Item 343
git_comments:

1. * * Advances synchronized processing time to the given value and fires processing-time timers * accordingly.
2. * Advances input watermark to the given value and fires event-time timers accordingly.
3. * Licensed to the Apache Software Foundation (ASF) under one * or more contributor license agreements. See the NOTICE file * distributed with this work for additional information * regarding copyright ownership. The ASF licenses this file * to you under the Apache License, Version 2.0 (the * "License"); you may not use this file except in compliance * 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.
4. * Advances output watermark to the given value.
5. Remove before firing, so that if the callback adds another identical timer we don't remove it.
6. * Advances processing time to the given value and fires processing-time timers accordingly.
7. * * Simulates the firing of timers and progression of input and output watermarks for a single * computation and key in a Windmill-like streaming environment.
8. * * Returns when the next timer in the given time domain will fire, or {@code null} * if there are no timers scheduled in that time domain.
9. * * Simulates state like {@link InMemoryStateInternals} and provides some extra helper methods.
10. * Licensed to the Apache Software Foundation (ASF) under one * or more contributor license agreements. See the NOTICE file * distributed with this work for additional information * regarding copyright ownership. The ASF licenses this file * to you under the Apache License, Version 2.0 (the * "License"); you may not use this file except in compliance * 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.
11. Nothing
12. * Processes the {@link TimerInternals.TimerData TimerData}.
13. * * A callback that processes a {@link TimerInternals.TimerData TimerData}.
14. * 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.
15. key
16. * Advance the processing time to the specified time.
17. TODO: Should test timer firings: see https://issues.apache.org/jira/browse/BEAM-694

git_commits:

1. **summary:** Closes #1023
   message: Closes #1023

github_issues:

github_issues_comments:

github_pulls:

1. **title:** Deduplicates the 3 in-memory TimerInternals
   body: There were 3 implementations of TimerInternals doing basically the same thing: - TestTimerInternals in ReduceFnTester - TestTimerInternals in TriggerTester (these two were nearly

identical) - BatchTimerInternals (it was a subset of the above) There were also 2 copies of TestInMemoryStateInternals. This change deduplicates and reorganizes them: - Deduplicates the TestInMemoryStateInternals. It might make sense to pull its methods into InMemoryStateInternals and delete the class, I'm not sure. - Factors out the common timer stuff into InMemoryTimerInternals. - TriggerTester's implementation of TestTimerInternals used to (unnecessarily) access TestInMemoryStateInternals, presumably due to copy-paste. Now it uses the regular InMemoryTimerInternals.

github_pulls_comments:

1. R: @tgroh
2. How easy would it be to split this into two commits, one which uses `BatchTimerInternals` everywhere it's relevant (and does any relevant refactoring that moves it to `InMemoryTimerInternals`) and then a second that's just the rename.
3. Which rename? In terms of offered features, `ReduceFnTester.TestTimerInternals` > `TriggerTester.TestTimerInternals` > `BatchTimerInternals`, so I created `InMemoryTimerInternals` which is almost equivalent to `ReduceFnTester.TestTimerInternals` (i.e. powerful enough for all 3 use contexts) and used it everywhere. The old BatchTimerInternals can't be used anywhere except the one place where it was already used.
4. And ReduceFnTester is in the Testing package. Hrm. That's disappointing. So, I'm going to propose a history that lasts for the duration of the PR: Split `ReduceFnTester.TestTimerInternals` into the `ReduceFnTester` and the `TestTimerInternals`, move `TestTimerInternals` to `InMemoryTimerInternals` Use `InMemoryTimerInternals` everywhere, delete unused code After the PR, these can be squashed together; but should make reviewing the two steps simpler.
5. Good idea. Split into 3 commits.
6. @bjchambers for committer

github_pulls_reviews:

1. just `throw new IllegalArgumentException`
2. move to `default` case
3. Assign to a new local variable.
4. Why don't we just use `queue.remove()` here?
5. Do we want to remove all of the timers that are permitted to fire if `timerCallback` is `null`? I don't think we should discard them If this should never happen, then this should be an `IllegalArgumentException` or an `IllegalStateException`
6. Done.
7. Done.
8. Done.
9. Done. I was refactoring the old code but missed this part.
10. I was using null as "no-op callback" (for users who only care about the times of timers, but not about doing anything with them), but now replaced it with an explicit NO_OP callback.
11. This isn't quite what I was expecting. Can we enumerate the cases in which `TimerCallback.NO_OP` is used? It would be useful to ensure that we're not discarding any timers (and letting data get stuck as a result)
12. It's used in 2 cases: 1) GroupAlsoByWindowsViaOutputBufferDoFn, to simply initialize the processing time and synchronized processing time as part of initializing the TimerInternals. No timers are set at that point. 2) In TriggerTester.advanceInputWatermark(Instant) and TriggerTester.advanceProcessingTime(Instant). Per offline discussion, filed https://issues.apache.org/jira/browse/BEAM-694 and linked to it from those methods.
13. ``` /** * Returns the {@link Instant} of the next timer ... */ ``` (No need for `@return` when the javadoc already includes "Returns ...")
14. Done.

jira_issues:

jira_issues_comments: