

# KIP-671: Introduce Kafka Streams Specific Uncaught Exception Handler


- [Status](#)
- [Motivation](#)
- [Public Interfaces](#)
- [Proposed Changes](#)
- [Rejected Alternatives](#)

## Status

**Current state:** *Accepted*

**Discussion thread:** [here](#)

**JIRA:** [here](#) or [here](#). The replace thread is

 Unable to render Jira issues macro, execution error.

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

## Motivation

Currently there is not a native way to shutdown an entire KStreams application from a StreamThread. By adding the handler we improve the way Streams can handle errors and preventing corruption. This functionality would be useful for immediately halting processing to prevent data from being corrupted. Though this is best effort it will shutdown every thread in the client but may have faults in certain network partition scenarios.

This would help recover from errors or problems such as the following:

- source/sink topic deleted. A handler will be exposed to the users created for error source topics in [KIP-662](#).
- Serialization or Deserialization failures. This can add another option to prevent rolling thread death.
  - currently they have the option of returning a Continue or Fail Enum. In the future we can expand this to include Shutdown and then throw ShutdownRequestedException in response
- Add another option in [KIP-399](#) if failing and having only the thread fail is not comprehensive enough

## Public Interfaces

KafkaStreams.java

```

package org.apache.kafka.streams;

public enum StreamsUncaughtExceptionResponse {
    REPLACE_THREAD,
    SHUTDOWN_CLIENT,
    SHUTDOWN_APPLICATION;
}

public interface StreamsUncaughtExceptionHandler {
    StreamsUncaughtExceptionResponse handle(final Throwable exception);
}

KafkaStreams.java/**
 * Set the handler invoked when a {@link StreamsConfig#NUM_STREAM_THREADS_CONFIG internal thread} abruptly
 * terminates due to an uncaught exception.
 *
 * @param eh the uncaught exception handler of type {@link StreamsUncaughtExceptionHandler} for all internal
 * threads; {@code null} deletes the current handler
 * @throws IllegalStateException if this {@code KafkaStreams} instance is not in state {@link State#CREATED
 * CREATED}.
 */public void setUncaughtExceptionHandler(final StreamsUncaughtExceptionHandler eh) ;

```

## Proposed Changes

We propose to add a new streams specific uncaught exception handler that will do the following:

### REPLACE\_THREAD:

- The current thread is shutdown and transits to state DEAD.
- A new thread is started if the Kafka Streams client is in state RUNNING or REBALANCING.
- For the Global thread this option will log an error and revert to shutting down the client until the option had been added

### SHUTDOWN\_CLIENT (default)

- All Stream Threads in the client are shutdown and they transit to state DEAD
- The Global Thread is shutdown if the client has one
- The Kafka Streams client transits to state NOT\_RUNNING.
- The State directory cleaner thread will stop
- The RocksDB metrics recording thread will stop.

### SHUTDOWN\_APPLICATION

- The shutdown is communicated to the other Kafka Streams clients through the rebalance protocol.
- All Stream Threads across the entire application are shutdown and they transit to state DEAD
- All Global Threads across the entire application are shutdown
- All Kafka Streams clients, i.e., the entire Kafka Streams application, is shutdown.
- All Kafka Streams clients transit to state ERROR.
- The State directory cleaner thread stop
- The RocksDB metrics recording thread will stop.

If the old handler is set and the new one is not the behavior will not change. Other wise the new handler will take precedence.

In order to communicate the shutdown request from one client to the others we propose to update the SubscriptionInfoData to include a short field which will encode an error code. The error will be propagated through the metadata during a rejoin event via the assignor. The actual shutdown will be handled by the StreamsRebalanceListener, this is where the INCOMPLETE\_SOURCE\_TOPIC\_METADATA error can also be handled.

## Compatibility, Deprecation, and Migration Plan

The SubscriptionInfoData will be upgraded to version 8 because we are adding a field for an error code to be propagated through the application.

Deprecate the old handler.

## Rejected Alternatives

- *Two paths, Internal Error via exception and a request method for users to call*
- *Add a config option to shutdown when ever a user error is thrown - no flexible enough*
- *Throwing an Exception instead of shutdown Application*
- *Have a method that shutdown the application instead of the error handler.*