

Kafka REST API SwaggerUI

Table of Contents

1. Overview	1
1.1. Version information	1
1.2. Contact information.....	1
1.3. URI scheme.....	1
1.4. Tags	1
2. Chapter of manual content 1	2
2.1. Sub chapter	2
3. Chapter of manual content 2	3
4. Resources	4
4.1. Kafka-controller	4
4.1.1. List brokers in this cluster	4
4.1.2. getMessage	4
4.1.3. deleteOldConsumerGroup	5
4.1.4. getLastCommitTimestamp	6
4.1.5. resetOffset.....	7
4.1.6. List new consumer groups	7
4.1.7. List new consumer groups lag and offset by consumer group and topic	8
4.1.8. List old consumer groups from zk	9
4.1.9. List old consumer groups lag and offset by consumer group and topic	10
4.1.10. Add a partition to the topic	10
4.1.11. Check the partition reassignment process	11
4.1.12. Execute the partition reassignment.....	12
4.1.13. Generate plan for the partition reassignment.....	13
4.1.14. List topics	14
4.1.15. Create a topic.....	14
4.1.16. Describe a topic by fetching the metadata and config	15
4.1.17. Delete a topic (you should enable topic deletion	15
4.1.18. Create topic configs	16
4.1.19. Get topic configs	17
4.1.20. Update topic configs.....	18
4.1.21. Delete topic configs	18
4.1.22. Get topic config by key	19
4.1.23. Delete a topic config by key	20
4.1.24. Create a topic config by key	21
4.1.25. Update a topic config by key	21
4.1.26. Tell if a topic exists.....	22

4.1.27. Write a message to the topic, for testing purpose	23
4.1.28. List topics Brief	24
4.2. Zookeeper-controller	24
4.2.1. zkConnState	24
4.2.2. getEnv	25
4.2.3. ls	25
4.2.4. ping	26
4.2.5. getStat	27
5. Definitions	28
5.1. AddPartition	28
5.2. BrokerInfo	28
5.3. ConsumerGroupDesc	28
5.4. HostAndPort	29
5.5. Map«int,long»	29
5.6. ReassignWrapper	29
5.7. TopicAndPartition	30
5.8. TopicBrief	30
5.9. TopicDetail	30
5.10. TopicMeta	30
5.11. TopicPartitionInfo	31
5.12. ZkServerClient	31
5.13. ZkServerEnvironment	32
5.14. ZkServerStat	32

Chapter 1. Overview

Kafka REST API SwaggerUI

1.1. Version information

Version : 0.1.0

1.2. Contact information

Contact : gnuhpc

Contact Email : gnuahpc@gmail.com

1.3. URI scheme

Host : localhost:8080

BasePath : /

1.4. Tags

- kafka-controller : Kafka Controller
- zookeeper-controller : Zookeeper Controller

Chapter 2. Chapter of manual content 1

This is some dummy text

2.1. Sub chapter

Dummy text of sub chapter

Chapter 3. Chapter of manual content 2

This is some dummy text

Chapter 4. Resources

4.1. Kafka-controller

Kafka Controller

4.1.1. List brokers in this cluster

```
GET /kafka/brokers
```

Responses

HTTP Code	Description	Schema
200	OK	< BrokerInfo > array
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.2. getMessage

```
GET /kafka/consumer/{topic}/{partition}/{offset}
```

Parameters

Type	Name	Description	Schema
Query	decoder <i>optional</i>	decoder	string
Query	offset <i>optional</i>	offset	integer(int64)

Type	Name	Description	Schema
Query	partition <i>optional</i>	partition	integer(int32)
Query	topic <i>optional</i>	topic	string

Responses

HTTP Code	Description	Schema
200	OK	string
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.3. deleteOldConsumerGroup

```
DELETE /kafka/consumergroup/old/{consumergroup}
```

Parameters

Type	Name	Description	Schema
Query	consumergroup <i>optional</i>	consumergroup	string

Responses

HTTP Code	Description	Schema
200	OK	boolean
204	No Content	No Content

HTTP Code	Description	Schema
401	Unauthorized	No Content
403	Forbidden	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.4. getLastCommitTimestamp

```
GET /kafka/consumergroup/{consumergroup}/{topic}/lastcommittime
```

Parameters

Type	Name	Description	Schema
Query	consumergroup <i>optional</i>	consumergroup	string
Query	topic <i>optional</i>	topic	string

Responses

HTTP Code	Description	Schema
200	OK	< string, < string, integer(int64) > map > map
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- /

4.1.5. resetOffset

```
PUT /kafka/consumergroup/{consumergroup}/{topic}/{partition}/{offset}
```

Parameters

Type	Name	Description	Schema
Query	consumergroup <i>optional</i>	consumergroup	string
Query	offset <i>optional</i>	offset	string
Query	partition <i>optional</i>	partition	integer(int32)
Query	topic <i>optional</i>	topic	string

Responses

HTTP Code	Description	Schema
200	OK	No Content
201	Created	No Content
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- /

4.1.6. List new consumer groups

```
GET /kafka/consumergroups/new
```

Parameters

Type	Name	Description	Schema
Query	t <i>optional</i>	t	string

Responses

HTTP Code	Description	Schema
200	OK	< string > array
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- *application/json*

Produces

- */*

4.1.7. List new consumergroups lag and offset by consumer group and topic

```
GET /kafka/consumergroups/new/{consumerGroup}/{topic}
```

Parameters

Type	Name	Description	Schema
Query	consumerGroup <i>optional</i>	consumerGroup	string
Query	topic <i>optional</i>	topic	string

Responses

HTTP Code	Description	Schema
200	OK	< ConsumerGroupDe sc > array
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.8. List old consumer groups from zk

```
GET /kafka/consumergroups/old
```

Parameters

Type	Name	Description	Schema
Query	t <i>optional</i>	t	string

Responses

HTTP Code	Description	Schema
200	OK	< string > array
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- /

4.1.9. List old consumergroups lag and offset by consumergroup and topic

```
GET /kafka/consumerGroups/old/{consumerGroup}/{topic}
```

Parameters

Type	Name	Description	Schema
Query	consumerGroup <i>optional</i>	consumerGroup	string
Query	topic <i>optional</i>	topic	string

Responses

HTTP Code	Description	Schema
200	OK	< ConsumerGroupDe sc > array
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- /

4.1.10. Add a partition to the topic

```
POST /kafka/partitions/add
```

Parameters

Type	Name	Description	Schema
Body	addPartition <i>required</i>	addPartition	AddPartition

Responses

HTTP Code	Description	Schema
200	OK	TopicMeta
201	Created	No Content
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- [application/json](#)

Produces

- [/](#)

4.1.11. Check the partition reassignment process

```
PUT /kafka/partitions/reassign/check
```

Parameters

Type	Name	Description	Schema
Body	reassignStr <i>required</i>	reassignStr	string

Responses

HTTP Code	Description	Schema
-1	Reassignment Failed	No Content
0	Reassignment In Progress	No Content

HTTP Code	Description	Schema
1	Reassignment Completed	No Content
200	OK	< string, integer(int32) > map
201	Created	No Content
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.12. Execute the partition reassignment

```
PUT /kafka/partitions/reassign/execute
```

Parameters

Type	Name	Description	Schema	Default
Query	throttle <i>optional</i>	throttle	integer(int64)	<code>"-1"</code>
Body	reassignStr <i>required</i>	reassignStr	string	

Responses

HTTP Code	Description	Schema
200	OK	< string, integer(int32) > map
201	Created	No Content
401	Unauthorized	No Content

HTTP Code	Description	Schema
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.13. Generate plan for the partition reassignment

POST /kafka/partitions/reassign/generate

Parameters

Type	Name	Description	Schema
Body	reassignWrapper <i>required</i>	reassignWrapper	ReassignWrapper

Responses

HTTP Code	Description	Schema
200	OK	< string > array
201	Created	No Content
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.14. List topics

GET /kafka/topics

Responses

HTTP Code	Description	Schema
200	OK	< string > array
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.15. Create a topic

POST /kafka/topics/create

Parameters

Type	Name	Description	Schema
Body	topic <i>required</i>	topic	TopicDetail

Responses

HTTP Code	Description	Schema
201	Created	boolean
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.16. Describe a topic by fetching the metadata and config

```
GET /kafka/topics/{topic}
```

Parameters

Type	Name	Description	Schema
Path	topic <i>required</i>	topic	string

Responses

HTTP Code	Description	Schema
200	OK	TopicMeta
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.17. Delete a topic (you should enable topic deletion

```
DELETE /kafka/topics/{topic}
```

Parameters

Type	Name	Description	Schema
Path	topic <i>required</i>	topic	string

Responses

HTTP Code	Description	Schema
200	OK	boolean
204	No Content	No Content
401	Unauthorized	No Content
403	Forbidden	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.18. Create topic configs

POST /kafka/topics/{topic}/conf

Parameters

Type	Name	Description	Schema
Path	topic <i>required</i>	topic	string
Body	prop <i>required</i>	prop	< string, object > map

Responses

HTTP Code	Description	Schema
200	OK	< string, object > map
201	Created	No Content

HTTP Code	Description	Schema
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.19. Get topic configs

```
GET /kafka/topics/{topic}/conf
```

Parameters

Type	Name	Description	Schema
Path	topic <i>required</i>	topic	string

Responses

HTTP Code	Description	Schema
200	OK	< string, object > map
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.20. Update topic configs

```
PUT /kafka/topics/{topic}/conf
```

Parameters

Type	Name	Description	Schema
Path	topic <i>required</i>	topic	string
Body	prop <i>required</i>	prop	< string, object > map

Responses

HTTP Code	Description	Schema
200	OK	< string, object > map
201	Created	No Content
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.21. Delete topic configs

```
DELETE /kafka/topics/{topic}/conf
```

Parameters

Type	Name	Description	Schema
Path	topic <i>required</i>	topic	string

Type	Name	Description	Schema
Body	delProps <i>required</i>	delProps	< string > array

Responses

HTTP Code	Description	Schema
200	OK	< string, object > map
204	No Content	No Content
401	Unauthorized	No Content
403	Forbidden	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.22. Get topic config by key

```
GET /kafka/topics/{topic}/conf/{key}
```

Parameters

Type	Name	Description	Schema
Path	key <i>required</i>	key	string
Path	topic <i>required</i>	topic	string

Responses

HTTP Code	Description	Schema
200	OK	< string, object > map

HTTP Code	Description	Schema
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.23. Delete a topic config by key

```
DELETE /kafka/topics/{topic}/conf/{key}
```

Parameters

Type	Name	Description	Schema
Path	key <i>required</i>	key	string
Path	topic <i>required</i>	topic	string

Responses

HTTP Code	Description	Schema
200	OK	boolean
204	No Content	No Content
401	Unauthorized	No Content
403	Forbidden	No Content

Consumes

- `application/json`

Produces

- /

4.1.24. Create a topic config by key

```
POST /kafka/topics/{topic}/conf/{key}={value}
```

Parameters

Type	Name	Description	Schema
Path	key <i>required</i>	key	string
Path	topic <i>required</i>	topic	string
Path	value <i>required</i>	value	string

Responses

HTTP Code	Description	Schema
200	OK	< string, object > map
201	Created	No Content
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- /

4.1.25. Update a topic config by key

```
PUT /kafka/topics/{topic}/conf/{key}={value}
```


Parameters

Type	Name	Description	Schema
Path	key <i>required</i>	key	string
Path	topic <i>required</i>	topic	string
Path	value <i>required</i>	value	string

Responses

HTTP Code	Description	Schema
200	OK	< string, object > map
201	Created	No Content
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.26. Tell if a topic exists

```
GET /kafka/topics/{topic}/exist
```

Parameters

Type	Name	Description	Schema
Query	topic <i>required</i>	topic	string

Responses

HTTP Code	Description	Schema
200	OK	boolean
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.1.27. Write a message to the topic, for testing purpose

```
POST /kafka/topics/{topic}/write
```

Parameters

Type	Name	Description	Schema
Path	topic <i>required</i>	topic	string
Body	message <i>required</i>	message	string

Responses

HTTP Code	Description	Schema
201	Created	No Content
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `text/plain`

Produces

- `/`

4.1.28. List topics Brief

```
GET /kafka/topicsbrief
```

Responses

HTTP Code	Description	Schema
200	OK	< TopicBrief > array
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.2. Zookeeper-controller

Zookeeper Controller

4.2.1. zkConnState

```
GET /zk/connstate
```

Responses

HTTP Code	Description	Schema
200	OK	string
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.2.2. getEnv

```
GET /zk/env
```

Responses

HTTP Code	Description	Schema
200	OK	< string, ZkServerEnvironment > map
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.2.3. ls

```
GET /zk/ls/{path}
```

Parameters

Type	Name	Description	Schema
Path	path <i>required</i>	path	string

Responses

HTTP Code	Description	Schema
200	OK	< string > array
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- `/`

4.2.4. ping

GET /zk/ping

Responses

HTTP Code	Description	Schema
200	OK	string
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- `application/json`

Produces

- /

4.2.5. getStat

GET /zk/stat

Responses

HTTP Code	Description	Schema
200	OK	< string, ZkServerStat > map
401	Unauthorized	No Content
403	Forbidden	No Content
404	Not Found	No Content

Consumes

- application/json

Produces

- /

Chapter 5. Definitions

5.1. AddPartition

Name	Schema
numPartitionsAdded <i>optional</i>	integer(int32)
replicaAssignment <i>optional</i>	string
topic <i>optional</i>	string

5.2. BrokerInfo

Name	Schema
endPoints <i>optional</i>	< string > array
host <i>optional</i>	string
id <i>optional</i>	integer(int32)
jmxPort <i>optional</i>	integer(int32)
port <i>optional</i>	integer(int32)
rack <i>optional</i>	string
securityProtocol <i>optional</i>	object
startTime <i>optional</i>	string(date-time)
version <i>optional</i>	integer(int32)

5.3. ConsumerGroupDesc

Name	Schema
consumerId <i>optional</i>	string
currentOffset <i>optional</i>	integer(int64)
host <i>optional</i>	string
lag <i>optional</i>	integer(int64)
logEndOffset <i>optional</i>	integer(int64)
partitionId <i>optional</i>	integer(int32)
state <i>optional</i>	enum (RUNNING, PENDING)
topic <i>optional</i>	string

5.4. HostAndPort

Name	Schema
hostText <i>optional</i>	string
port <i>optional</i>	integer(int32)

5.5. Map«int,long»

Type : < string, integer(int64) > map

5.6. ReassignWrapper

Name	Schema
brokers <i>optional</i>	< integer(int32) > array
topics <i>optional</i>	< string > array

5.7. TopicAndPartition

Type : object

5.8. TopicBrief

Name	Schema
isrRate <i>optional</i>	integer(int64)
numPartition <i>optional</i>	integer(int32)
topic <i>optional</i>	string

5.9. TopicDetail

Name	Schema
factor <i>optional</i>	integer(int32)
name <i>optional</i>	string
partitions <i>optional</i>	integer(int32)
prop <i>optional</i>	< string, object > map
reassignStr <i>optional</i>	string

5.10. TopicMeta

Name	Schema
partitionCount <i>optional</i>	integer(int32)
replicationFactor <i>optional</i>	integer(int32)
topicCustomConfigs <i>optional</i>	< string, object > map

Name	Schema
topicName <i>optional</i>	string
topicPartitionInfos <i>optional</i>	< TopicPartitionInfo > array

5.11. TopicPartitionInfo

Name	Schema
endOffset <i>optional</i>	integer(int64)
in_sync <i>optional</i>	boolean
isr <i>optional</i>	< string > array
leader <i>optional</i>	string
messageAvailable <i>optional</i>	integer(int64)
partitionId <i>optional</i>	integer(int32)
replicas <i>optional</i>	< string > array
startOffset <i>optional</i>	integer(int64)

5.12. ZkServerClient

Name	Schema
host <i>optional</i>	string
ops <i>optional</i>	integer(int32)
port <i>optional</i>	integer(int32)
queued <i>optional</i>	integer(int32)

Name	Schema
received <i>optional</i>	integer(int32)
sent <i>optional</i>	integer(int32)

5.13. ZkServerEnvironment

Name	Schema
attributes <i>optional</i>	< string, string > map

5.14. ZkServerStat

Name	Schema
avgLatency <i>optional</i>	integer(int32)
buildDate <i>optional</i>	string
clients <i>optional</i>	< ZkServerClient > array
connections <i>optional</i>	integer(int32)
maxLatency <i>optional</i>	integer(int32)
minLatency <i>optional</i>	integer(int32)
mode <i>optional</i>	enum (Leader, Follower, Observer)
nodes <i>optional</i>	integer(int32)
outstanding <i>optional</i>	integer(int32)
received <i>optional</i>	integer(int32)
sent <i>optional</i>	integer(int32)

Name	Schema
version <i>optional</i>	string
zxId <i>optional</i>	string