

# Package ‘AWR.Kinesis’

January 20, 2025

**Type** Package

**Maintainer** Gergely Daroczi <daroczig@rappporter.net>

**Author** Gergely Daroczi <daroczig@rappporter.net>

**Title** Amazon 'Kinesis' Consumer Application for Stream Processing

**Description** Fetching data from Amazon 'Kinesis' Streams using the Java-based 'MultiLangDaemon' interacting with Amazon Web Services ('AWS') for easy stream processing from R. For more information on 'Kinesis', see <<https://aws.amazon.com/kinesis>>.

**Version** 1.7.6

**Date** 2023-08-17

**URL** <https://github.com/daroczig/AWR.Kinesis>

**License** AGPL-3

**Imports** AWR, logger, jsonlite, rJava

**RoxygenNote** 7.2.3

**Encoding** UTF-8

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2023-08-19 00:02:34 UTC

## Contents

AWR.Kinesis-package . . . . .	2
checkpoint . . . . .	2
kinesis_consumer . . . . .	2
kinesis_get_records . . . . .	4
kinesis_put_record . . . . .	5

<b>Index</b>	<b>6</b>
--------------	----------

---

AWR.Kinesis-package    *An R Kinesis Consumer*

---

**Description**

Please find more details in the README.md file.

---

checkpoint                    *Checkpoint at current or given sequence number*

---

**Description**

Checkpoint at current or given sequence number

**Usage**

```
checkpoint(sequenceNumber)
```

**Arguments**

sequenceNumber    optional

---

kinesis\_consumer            *Run Kinesis Consumer application*

---

**Description**

Run Kinesis Consumer application

**Usage**

```
kinesis_consumer(  
  initialize,  
  processRecords,  
  shutdown,  
  checkpointing = TRUE,  
  updater,  
  logfile = tempfile()  
)
```

**Arguments**

initialize	optional function to be run on startup. Please note that the variables created inside of this function will not be available to eg processRecords, so make sure to store the shared variables in the parent or global namespace
processRecords	function to process records taking a data.frame object with partitionKey, sequenceNumber and data columns as the records argument. Probably you only need the data column from this object
shutdown	optional function to be run when finished processing all records in a shard
checkpointing	if set to TRUE (default), kinesis_consumer will checkpoint after each processRecords call. To disable checkpointing altogether, set this to FALSE. If you want to checkpoint periodically, set this to the frequency in minutes as integer.
updater	optional list of list(s) including frequency (in minutes) and function to be run, most likely to update some objects in the parent or global namespace populated first in the initialize call. If the frequency is smaller than how long the processRecords call runs, it will be triggered once after each processRecords call
logfile	file path of the log file. To disable logging, set log_threshold to something high with the AWR.Kinesis namespace

**Note**

Don't run this function directly, it is to be called by the MultiLangDaemon. See the package README for more details.

**References**

<https://github.com/aws-labs/amazon-kinesis-client/blob/v1.x/src/main/java/com/amazonaws/services/kinesis/multilang/package-info.java>

**Examples**

```
## Not run:
log_threshold(FATAL, namespace = 'AWR.Kinesis')
AWS.Kinesis::kinesis_consumer(
  initialize = function() log_info('Loading some data'),
  processRecords = function(records) log_info('Received some records from Kinesis'),
  updater = list(list(1, function() log_info('Updating some data every minute')),
                list(1/60, function() log_info('This is a high frequency updater call')))
)

## End(Not run)
```

---

kinesis\_get\_records     *Get record from a Kinesis Stream*

---

### Description

Get record from a Kinesis Stream

### Usage

```
kinesis_get_records(  
    stream,  
    region = "us-west-1",  
    limit = 25,  
    shard_id,  
    iterator_type = c("TRIM_HORIZON", "LATEST", "AT_SEQUENCE_NUMBER",  
        "AFTER_SEQUENCE_NUMBER", "AT_TIMESTAMP"),  
    start_sequence_number,  
    start_timestamp  
)
```

### Arguments

stream	stream name (string)
region	AWS region (string)
limit	number of records to fetch
shard_id	optional shard id - will pick a random active shard if left empty
iterator_type	shard iterator type
start_sequence_number	for AT_SEQUENCE_NUMBER and AFTER_SEQUENCE_NUMBER iterators
start_timestamp	for AT_TIMESTAMP iterator

### Value

character vector that you might want to post-process eg with `jsonlite::stream_in`

### Note

Use this no more than getting sample data from a stream - it's not intended for prod usage.

### References

<https://docs.aws.amazon.com/AWSJavaSDK/latest/javadoc/com/amazonaws/services/kinesis/model/GetRecordsRequest.html>

---

kinesis_put_record	<i>Write a record to a Kinesis Stream</i>
--------------------	---

---

**Description**

Write a record to a Kinesis Stream

**Usage**

```
kinesis_put_record(stream, region = "us-west-1", data, partitionKey)
```

**Arguments**

stream	stream name (string)
region	AWS region (string)
data	data blob (string)
partitionKey	determines which shard in the stream the data record is assigned to, eg username, stock symbol etc (string)

**Value**

invisible list including the shard id and sequence number

**References**

<https://docs.aws.amazon.com/AWSJavaSDK/latest/javadoc/com/amazonaws/services/kinesis/model/PutRecordRequest.html>

**Examples**

```
## Not run:  
df <- mtcars[1, ]  
str(kinesis_put_record('test-AWR', data = jsonlite::toJSON(df), partitionKey = row.names(df)))  
  
## End(Not run)
```

# Index

[AWR.Kinesis-package](#), [2](#)

[checkpoint](#), [2](#)

[kinesis\\_consumer](#), [2](#)

[kinesis\\_get\\_records](#), [4](#)

[kinesis\\_put\\_record](#), [5](#)