KIP-622: Add currentSystemTimeMs and currentStreamTimeMs to ProcessorContext

Status

Current state: Adopted (3.0.0)

Discussion thread: *here*

JIRA: KAFKA-10062

Please keep the discussion on the mailing list rather than commenting on the wiki (wiki discussions get unwieldy fast).

Motivation

The motivation is to source the internally cached system timestamp from the Kafka Stream runtime. A generic example would be if you want to execute an action when a message arrives based on the current time and then potentially some time later. Currently, if you want to implement a solution that uses wall-clock time and performs actions based on it, you would use System.currentTimeMillis(). The problem is that tests have no control over this value because it is using 'MockTime'.

For stream time, the time is advanced before processing a new record that has a time higher than the currently known stream time. Therefore, it is possible that the ProcessorContext#timestamp may be earlier than the streamTime if the records not processed in order by time. It is simply not reliable to use solely timestamp and System.currentTimeMillis() for actions based on time.

Public Interfaces

> ***Update:***
>
> *This KIP overlaps with KIP-478 and we also added the new methods currentSystemTimeMs() and currentStreamTimeMs() to `api.ProcessorContext` in 3.2.0 release.*
>
> ***Update (2022-04-14):***
>
> *As part of KIP-820 implementation, we are adding the new methods currentSystemTimeMs() and currentStreamTimeMs() to `api.MockProcessorContext` as well for completeness.*

Add new public API for currentSystemTimeMs:

org.apache.kafka.streams.processor.ProcessorContext#currentSystemTimeMs

```
/**
 * Return the current system timestamp (also called wall-clock time) in milliseconds.
 *
 * <p>
 * Note: this method returns the internally cached system timestamp from the Kafka Stream runtime.
 * Thus, it may return a different value compared to {@code System.currentTimeMillis()}
 * <p>
 *
 * @return the current system timestamp in milliseconds
 */
    long currentSystemTimeMs();
```

Add new public API for currentStreamTimeMs:

org.apache.kafka.streams.processor.ProcessorContext#currentStreamTimeMs

```
    /**
     * Return the current stream-time in milliseconds.
     *
     * <p>
     * Stream-time is the maximum observed {@link TimestampExtractor record timestamp} so far
     * (including the currently processed record), i.e., it can be considered a high-watermark.
     * Stream-time is tracked on a per-task basis and is preserved across restarts and during task migration.
     * <p>
     *
     * Note: this method is not supported for global processors (cf. {@link Topology#addGlobalStore} (...)
     * and {@link StreamsBuilder#addGlobalStore} (...),
     * because there is no concept of stream-time for this case.
     * Calling this method in a global processor will result in an {@link UnsupportedOperationException}.
     *
     * @return the current stream-time in milliseconds
     */
        long currentStreamTimeMs();
```

Add new methods to MockProcessorContext for testing purposes:

`org.apache.kafka.streams.processor.MockProcessorContext#setRecordTimestamp`

This method sets record timestamp.

```
        public void setRecordTimestamp(final long recordTimestamp) {
        this.recordTimestamp = recordTimestamp;
    }
```

`org.apache.kafka.streams.processor.MockProcessorContext#setCurrentSystemTimeMs`

This method sets system timestamp.

```
        public void setCurrentSystemTimeMs(final long currentSystemTimeMs) {
        this.currentSystemTimeMs = currentSystemTimeMs;
    }
```

`org.apache.kafka.streams.processor.MockProcessorContext#setCurrentStreamTimeMs`

This method sets stream time.

```
        public void setCurrentStreamTimeMs(final long currentStreamTimeMs) {
        this.currentStreamTimeMs = currentStreamTimeMs;
    }
```

Proposed Changes

The goal is to create two separate public API's that return the current cached system time in milliseconds and the current stream time in milliseconds.

currentSystemTimeMs

It is expected that this will return the internally cached system timestamp from the Kafka Stream runtime. Thus, it may return a different value compared to System.currentTimeMillis(). The cached system time represents the time when we start processing / punctuating, and it would not change throughout the process / punctuate. So this method will return current system time (also called wall-clock time) known from kafka streams runtime.

currentStreamTimeMs

It is expected that this will return the StreamTask's time from the partition group. This stream time will be the maximum timestamp of any record yet processed by the task. This would provide the actual stream time because relying on the timestamp of records is not reliable when the records might not be processed in order of time.

Compatibility, Deprecation, and Migration Plan

This change will be backwards compatible because it is adding two new API's. However we want to deprecate org.apache.kafka.streams.processor.MockProcessorContext#setTimestamp as it's name is misleading and we are adding new method org.apache.kafka.streams.processor.MockProcessorContext#setRecordTimestamp which does the same work.

Rejected Alternatives

One single API with punctuator type as input

It seems more straightforward to create two separate, well-named API's instead of a single API with punctuator type as an input.