MockTime(), Serdes.String(), new SerdeThatDoesntHandleNull()); @@ -98,7 +98,7 @@ public Sensor addSensor(String name, Sensor.RecordingLevel recordLevel, Sensor.. }; - context = new MockProcessorContext(+ context = new InternalMockProcessorContext(TestUtils.tempDirectory(), Serdes.String(), Serdes.Long(), diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBKeyValueStoreSupplierTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBKeyValueStoreSupplierTest.java index 66cc9add145..098c3262e34 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBKeyValueStoreSupplierTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBKeyValueStoreSupplierTest.java @@ -24,7 +24,7 @@ import org.apache.kafka.streams.StreamsMetrics; import org.apache.kafka.streams.processor.internals.MockStreamsMetrics; import org.apache.kafka.streams.state.KeyValueStore; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.NoOpRecordCollector; import org.apache.kafka.test.TestUtils; import org.junit.After; @@ -44,7 +44,7 @@ private static final String STORE_NAME = "name"; private final ThreadCache cache = new ThreadCache(new LogContext("test "), 1024, new MockStreamsMetrics(new Metrics())); - private final MockProcessorContext context = new MockProcessorContext(TestUtils.tempDirectory(), + private final InternalMockProcessorContext context = new InternalMockProcessorContext(TestUtils.tempDirectory(), Serdes.String(), Serdes.String(), new NoOpRecordCollector(), @@ -73,7 +73,7 @@ public void shouldCreateLoggingEnabledStoreWhenStoreLogged() { logged.add(new ProducerRecord<K, V>(topic, partition, timestamp, key, value)); } }; - final MockProcessorContext context = new MockProcessorContext(TestUtils.tempDirectory(), + final InternalMockProcessorContext context = new InternalMockProcessorContext(TestUtils.tempDirectory(), Serdes.String(), Serdes.String(), collector, @@ -100,7 +100,7 @@ public void shouldNotBeLoggingEnabledStoreWhenLoggingNotEnabled() { logged.add(new ProducerRecord<>(topic, partition, timestamp, key, value)); } }; - final MockProcessorContext context = new MockProcessorContext(TestUtils.tempDirectory(), + final InternalMockProcessorContext context = new InternalMockProcessorContext(TestUtils.tempDirectory(), Serdes.String(), Serdes.String(), collector, diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBSegmentedBytesStoreTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBSegmentedBytesStoreTest.java index e34d3cccc67..388a2fc47ba 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBSegmentedBytesStoreTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBSegmentedBytesStoreTest.java @@ -26,7 +26,7 @@ import org.apache.kafka.streams.kstream.internals.SessionWindow; import org.apache.kafka.streams.processor.internals.MockStreamsMetrics; import org.apache.kafka.streams.state.KeyValueIterator; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.NoOpRecordCollector; import org.apache.kafka.test.TestUtils; import org.junit.After; @@ -56,7 +56,7 @@ private final long retention = 60000L; private final int numSegments = 3; - private MockProcessorContext context; + private InternalMockProcessorContext context; private final String storeName = "bytes-store"; private RocksDBSegmentedBytesStore bytesStore; private File stateDir; @@ -71,7 +71,7 @@ public void before() { schema); stateDir = TestUtils.tempDirectory(); - context = new MockProcessorContext(+ context = new InternalMockProcessorContext(stateDir, Serdes.String(), Serdes.Long(), diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBSessionStoreSupplierTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBSessionStoreSupplierTest.java index 19bfdedb072..272e0b0f5e2 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBSessionStoreSupplierTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBSessionStoreSupplierTest.java @@ -26,7 +26,7 @@ import org.apache.kafka.streams.kstream.internals.SessionWindow; import org.apache.kafka.streams.processor.internals.MockStreamsMetrics; import org.apache.kafka.streams.state.SessionStore; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.NoOpRecordCollector; import org.apache.kafka.test.TestUtils; import org.junit.After; @@ -47,7 +47,7 @@ private static final String STORE_NAME = "name"; private final List<ProducerRecord> logged = new ArrayList<>(); private final ThreadCache cache = new ThreadCache(new LogContext("test "), 1024, new MockStreamsMetrics(new Metrics())); - private final MockProcessorContext context = new MockProcessorContext(TestUtils.tempDirectory(), + private final InternalMockProcessorContext context = new InternalMockProcessorContext(TestUtils.tempDirectory(), Serdes.String(), Serdes.String(), new NoOpRecordCollector() { diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBSessionStoreTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBSessionStoreTest.java index b25d72586ca..64953153045 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBSessionStoreTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBSessionStoreTest.java @@ -25,7 +25,7 @@ import org.apache.kafka.streams.processor.internals.MockStreamsMetrics; import org.apache.kafka.streams.state.KeyValueIterator; import org.apache.kafka.streams.state.SessionStore; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.NoOpRecordCollector; import org.apache.kafka.test.TestUtils; import org.junit.After; @@ -45,7 +45,7 @@ public class RocksDBSessionStoreTest { private SessionStore<String, Long> sessionStore; - private MockProcessorContext context; + private InternalMockProcessorContext context; @Before public void before() { @@ -59,7 +59,7 @@ public void before() { Serdes.String(), Serdes.Long()); - context = new MockProcessorContext(TestUtils.tempDirectory(), + context = new InternalMockProcessorContext(TestUtils.tempDirectory(), Serdes.String(), Serdes.Long(), new NoOpRecordCollector(), diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBStoreTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBStoreTest.java index a89dc60e665..a09d87dbad3 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBStoreTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBStoreTest.java @@ -31,7 +31,7 @@ import org.apache.kafka.streams.processor.internals.MockStreamsMetrics; import org.apache.kafka.streams.state.KeyValueIterator; import org.apache.kafka.streams.state.RocksDBConfigSetter; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.NoOpRecordCollector; import org.apache.kafka.test.TestUtils; import org.junit.After; @@ -62,14 +62,14 @@ private Serializer<String> stringSerializer = new

StringSerializer(); private Deserializer<String> stringDeserializer = new StringDeserializer(); private RocksDBStore rocksDBStore; - private MockProcessorContext context; + private InternalMockProcessorContext context; private File dir; @Before public void setUp() { rocksDBStore = new RocksDBStore("test"); dir = TestUtils.tempDirectory(); - context = new MockProcessorContext(dir, + context = new InternalMockProcessorContext(dir, Serdes.String(), Serdes.String(), new NoOpRecordCollector(), @@ -115,7 +115,7 @@ public void verifyRocksDbConfigSetterIsCalled() { configs.put(StreamsConfig.BOOTSTRAP_SERVERS_CONFIG, "test-server:9092"); configs.put(StreamsConfig.ROCKSDB_CONFIG_SETTER_CLASS_CONFIG, MockRocksDbConfigSetter.class); MockRocksDbConfigSetter.called = false; - rocksDBStore.openDB(new MockProcessorContext(tempDir, new StreamsConfig(configs))); + rocksDBStore.openDB(new InternalMockProcessorContext(tempDir, new StreamsConfig(configs))); assertTrue(MockRocksDbConfigSetter.called); } @@ -123,7 +123,7 @@ public void verifyRocksDbConfigSetterIsCalled() { @Test(expected = ProcessorStateException.class) public void shouldThrowProcessorStateExceptionOnOpeningReadOnlyDir() throws IOException { final File tmpDir = TestUtils.tempDirectory(); - MockProcessorContext tmpContext = new MockProcessorContext(tmpDir, + InternalMockProcessorContext tmpContext = new InternalMockProcessorContext(tmpDir, Serdes.String(), Serdes.Long(), new NoOpRecordCollector(), diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBWindowStoreSupplierTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBWindowStoreSupplierTest.java index bca6949c6b3..a6ccfdf9337 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBWindowStoreSupplierTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBWindowStoreSupplierTest.java @@ -24,7 +24,7 @@ import org.apache.kafka.streams.StreamsMetrics; import org.apache.kafka.streams.processor.internals.MockStreamsMetrics; import org.apache.kafka.streams.state.WindowStore; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.NoOpRecordCollector; import org.apache.kafka.test.TestUtils; import org.junit.After; @@ -45,7 +45,7 @@ private static final String STORE_NAME = "name"; private WindowStore<String, String> store; private final ThreadCache cache = new ThreadCache(new LogContext("test "), 1024, new MockStreamsMetrics(new Metrics())); - private final MockProcessorContext context = new MockProcessorContext(TestUtils.tempDirectory(), + private final InternalMockProcessorContext context = new InternalMockProcessorContext(TestUtils.tempDirectory(), Serdes.String(), Serdes.String(), new NoOpRecordCollector(), @@ -75,7 +75,7 @@ public void shouldCreateLoggingEnabledStoreWhenWindowStoreLogged() { logged.add(new ProducerRecord<K, V>(topic, partition, timestamp, key, value)); } }; - final MockProcessorContext context = new MockProcessorContext(TestUtils.tempDirectory(), + final InternalMockProcessorContext context = new InternalMockProcessorContext(TestUtils.tempDirectory(), Serdes.String(), Serdes.String(), collector, @@ -102,7 +102,7 @@ public void shouldNotBeLoggingEnabledStoreWhenLogginNotEnabled() { logged.add(new ProducerRecord<K, V>(topic, partition, timestamp, key, value)); } }; - final MockProcessorContext context = new MockProcessorContext(TestUtils.tempDirectory(), + final InternalMockProcessorContext context = new InternalMockProcessorContext(TestUtils.tempDirectory(), Serdes.String(), Serdes.String(), collector, diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBWindowStoreTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBWindowStoreTest.java index c745e702b09..b3a60a965d8 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBWindowStoreTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/RocksDBWindowStoreTest.java @@ -37,7 +37,7 @@ import org.apache.kafka.streams.state.Stores; import org.apache.kafka.streams.state.WindowStore; import org.apache.kafka.streams.state.WindowStoreIterator; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.StreamsTestUtils; import org.apache.kafka.test.TestUtils; import org.junit.After; @@ -94,7 +94,7 @@ }; private final File baseDir = TestUtils.tempDirectory("test"); - private final MockProcessorContext context = new MockProcessorContext(baseDir, Serdes.ByteArray(), Serdes.ByteArray(), recordCollector, cache); + private final InternalMockProcessorContext context = new InternalMockProcessorContext(baseDir, Serdes.ByteArray(), Serdes.ByteArray(), recordCollector, cache); private WindowStore<Integer, String> windowStore; private WindowStore<Integer, String> createWindowStore(final ProcessorContext context, final boolean retainDuplicates) { @@ -842,7 +842,7 @@ public void shouldFetchAndIterateOverExactBinaryKeys() { assertThat(toList(windowStore.fetch(key3, 0, Long.MAX_VALUE)), equalTo(expectedKey3)); } - private void putFirstBatch(final WindowStore<Integer, String> store, final long startTime, final MockProcessorContext context) { + private void putFirstBatch(final WindowStore<Integer, String> store, final long startTime, final InternalMockProcessorContext context) { context.setRecordContext(createRecordContext(startTime)); store.put(0, "zero"); context.setRecordContext(createRecordContext(startTime + 1L)); @@ -855,7 +855,7 @@ private void putFirstBatch(final WindowStore<Integer, String> store, final long store.put(5, "five"); } - private void putSecondBatch(final WindowStore<Integer, String> store, final long startTime, MockProcessorContext context) { + private void putSecondBatch(final WindowStore<Integer, String> store, final long startTime, InternalMockProcessorContext context) { context.setRecordContext(createRecordContext(startTime + 3L)); store.put(2, "two+1"); context.setRecordContext(createRecordContext(startTime + 4L)); diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/SegmentIteratorTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/SegmentIteratorTest.java index 9c150c58682..d61218eb9b5 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/SegmentIteratorTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/SegmentIteratorTest.java @@ -23,7 +23,7 @@ import org.apache.kafka.streams.KeyValue; import org.apache.kafka.streams.processor.internals.MockStreamsMetrics; import org.apache.kafka.streams.state.KeyValueIterator; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.NoOpRecordCollector; import org.apache.kafka.test.TestUtils; import org.junit.After; @@ -48,12 +48,12 @@ public boolean hasNext(final KeyValueIterator iterator) { } }; - private MockProcessorContext context; + private InternalMockProcessorContext context; private SegmentIterator iterator = null; @Before public void before() { - context = new MockProcessorContext(+ context = new InternalMockProcessorContext(TestUtils.tempDirectory(), Serdes.String(), Serdes.String(), diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/SegmentsTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/SegmentsTest.java index deb26f735c9..ec59a008112 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/SegmentsTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/SegmentsTest.java @@ -20,7 +20,7 @@ import org.apache.kafka.common.serialization.Serdes; import org.apache.kafka.common.utils.LogContext; import org.apache.kafka.streams.processor.internals.MockStreamsMetrics; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.NoOpRecordCollector; import org.apache.kafka.test.TestUtils; import org.junit.After; @@ -44,7 +44,7 @@ public class SegmentsTest { private static final int NUM_SEGMENTS = 5; - private MockProcessorContext context; + private InternalMockProcessorContext context; private Segments segments; private long segmentInterval; private File stateDirectory; @@ -54,7 +54,7 @@ @Before public void createContext() { stateDirectory = TestUtils.tempDirectory(); - context = new MockProcessorContext(stateDirectory, + context = new InternalMockProcessorContext(stateDirectory, Serdes.String(), Serdes.Long(), new NoOpRecordCollector(), diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/StateStoreTestUtils.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/StateStoreTestUtils.java index d30372fd80b..b1818c21ef1 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/StateStoreTestUtils.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/StateStoreTestUtils.java @@ -21,7 +21,7 @@ import org.apache.kafka.streams.processor.internals.ProcessorStateManager; import org.apache.kafka.streams.state.KeyValueStore; import org.apache.kafka.streams.state.StateSerdes; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.apache.kafka.test.NoOpRecordCollector; import java.util.Collections; @@ -42,7 +42,7 @@ final StateStore stateStore = supplier.get(); stateStore.init(- new MockProcessorContext(+ new InternalMockProcessorContext(StateSerdes.withBuiltinTypes(ProcessorStateManager.storeChangelogTopic(applicationId, name), keyType, diff --git a/streams/src/test/java/org/apache/kafka/streams/state/internals/StoreChangeLoggerTest.java b/streams/src/test/java/org/apache/kafka/streams/state/internals/StoreChangeLoggerTest.java index 32b56bbecfb..c62b09b640e 100644 --- a/streams/src/test/java/org/apache/kafka/streams/state/internals/StoreChangeLoggerTest.java +++ b/streams/src/test/java/org/apache/kafka/streams/state/internals/StoreChangeLoggerTest.java @@ -23,7 +23,7 @@ import org.apache.kafka.streams.processor.StreamPartitioner; import org.apache.kafka.streams.processor.internals.RecordCollectorImpl; import org.apache.kafka.streams.state.StateSerdes; -import org.apache.kafka.test.MockProcessorContext; +import org.apache.kafka.test.InternalMockProcessorContext; import org.junit.After; import org.junit.Test; @@ -39,7 +39,7 @@ private final Map<Integer, String> logged = new HashMap<>(); - private final MockProcessorContext context = new MockProcessorContext(StateSerdes.withBuiltinTypes(topic, Integer.class, String.class), + private final InternalMockProcessorContext context = new InternalMockProcessorContext(StateSerdes.withBuiltinTypes(topic, Integer.class, String.class), new RecordCollectorImpl(null, "StoreChangeLoggerTest", new LogContext("StoreChangeLoggerTest "), new DefaultProductionExceptionHandler()) { @Override public <K1, V1> void send(final String topic, diff --git a/streams/src/test/java/org/apache/kafka/test/MockProcessorContext.java b/streams/src/test/java/org/apache/kafka/test/InternalMockProcessorContext.java similarity index 87% rename from streams/src/test/java/org/apache/kafka/test/MockProcessorContext.java rename to streams/src/test/java/org/apache/kafka/test/InternalMockProcessorContext.java index 6b0cb66bcb7..74bb5d14062 100644 --- a/streams/src/test/java/org/apache/kafka/test/MockProcessorContext.java +++ b/streams/src/test/java/org/apache/kafka/test/InternalMockProcessorContext.java @@ -48,7 +48,7 @@ import java.util.List; import java.util.Map; -public class MockProcessorContext extends AbstractProcessorContext implements RecordCollector.Supplier { +public class InternalMockProcessorContext extends AbstractProcessorContext implements RecordCollector.Supplier { private final File stateDir; private final Metrics metrics; @@ -61,19 +61,19 @@ private Serde<?> valSerde; private long timestamp = -1L; - public MockProcessorContext(final File stateDir, - final StreamsConfig config) { + public InternalMockProcessorContext(final File stateDir, + final StreamsConfig config) { this(stateDir, null, null, new Metrics(), config, null, null); } - public MockProcessorContext(final StateSerdes<?, ?> serdes, - final RecordCollector collector) { + public InternalMockProcessorContext(final StateSerdes<?, ?> serdes, + final RecordCollector collector) { this(null, serdes.keySerde(), serdes.valueSerde(), collector, null); } - public MockProcessorContext(final StateSerdes<?, ?> serdes, - final RecordCollector collector, - final Metrics metrics) { + public InternalMockProcessorContext(final StateSerdes<?, ?> serdes, + final RecordCollector collector, + final Metrics metrics) { this(null, serdes.keySerde(), serdes.valueSerde(), metrics, new StreamsConfig(StreamsTestUtils.minimalStreamsConfig()), new RecordCollector.Supplier() { @Override public RecordCollector recordCollector() { @@ -82,11 +82,11 @@ public RecordCollector recordCollector() { }, null); } - public MockProcessorContext(final File stateDir, - final Serde<?> keySerde, - final Serde<?> valSerde, - final RecordCollector collector, - final ThreadCache cache) { + public InternalMockProcessorContext(final File stateDir, + final Serde<?> keySerde, + final Serde<?> valSerde, + final RecordCollector collector, + final ThreadCache cache) { this(stateDir, keySerde, valSerde, new Metrics(), new StreamsConfig(StreamsTestUtils.minimalStreamsConfig()), new RecordCollector.Supplier() { @Override public RecordCollector recordCollector() { @@ -95,13 +95,13 @@ public RecordCollector recordCollector() { }, cache); } - private MockProcessorContext(final File stateDir, - final Serde<?> keySerde, - final Serde<?> valSerde, - final Metrics metrics, - final StreamsConfig config, - final RecordCollector.Supplier collectorSupplier, - final ThreadCache cache) { + private InternalMockProcessorContext(final File stateDir, + final Serde<?> keySerde, + final Serde<?> valSerde, + final Metrics metrics, + final StreamsConfig config, + final RecordCollector.Supplier collectorSupplier, + final ThreadCache cache) { super(new TaskId(0, 0), config, new MockStreamsMetrics(metrics), diff --git a/streams/src/test/java/org/apache/kafka/test/KStreamTestDriver.java b/streams/src/test/java/org/apache/kafka/test/KStreamTestDriver.java index 3a9ed751e9c..39183d942ea 100644 --- a/streams/src/test/java/org/apache/kafka/test/KStreamTestDriver.java +++ b/streams/src/test/java/org/apache/kafka/test/KStreamTestDriver.java @@ -48,7 +48,7 @@ private static final long DEFAULT_CACHE_SIZE_BYTES = 1 * 1024 * 1024L; private ProcessorTopology topology; - private MockProcessorContext context; + private InternalMockProcessorContext context; private ProcessorTopology globalTopology; private final LogContext logContext = new LogContext("testCache "); @@ -85,7 +85,7 @@ public void setUp(final KStreamBuilder builder, topology = builder.build(null); globalTopology = builder.buildGlobalStateTopology(); final ThreadCache cache = new ThreadCache(logContext, cacheSize, new MockStreamsMetrics(new Metrics())); - context = new MockProcessorContext(stateDir, keySerde, valSerde, new MockRecordCollector(), cache); + context = new InternalMockProcessorContext(stateDir, keySerde, valSerde, new MockRecordCollector(), cache); context.setRecordContext(new ProcessorRecordContext(0, 0, 0, "topic")); // init global topology first as it will add stores to the // store map that are required for joins etc. @@ -126,7 +126,7 @@ public void setUp(final StreamsBuilder builder, globalTopology = internalTopologyBuilder.buildGlobalStateTopology(); final ThreadCache cache = new ThreadCache(logContext, cacheSize, new MockStreamsMetrics(new Metrics())); - context = new MockProcessorContext(stateDir, keySerde, valSerde, new MockRecordCollector(), cache); + context = new InternalMockProcessorContext(stateDir, keySerde, valSerde, new MockRecordCollector(), cache); context.setRecordContext(new ProcessorRecordContext(0, 0, 0, "topic")); // init global topology first as it will add stores to the diff --git a/streams/test-utils/src/main/java/org/apache/kafka/streams/processor/MockProcessorContext.java b/streams/test-utils/src/main/java/org/apache/kafka/streams/processor/MockProcessorContext.java new file mode 100644 index 00000000000..03f871a7284 --- /dev/null +++ b/streams/test-utils/src/main/java/org/apache/kafka/streams/processor/MockProcessorContext.java @@ -0,0 +1,478 @@ +/* + * Licensed to the Apache Software Foundation (ASF) under one or more + * contributor license agreements. See the NOTICE file distributed with + * this work for additional information regarding copyright ownership. + * The ASF licenses this file to You under the Apache License, Version 2.0 + * (the "License"); you may not use this file except in compliance with + * the License. You may obtain a copy of the License at + * + * http://www.apache.org/licenses/LICENSE-2.0 + * + * Unless required by applicable law or agreed to in writing, software + * distributed under the License is distributed on an "AS IS" BASIS, + * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. + * See the License for the specific language governing permissions and + * limitations under the License. + */ +package org.apache.kafka.streams.processor; + +import org.apache.kafka.common.annotation.InterfaceStability; +import org.apache.kafka.common.metrics.Metrics; +import org.apache.kafka.common.serialization.Serde; +import org.apache.kafka.streams.KeyValue; +import org.apache.kafka.streams.StreamsConfig; +import org.apache.kafka.streams.StreamsMetrics;

```
+import org.apache.kafka.streams.Topology; +import org.apache.kafka.streams.TopologyTestDriver; +import
org.apache.kafka.streams.kstream.Transformer; +import org.apache.kafka.streams.kstream.ValueTransformer; +import
org.apache.kafka.streams.processor.internals.RecordCollector; +import org.apache.kafka.streams.processor.internals.StreamsMetricsImpl;
+import org.apache.kafka.streams.state.internals.InMemoryKeyValueStore; + +import java.io.File; +import java.util.HashMap; +import
java.util.LinkedList; +import java.util.List; +import java.util.Map; +import java.util.Properties; + +/** + * {@link
MockProcessorContext} is a mock of {@link ProcessorContext} for users to test their {@link Processor}, + * {@link Transformer}, and
{@link ValueTransformer} implementations. + * <p> + * The tests for this class (org.apache.kafka.streams.MockProcessorContextTest)
include several behavioral + * tests that serve as example usage. + * <p> + * Note that this class does not take any automated actions (such
as firing scheduled punctuators). + * It simply captures any data it witnessess. + * If you require more automated tests, we recommend
wrapping your {@link Processor} in a minimal source-processor-sink + * {@link Topology} and using the {@link TopologyTestDriver}.
+ */ +@InterfaceStability.Evolving +public class MockProcessorContext implements ProcessorContext, RecordCollector.Supplier { + //
Immutable fields ================================================ + private final StreamsMetricsImpl metrics; +
private final TaskId taskId; + private final StreamsConfig config; + private final File stateDir; + + // settable record metadata
================================================ + private String topic; + private Integer partition; + private Long
offset; + private Long timestamp; + + // mocks ================================================ + private final
Map<String, StateStore> stateStores = new HashMap<>(); + private final List<CapturedPunctuator> punctuators = new LinkedList<>(); +
private final List<CapturedForward> capturedForwards = new LinkedList<>(); + private boolean committed = false; + + /** + * {@link
CapturedPunctuator} holds captured punctuators, along with their scheduling information. + */ + public static class CapturedPunctuator {
+ private final long intervalMs; + private final PunctuationType type; + private final Punctuator punctuator; + private boolean cancelled =
false; + + private CapturedPunctuator(final long intervalMs, final PunctuationType type, final Punctuator punctuator) { + this.intervalMs =
intervalMs; + this.type = type; + this.punctuator = punctuator; + } + + public long getIntervalMs() { + return intervalMs; + } + + public
PunctuationType getType() { + return type; + } + + public Punctuator getPunctuator() { + return punctuator; + } + + public void cancel() {
+ this.cancelled = true; + } + + public boolean cancelled() { + return cancelled; + } + } + + + public static class CapturedForward { +
private final String childName; + private final long timestamp; + private final KeyValue keyValue; + + private CapturedForward(final To
to, final KeyValue keyValue) { + if (keyValue == null) throw new IllegalArgumentException(); + + this.childName = to.childName; +
this.timestamp = to.timestamp; + this.keyValue = keyValue; + } + + /** + * The child this data was forwarded to. + * + * @return The
child name, or {@code null} if it was broadcasted. + */ + public String childName() { + return childName; + } + + /** + * The timestamp
attached to the forwarded record. + * + * @return A timestamp, or {@code -1} if none was forwarded. + */ + public long timestamp() { +
return timestamp; + } + + /** + * The data forwarded. + * + * @return A key/value pair. Not null. + */ + public KeyValue keyValue() { +
return keyValue; + } + } + + // contructors ================================================ + + /** + * Create a {@link
MockProcessorContext} with dummy {@code config} and {@code taskId} and {@code null} {@code stateDir}. + * Most unit tests
using this mock won't need to know the taskId, + * and most unit tests should be able to get by with the + * {@link
InMemoryKeyValueStore}, so the stateDir won't matter. + */ + public MockProcessorContext() { + //noinspection
DoubleBraceInitialization + this( + new Properties() { + { + put(StreamsConfig.APPLICATION_ID_CONFIG, ""); +
put(StreamsConfig.BOOTSTRAP_SERVERS_CONFIG, ""); + } + }, + new TaskId(0, 0), + null); + } + + /** + * Create a {@link
MockProcessorContext} with dummy {@code taskId} and {@code null} {@code stateDir}. + * Most unit tests using this mock won't
need to know the taskId, + * and most unit tests should be able to get by with the + * {@link InMemoryKeyValueStore}, so the stateDir
won't matter. + * + * @param config a Properties object, used to configure the context and the processor. + */ + public
MockProcessorContext(final Properties config) { + this(config, new TaskId(0, 0), null); + } + + /** + * Create a {@link
MockProcessorContext} with a specified taskId and null stateDir. + * + * @param config a {@link Properties} object, used to configure
the context and the processor. + * @param taskId a {@link TaskId}, which the context makes available via {@link
MockProcessorContext#taskId()}. + * @param stateDir a {@link File}, which the context makes available viw {@link
MockProcessorContext#stateDir()}. + */ + public MockProcessorContext(final Properties config, final TaskId taskId, final File stateDir) {
+ final StreamsConfig streamsConfig = new StreamsConfig(config); + this.taskId = taskId; + this.config = streamsConfig; + this.stateDir =
stateDir; + this.metrics = new StreamsMetricsImpl(new Metrics(), "mock-processor-context", new HashMap<String, String>()); + } + +
@Override + public String applicationId() { + return config.getString(StreamsConfig.APPLICATION_ID_CONFIG); + } + + @Override
+ public TaskId taskId() { + return taskId; + } + + @Override + public Map<String, Object> appConfigs() { + final Map<String, Object>
combined = new HashMap<>(); + combined.putAll(config.originals()); + combined.putAll(config.values()); + return combined; + } + +
@Override + public Map<String, Object> appConfigsWithPrefix(final String prefix) { + return config.originalsWithPrefix(prefix); + } + +
@Override + public Serde<?> keySerde() { + return config.defaultKeySerde(); + } + + @Override + public Serde<?> valueSerde() { +
return config.defaultValueSerde(); + } + + @Override + public File stateDir() { + return stateDir; + } + + @Override + public
StreamsMetrics metrics() { + return metrics; + } + + // settable record metadata
================================================ + + /** + * The context exposes these metadata for use in the
processor. Normally, they are set by the Kafka Streams framework, + * but for the purpose of driving unit tests, you can set them directly.
+ * + * @param topic A topic name + * @param partition A partition number + * @param offset A record offset + * @param timestamp
A record timestamp + */ + public void setRecordMetadata(final String topic, final int partition, final long offset, final long timestamp) { +
this.topic = topic; + this.partition = partition; + this.offset = offset; + this.timestamp = timestamp; + } + + + /** + * The context exposes
this metadata for use in the processor. Normally, they are set by the Kafka Streams framework, + * but for the purpose of driving unit tests,
you can set it directly. Setting this attribute doesn't affect the others. + * + * @param topic A topic name + */ + public void setTopic(final
String topic) { + this.topic = topic; + } + + /** + * The context exposes this metadata for use in the processor. Normally, they are set by the
Kafka Streams framework, + * but for the purpose of driving unit tests, you can set it directly. Setting this attribute doesn't affect the
others. + * + * @param partition A partition number + */ + public void setPartition(final int partition) { + this.partition = partition; + } + +
+ /** + * The context exposes this metadata for use in the processor. Normally, they are set by the Kafka Streams framework, + * but for
the purpose of driving unit tests, you can set it directly. Setting this attribute doesn't affect the others. + * + * @param offset A record
offset + */ + public void setOffset(final long offset) { + this.offset = offset; + } + + + /** + * The context exposes this metadata for use in
the processor. Normally, they are set by the Kafka Streams framework, + * but for the purpose of driving unit tests, you can set it directly.
Setting this attribute doesn't affect the others. + * + * @param timestamp A record timestamp + */ + public void setTimestamp(final long
timestamp) { + this.timestamp = timestamp; + } + + @Override + public String topic() { + if (topic == null) { + throw new
IllegalStateException("Topic must be set before use via setRecordMetadata() or setTopic()."); + } + return topic; + } + + @Override +
public int partition() { + if (partition == null) { + throw new IllegalStateException("Partition must be set before use via
setRecordMetadata() or setPartition()."); + } + return partition; + } + + @Override + public long offset() { + if (offset == null) { + throw
new IllegalStateException("Offset must be set before use via setRecordMetadata() or setOffset()."); + } + return offset; + } + + @Override
+ public long timestamp() { + if (timestamp == null) { + throw new IllegalStateException("Timestamp must be set before use via
setRecordMetadata() or setTimestamp()."); + } + return timestamp; + } + + // mocks
================================================ + + + @Override + public void register(final StateStore store, + final
boolean loggingEnabledIsDeprecatedAndIgnored, + final StateRestoreCallback stateRestoreCallbackIsIgnoredInMock) { +
```

stateStores.put(store.name(), store); + } + + @Override + public StateStore getStateStore(final String name) { + return stateStores.get(name); + } + + @Override + public Cancellable schedule(final long intervalMs, final PunctuationType type, final Punctuator callback) { + final CapturedPunctuator capturedPunctuator = new CapturedPunctuator(intervalMs, type, callback); + + punctuators.add(capturedPunctuator); + + return new Cancellable() { + @Override + public void cancel() { + capturedPunctuator.cancel(); + } + }; + } + + @Override + public void schedule(final long interval) { + throw new UnsupportedOperationException(+ "schedule() is deprecated and not supported in Mock. " + + "Use schedule(final long intervalMs, final PunctuationType type, final Punctuator callback) instead." +); + } + + /** + * Get the punctuators scheduled so far. The returned list is not affected by subsequent calls to {@code schedule(...)}. + * + * @return A list of captured punctuators. + */ + public List<CapturedPunctuator> scheduledPunctuators() { + final LinkedList<CapturedPunctuator> capturedPunctuators = new LinkedList<>(); + capturedPunctuators.addAll(punctuators); + return capturedPunctuators; + } + + @Override + public <K, V> void forward(final K key, final V value) { + //noinspection unchecked + capturedForwards.add(new CapturedForward(To.all(), new KeyValue(key, value))); + } + + @Override + public <K, V> void forward(final K key, final V value, final To to) { + //noinspection unchecked + capturedForwards.add(new CapturedForward(to, new KeyValue(key, value))); + } + + @Override + public <K, V> void forward(final K key, final V value, final int childIndex) { + throw new UnsupportedOperationException(+ "Forwarding to a child by index is deprecated. " + + "Please transition processors to forward using a 'To' object instead." +); + } + + @Override + public <K, V> void forward(final K key, final V value, final String childName) { + throw new UnsupportedOperationException(+ "Forwarding to a child by name is deprecated. " + + "Please transition processors to forward using 'To.child(childName)' instead." +); + } + + /** + * Get all the forwarded data this context has observed. The returned list will not be + * affected by subsequent interactions with the context. The data in the list is in the same order as the calls to + * {@code forward(...)}. + * + * @return A list of key/value pairs that were previously passed to the context. + */ + public List<CapturedForward> forwarded() { + final LinkedList<CapturedForward> result = new LinkedList<>(); + result.addAll(capturedForwards); + return result; + } + + /** + * Get all the forwarded data this context has observed for a specific child by name. + * The returned list will not be affected by subsequent interactions with the context. + * The data in the list is in the same order as the calls to {@code forward(...)}. + * + * @param childName The child name to retrieve forwards for + * @return A list of key/value pairs that were previously passed to the context. + */ + public List<CapturedForward> forwarded(final String childName) { + final LinkedList<CapturedForward> result = new LinkedList<>(); + for (final CapturedForward capture : capturedForwards) { + if (capture.childName() == null || capture.childName().equals(childName)) { + result.add(capture); + } + } + return result; + } + + /** + * Clear the captured forwarded data. + */ + public void resetForwards() { + capturedForwards.clear(); + } + + @Override + public void commit() { + committed = true; + } + + /** + * Whether {@link ProcessorContext#commit()} has been called in this context. + * + * @return {@code true} iff {@link ProcessorContext#commit()} has been called in this context since construction or reset. + */ + public boolean committed() { + return committed; + } + + /** + * Reset the commit capture to {@code false} (whether or not it was previously {@code true}). + */ + public void resetCommit() { + committed = false; + } + + @Override + public RecordCollector recordCollector() { + // This interface is assumed by state stores that add change-logging. + // Rather than risk a mysterious ClassCastException during unit tests, throw an explanatory exception. + + throw new UnsupportedOperationException("MockProcessorContext does not provide record collection. " + + "For processor unit tests, use an in-memory state store with change-logging disabled. " + + "Alternatively, use the TopologyTestDriver for testing processor/store/topology integration."); + } +} diff --git a/streams/test-utils/src/test/java/org/apache/kafka/streams/MockProcessorContextTest.java b/streams/test-utils/src/test/java/org/apache/kafka/streams/MockProcessorContextTest.java new file mode 100644 index 00000000000..8c5ec4604fa --- /dev/null +++ b/streams/test-utils/src/test/java/org/apache/kafka/streams/MockProcessorContextTest.java @@ -0,0 +1,406 @@ +/* + * Licensed to the Apache Software Foundation (ASF) under one or more + * contributor license agreements. See the NOTICE file distributed with + * this work for additional information regarding copyright ownership. + * The ASF licenses this file to You under the Apache License, Version 2.0 + * (the "License"); you may not use this file except in compliance with + * the License. You may obtain a copy of the License at + * + * http://www.apache.org/licenses/LICENSE-2.0 + * + * Unless required by applicable law or agreed to in writing, software + * distributed under the License is distributed on an "AS IS" BASIS, + * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. + * See the License for the specific language governing permissions and + * limitations under the License. + */ +package org.apache.kafka.streams; + +import org.apache.kafka.common.serialization.Serdes; +import org.apache.kafka.streams.processor.AbstractProcessor; +import org.apache.kafka.streams.processor.MockProcessorContext; +import org.apache.kafka.streams.processor.MockProcessorContext.CapturedForward; +import org.apache.kafka.streams.processor.Processor; +import org.apache.kafka.streams.processor.ProcessorContext; +import org.apache.kafka.streams.processor.PunctuationType; +import org.apache.kafka.streams.processor.Punctuator; +import org.apache.kafka.streams.processor.TaskId; +import org.apache.kafka.streams.processor.To; +import org.apache.kafka.streams.state.KeyValueStore; +import org.apache.kafka.streams.state.internals.InMemoryKeyValueStore; +import org.junit.Test; + +import java.io.File; +import java.util.Iterator; +import java.util.Properties; + +import static org.junit.Assert.assertEquals; +import static org.junit.Assert.assertFalse; +import static org.junit.Assert.assertTrue; +import static org.junit.Assert.fail; + +public class MockProcessorContextTest { + @Test + public void shouldCaptureOutputRecords() { + final AbstractProcessor<String, Long> processor = new AbstractProcessor<String, Long>() { + @Override + public void process(final String key, final Long value) { + context().forward(key + value, key.length() + value); + } + }; + + final MockProcessorContext context = new MockProcessorContext(); + processor.init(context); + + processor.process("foo", 5L); + processor.process("barbaz", 50L); + + final Iterator<CapturedForward> forwarded = context.forwarded().iterator(); + assertEquals(new KeyValue<>("foo5", 8L), forwarded.next().keyValue()); + assertEquals(new KeyValue<>("barbaz50", 56L), forwarded.next().keyValue()); + assertFalse(forwarded.hasNext()); + + context.resetForwards(); + + assertEquals(0, context.forwarded().size()); + } + + @Test + public void shouldCaptureOutputRecordsUsingTo() { + final AbstractProcessor<String, Long> processor = new AbstractProcessor<String, Long>() { + @Override + public void process(final String key, final Long value) { + context().forward(key + value, key.length() + value, To.all()); + } + }; + + final MockProcessorContext context = new MockProcessorContext(); + + processor.init(context); + + processor.process("foo", 5L); + processor.process("barbaz", 50L); + + final Iterator<CapturedForward> forwarded = context.forwarded().iterator(); + assertEquals(new KeyValue<>("foo5", 8L), forwarded.next().keyValue()); + assertEquals(new KeyValue<>("barbaz50", 56L), forwarded.next().keyValue()); + assertFalse(forwarded.hasNext()); + + context.resetForwards(); + + assertEquals(0, context.forwarded().size()); + } + + @Test + public void shouldCaptureRecordsOutputToChildByName() { + final AbstractProcessor<String, Long> processor = new AbstractProcessor<String, Long>() { + private int count = 0; + + @Override + public void process(final String key, final Long value) { + if (count == 0) { + context().forward("start", -1L, To.all()); // broadcast + } + final To toChild = count % 2 == 0 ? To.child("george") : To.child("pete"); + context().forward(key + value, key.length() + value, toChild); + count++; + } + }; + + final MockProcessorContext context = new MockProcessorContext(); + + processor.init(context); + + processor.process("foo", 5L); + processor.process("barbaz", 50L); + + { + final Iterator<CapturedForward> forwarded = context.forwarded().iterator(); + + final CapturedForward forward1 = forwarded.next(); + assertEquals(new KeyValue<>("start", -1L), forward1.keyValue()); + assertEquals(null, forward1.childName()); + + final CapturedForward forward2 = forwarded.next(); + assertEquals(new KeyValue<>("foo5", 8L), forward2.keyValue()); + assertEquals("george", forward2.childName()); + + final CapturedForward forward3 = forwarded.next(); + assertEquals(new KeyValue<>("barbaz50", 56L), forward3.keyValue()); + assertEquals("pete", forward3.childName()); + + assertFalse(forwarded.hasNext()); + } + + { + final Iterator<CapturedForward> forwarded = context.forwarded("george").iterator(); + assertEquals(new KeyValue<>("start", -1L), forwarded.next().keyValue()); +

```
assertEquals(new KeyValue<>("foo5", 8L), forwarded.next().keyValue()); + assertFalse(forwarded.hasNext()); + } + + { + final
Iterator<CapturedForward> forwarded = context.forwarded("pete").iterator(); + assertEquals(new KeyValue<>("start", -1L),
forwarded.next().keyValue()); + assertEquals(new KeyValue<>("barbaz50", 56L), forwarded.next().keyValue()); +
assertFalse(forwarded.hasNext()); + } + + { + final Iterator<CapturedForward> forwarded = context.forwarded("steve").iterator(); +
assertEquals(new KeyValue<>("start", -1L), forwarded.next().keyValue()); + assertFalse(forwarded.hasNext()); + } + } + + @Test +
public void shouldThrowIfForwardedWithDeprecatedChildIndex() { + final AbstractProcessor<String, Long> processor = new
AbstractProcessor<String, Long>() { + @Override + public void process(final String key, final Long value) { + //noinspection deprecation
+ context().forward(key, value, 0); + } + }; + + final MockProcessorContext context = new MockProcessorContext(); + +
processor.init(context); + + try { + processor.process("foo", 5L); + fail("Should have thrown an UnsupportedOperationException."); + }
catch (final UnsupportedOperationException expected) { + // expected + } + } + + @Test + public void
shouldThrowIfForwardedWithDeprecatedChildName() { + final AbstractProcessor<String, Long> processor = new
AbstractProcessor<String, Long>() { + @Override + public void process(final String key, final Long value) { + //noinspection deprecation
+ context().forward(key, value, "child1"); + } + }; + + final MockProcessorContext context = new MockProcessorContext(); + +
processor.init(context); + + try { + processor.process("foo", 5L); + fail("Should have thrown an UnsupportedOperationException."); + }
catch (final UnsupportedOperationException expected) { + // expected + } + } + + @Test + public void
shouldCaptureCommitsAndAllowReset() { + final AbstractProcessor<String, Long> processor = new AbstractProcessor<String, Long>()
{ + private int count = 0; + + @Override + public void process(final String key, final Long value) { + if (++count > 2) context().commit();
+ } + }; + + final MockProcessorContext context = new MockProcessorContext(); + + processor.init(context); + + processor.process("foo",
5L); + processor.process("barbaz", 50L); + + assertFalse(context.committed()); + + processor.process("foobar", 500L); + +
assertTrue(context.committed()); + + context.resetCommit(); + + assertFalse(context.committed()); + } + + @Test + public void
shouldStoreAndReturnStateStores() { + final AbstractProcessor<String, Long> processor = new AbstractProcessor<String, Long>() { +
@Override + public void process(final String key, final Long value) { + //noinspection unchecked + final KeyValueStore<String, Long>
stateStore = (KeyValueStore<String, Long>) context().getStateStore("my-state"); + stateStore.put(key, (stateStore.get(key) == null ? 0 :
stateStore.get(key)) + value); + stateStore.put("all", (stateStore.get("all") == null ? 0 : stateStore.get("all")) + value); + } + }; + + final
MockProcessorContext context = new MockProcessorContext(); + final KeyValueStore<String, Long> store = new
InMemoryKeyValueStore<>("my-state", Serdes.String(), Serdes.Long()); + context.register(store, false, null); + + store.init(context, store);
+ processor.init(context); + + processor.process("foo", 5L); + processor.process("bar", 50L); + + assertEquals(5L, (long) store.get("foo"));
+ assertEquals(50L, (long) store.get("bar")); + assertEquals(55L, (long) store.get("all")); + } + + @Test + public void
shouldCaptureApplicationAndRecordMetadata() { + final Properties config = new Properties(); +
config.put(StreamsConfig.APPLICATION_ID_CONFIG, "testMetadata"); +
config.put(StreamsConfig.BOOTSTRAP_SERVERS_CONFIG, ""); + + final AbstractProcessor<String, Object> processor = new
AbstractProcessor<String, Object>() { + @Override + public void process(final String key, final Object value) { +
context().forward("appId", context().applicationId()); + context().forward("taskId", context().taskId()); + + context().forward("topic",
context().topic()); + context().forward("partition", context().partition()); + context().forward("offset", context().offset()); +
context().forward("timestamp", context().timestamp()); + + context().forward("key", key); + context().forward("value", value); + } + }; +
+ final MockProcessorContext context = new MockProcessorContext(config); + processor.init(context); + + try { +
processor.process("foo", 5L); + fail("Should have thrown an exception."); + } catch (final IllegalStateException expected) { + // expected,
since the record metadata isn't initialized + } + + context.resetForwards(); + context.setRecordMetadata("t1", 0, 0L, 0L); + + { +
processor.process("foo", 5L); + final Iterator<CapturedForward> forwarded = context.forwarded().iterator(); + assertEquals(new
KeyValue<>("appId", "testMetadata"), forwarded.next().keyValue()); + assertEquals(new KeyValue<>("taskId", new TaskId(0, 0)),
forwarded.next().keyValue()); + assertEquals(new KeyValue<>("topic", "t1"), forwarded.next().keyValue()); + assertEquals(new
KeyValue<>("partition", 0), forwarded.next().keyValue()); + assertEquals(new KeyValue<>("offset", 0L), forwarded.next().keyValue()); +
assertEquals(new KeyValue<>("timestamp", 0L), forwarded.next().keyValue()); + assertEquals(new KeyValue<>("key", "foo"),
forwarded.next().keyValue()); + assertEquals(new KeyValue<>("value", 5L), forwarded.next().keyValue()); + } + +
context.resetForwards(); + + // record metadata should be "sticky" + context.setOffset(1L); + context.setTimestamp(10L); + + { +
processor.process("bar", 50L); + final Iterator<CapturedForward> forwarded = context.forwarded().iterator(); + assertEquals(new
KeyValue<>("appId", "testMetadata"), forwarded.next().keyValue()); + assertEquals(new KeyValue<>("taskId", new TaskId(0, 0)),
forwarded.next().keyValue()); + assertEquals(new KeyValue<>("topic", "t1"), forwarded.next().keyValue()); + assertEquals(new
KeyValue<>("partition", 0), forwarded.next().keyValue()); + assertEquals(new KeyValue<>("offset", 1L), forwarded.next().keyValue()); +
assertEquals(new KeyValue<>("timestamp", 10L), forwarded.next().keyValue()); + assertEquals(new KeyValue<>("key", "bar"),
forwarded.next().keyValue()); + assertEquals(new KeyValue<>("value", 50L), forwarded.next().keyValue()); + } + +
context.resetForwards(); + // record metadata should be "sticky" + context.setTopic("t2"); + context.setPartition(30); + + { +
processor.process("baz", 500L); + final Iterator<CapturedForward> forwarded = context.forwarded().iterator(); + assertEquals(new
KeyValue<>("appId", "testMetadata"), forwarded.next().keyValue()); + assertEquals(new KeyValue<>("taskId", new TaskId(0, 0)),
forwarded.next().keyValue()); + assertEquals(new KeyValue<>("topic", "t2"), forwarded.next().keyValue()); + assertEquals(new
KeyValue<>("partition", 30), forwarded.next().keyValue()); + assertEquals(new KeyValue<>("offset", 1L), forwarded.next().keyValue());
+ assertEquals(new KeyValue<>("timestamp", 10L), forwarded.next().keyValue()); + assertEquals(new KeyValue<>("key", "baz"),
forwarded.next().keyValue()); + assertEquals(new KeyValue<>("value", 500L), forwarded.next().keyValue()); + } + } + + @Test + public
void shouldCapturePunctuator() { + final Processor<String, Long> processor = new Processor<String, Long>() { + @Override + public
void init(final ProcessorContext context) { + context.schedule( + 1000L, + PunctuationType.WALL_CLOCK_TIME, + new Punctuator() {
+ @Override + public void punctuate(final long timestamp) { + context.commit(); + } + } + ); + } + + @Override + public void
process(final String key, final Long value) { + } + + @Override + public void punctuate(final long timestamp) { + } + + @Override +
public void close() { + } + }; + + final MockProcessorContext context = new MockProcessorContext(); + + processor.init(context); + +
final MockProcessorContext.CapturedPunctuator capturedPunctuator = context.scheduledPunctuators().get(0); + assertEquals(1000L,
capturedPunctuator.getIntervalMs()); + assertEquals(PunctuationType.WALL_CLOCK_TIME, capturedPunctuator.getType()); +
assertFalse(capturedPunctuator.cancelled()); + + final Punctuator punctuator = capturedPunctuator.getPunctuator(); +
assertFalse(context.committed()); + punctuator.punctuate(1234L); + assertTrue(context.committed()); + } + + @Test + public void
fullConstructorShouldSetAllExpectedAttributes() { + final Properties config = new Properties(); +
config.put(StreamsConfig.APPLICATION_ID_CONFIG, "testFullConstructor"); +
config.put(StreamsConfig.BOOTSTRAP_SERVERS_CONFIG, ""); +
config.put(StreamsConfig.DEFAULT_KEY_SERDE_CLASS_CONFIG, Serdes.String().getClass()); +
config.put(StreamsConfig.DEFAULT_VALUE_SERDE_CLASS_CONFIG, Serdes.Long().getClass()); + + final File dummyFile = new
File(""); + final MockProcessorContext context = new MockProcessorContext(config, new TaskId(1, 1), dummyFile); + +
assertEquals("testFullConstructor", context.applicationId()); + assertEquals(new TaskId(1, 1), context.taskId()); +
assertEquals("testFullConstructor", context.appConfigs().get(StreamsConfig.APPLICATION_ID_CONFIG)); +
assertEquals("testFullConstructor", context.appConfigsWithPrefix("application.").get("id")); + assertEquals(Serdes.String().getClass(),
```

context.keySerde().getClass()); + assertEquals(Serdes.Long().getClass(), context.valueSerde().getClass()); + assertEquals(dummyFile, context.stateDir()); + } +} -- This is an automated message from the Apache Git Service. To respond to the message, please log on GitHub and use the URL above to go to the specific comment. For queries about this service, please contact Infrastructure at: users@infra.apache.org

4. The PR for this work is merged: https://github.com/apache/kafka/pull/4736