aliyun-log-cli Documentation

Release 0.1.6

Alibaba Cloud

Jan 25, 2018

Contents

1 User Guide **1**
- 1.1 Content . . . 1
- 1.2 Introduction . . . 2
- 1.3 Installation . . . 3
- 1.4 Configure CLI . . . 3
- 1.5 Input and Output . . . 5
- 1.6 Command Reference . . . 6
- 1.7 Other resources . . . 13

2 **15**
- 2.1 . . . 15

3 **27**
- 3.1 Logtail . . . 27
- 3.2 . . . 35
- 3.3 . . . 36
- 3.4 . . . 36
- 3.5 JMES . . . 36

4 Mapped SDK API **37**
- 4.1 Request and Config Class . . . 37
- 4.2 Project . . . 37
- 4.3 Logstore . . . 38
- 4.4 Index . . . 38
- 4.5 Logtail Config . . . 38
- 4.6 Machine Group . . . 38
- 4.7 Apply Logtail Config . . . 39
- 4.8 Shard . . . 39
- 4.9 Cursor . . . 39
- 4.10 Logs . . . 39
- 4.11 Consumer group . . . 39
- 4.12 Shipper . . . 40
- 4.13 Definitions . . . 40

5 Indices and tables **63**

CHAPTER 1

User Guide

README

1.1 Content

- *Introduction*
  - *Brief*
  - *Major Features*
- *Installation*
  - *Operation System*
  - *Supported Version*
  - *Installation Method*
  - *Full Usage list*
- *Configure CLI*
  - *Configure AK and Endpoint*
  - *Modify the configuration file*
  - *Multiple Account*
- *Input and Output*
  - *Inputs*
  - *Parameter Validation*
  - *Output*
  - *Filter output*

    - *Further Process*
- *Command Reference*
    - *Command Specification*
    - *Alias*
    - *Subcommand and parameters*
    - *Global options*
    - *Command categories*
        * *Project management*
        * *Logstore management*
        * *Shard management*
        * *Machine group management*
        * *Logtail config management*
        * *Machine group and Logtail Config Mapping*
        * *Index management*
        * *Cursor management*
        * *Logs write and consume*
        * *Shipper management*
        * *Consumer group management*
- *Best Practice*
- *Troubleshooting*
- *Other resources*

1.2 Introduction

The Alicloud log service provides with Web and SDK flavor to operate log service and analyzie logs. To make it more convinient to do automation, we release this command line interface (CLI).

1.2.1 Brief

Alicloud log service command line console, support almost all operations as web. It also supports incomplete log query check and query cross multiple pages. It could even do project settings copy cross multiple regions.

1.2.2 Major Features

- Support almost all 50+ REST API of log service.
- Multiple account support to support cross region operation.
- Log query incomplete check and automatically query cross pagination.
- Multiple confidential storage types, from file, commandline to env variables.
- Support command line based or file based inputs, complete formation validations.

- Support JMES filter to do further process on results, e.g. select specific fields from json.
- Cross platforms support (Windows, Linux and Mac), Python based and friendly to Py2 and Py3 even Pypy. Support Pip installation.

1.3 Installation

1.3.1 Operation System

The CLI supports below operation system:

- Windows
- Mac
- Linux

1.3.2 Supported Version

Python 2.6, 2.7, 3.3, 3.4, 3.5, 3.6, PyPy, PyPy3

1.3.3 Installation Method

Run below command to install the CLI:

```
> pip install -U aliyun-log-cli
```

1.3.4 Full Usage list

Run below command to get the full usage list:

```
> aliyun --help
```

1.4 Configure CLI

Refer to Configuration to get the access ID/key and endpoints.

1.4.1 Configure AK and Endpoint

There're three ways to configure the access key and endpoint and they're prioritized as below:

- Parameters

```
> aliyun log create_project ..... --access-id=<value> --access-key=<value> --region-
↪endpoint=<value>
```

Note: Any sub command support such way to overwrite the AK setings in later ways (env or config file) for the specific operations.

- Environment Variables

- ALIYUN_LOG_CLI_ACCESSID
- ALIYUN_LOG_CLI_ACCESSKEY
- ALIYUN_LOG_CLI_ENDPOINT
- Local configuration file

You could store them at `~/.aliyunlogcli`, the default section name is `main`

```
[main]
access-id=
access-key=
region-endpoint=
```

1.4.2 Modify the configuration file

Use the command "configure" to modify the configuration file:

```
> aliyun configure access_id access_key cn-beijing.log.aliyuncs.com
```

1.4.3 Multiple Account

1. Store multiple accounts for some use cases (e.g. test, multiple region operations)

```
> aliyun configure access_id1 access_key1 cn-beijing.log.aliyuncs.com
> aliyun configure access_id2 access_key2 cn-hangzhou.log.aliyuncs.com test
```

AK is stored as:

```
[main]
access-id=access_id1
access-key=access_key1
region-endpoint=cn-beijing.log.aliyuncs.com

[test]
access-id=access_id2
access-key=access_key2
region-endpoint=cn-hangzhou.log.aliyuncs.com
```

2. Use specific account

Any subcommand could use global opton `--client-name=<value>` to use specific configured account. e.g:

```
> aliyun log create_project ..... --client-name=test
```

It will use `test` to create the project.

3. Other Case

In some case, we need to operate cross regions, e.g.

```
> aliyun log copy_project --from_project="p1" --to_project="p1" --to_client=test
```

It will use account `main` to copy project `p1` in its region to another region under account `test`

1.5 Input and Output

1.5.1 Inputs

1. Normally case:

```
> aliyun log get_logs --request="{\"topic\": \"\", \"logstore\": \"logstore1\", \
↪"project\": \"dlq-test-cli-123\", \"toTime\": \"2018-01-01 10:10:10\", \"offset\": \
↪"0\", \"query\": \"*\", \"line\": \"10\", \"fromTime\": \"2018-01-01 08:08:08\", \
↪"reverse\":\"false\"}"
```

2. Input via file: You could store the content of one parameter into a file and pass it via the command line with prefix `file://`:

```
> aliyun log get_logs --request="file://./get_logs.json"
```

the content in file `get_logs.json` as below. Note: the \ is unnecessary to escape the ".

```
{
  "topic": "",
  "logstore": "logstore1",
  "project": "project1",
  "toTime": "2018-01-01 11:11:11",
  "offset": "0",
  "query": "*",
  "line": "10",
  "fromTime": "2018-01-01 10:10:10",
  "reverse": "true"
}
```

1.5.2 Parameter Validation

- Mandatory check: if one mandatory parameter is missed, it will report error with usage info.
- Format of parameter's value will be validated. e.g. int, bool, string list, special data structure.
- for boolean, it support:
- true (case insensitive), T, 1
- false (case insensitive), F, 0
- String list support as ["s1", "s2"]

1.5.3 Output

1. For operations like Create, Update and Delete, there's no output except the exit code is 0 which means success.
2. For operations like Get and List, it will output in **json** format.
3. For errors, it will report in json format as below:

```
{
  "errorCode":"...",
  "errorMessage":"..."
}
```

1.5.4 Filter output

It's supported to filter output via JMES:

Examples:

```
> aliyun log get_logs ...
```

which outputs:

```
[ {"__source__": "ip1", "key": "log1"}, {"__source__": "ip2", "key": "log2"} ]
```

You could use below `--jmes-filter` to break log into each line:

```
> aliyun log get_logs ... --jmes-filter="join('\n', map(&to_string(@), @))"
```

output:

```
{"__source__": "ip1", "key": "log1"}
{"__source__": "ip2", "key": "log2"}
```

1.5.5 Further Process

You could use >> to store the output to a file. or you may want to process the output using your own cmd. For example, there's another way to if you may want to break the logs into each line. you could append thd command with a `|` on linux/unix:

```
| python2 -c "from __future__ import print_function;import json;map(lambda x:␣
↪print(json.dumps(x).encode('utf8')), json.loads(raw_input()));"
or
| python3 -c "import json;list(map(lambda x: print(json.dumps(x)), json.
↪loads(input())));"
```

e.g.

```
aliyun log get_log .... | | python2 -c "from __future__ import print_function;import␣
↪json;map(lambda x: print(json.dumps(x).encode('utf8')), json.loads(raw_input()));" >
↪> data.txt
```

1.6 Command Reference

1.6.1 Command Specification

```
1. aliyun log <subcommand> [parameters | global options]
2. aliyun configure <access_id> <access-key> <endpoint>
3. aliyun [--help | --version]
```

1.6.2 Alias

There's also an alias `aliyunlog` for the CLI in case the command `aliyun` conflicts with others.

```
1. aliyunlog log <subcommand> [parameters | global options]
2. aliyunlog configure <access_id> <access-key> <endpoint>
3. aliyunlog [--help | --version]
```

1.6.3 Subcommand and parameters

Actually, the CLI leverage `aliyun-log-python-sdk`, which maps the command into the methods of `aliyun.log.LogClient`. The parameters of command line is mapped to the parameters of methods. For the detail spec of parameters, please refer to the Mapped Python SDK API Spec

Examples:

```
def create_logstore(self, project_name, logstore_name, ttl=2, shard_count=30):
```

Mapped to CLI:

```
> aliyun log create_logstore
  --project_name=<value>
  --logstore_name=<value>
  [--ttl=<value>]
  [--shard_count=<value>]
```

1.6.4 Global options

All the commands support below optional global options:

```
[--access-id=<value>]
[--access-key=<value>]
[--region-endpoint=<value>]
[--client-name=<value>]
[--jmes-filter=<value>]
```

1.6.5 Command categories

1. *Project management*
2. *Logstore management*
3. *Shard management*
4. *Machine group management*
5. *Logtail config management*
6. *Machine group and Logtail Config Mapping*
7. *Index management*
8. *Cursor management*
9. *Logs write and consume*
10. *Shipper management*
11. *Consumer group management*

1. Project management

- list_project
- create_project
- get_project
- delete_project
- **copy_project**
- copy all configurations including logstore, logtail, and index config from project to another project which could be in different region.

```
> aliyun log copy_project --from_project="p1" --to_project="p1" --to_client="account2"
```

- Note: `to_client` is another account configured via `aliyun configure`, it's OK to pass `main` or not to copy inside the same region.
- Refer to Copy project settings cross regions to learn more.

2. Logstore management

- create_logstore
- delete_logstore
- get_logstore
- update_logstore
- list_logstore

3. Shard management

- list_shards
- split_shard
- merge_shard

4. Machine group management

- create_machine_group
- Format of partial parameter:

```
{
 "machine_list": [
   "machine1",
   "machine2"
 ],
 "machine_type": "userdefined",
 "group_name": "group_name2",
 "group_type": "Armory",
 "group_attribute": {
   "externalName": "ex name",
   "groupTopic": "topic x"
 }
}
```

- delete_machine_group
- update_machine_group
- get_machine_group

- list_machine_group
- list_machines

5. Logtail config management

- create_logtail_config
- LogtailLogtail.
- update_logtail_config
- delete_logtail_config
- get_logtail_config
- list_logtail_config

6. Machine group and Logtail Config Mapping

- apply_config_to_machine_group
- remove_config_to_machine_group
- get_machine_group_applied_configs
- get_config_applied_machine_groups

7. Index management

- create_index
- Format of partial parameter:

```
{
 "keys": {
   "f1": {
     "caseSensitive": false,
     "token": [
       ",",
       " ",
       "\"",
       "\"",
       ";",
       "=",
       "(",
       ")",
       "[",
       "]",
       "{",
       "}",
       "?",
       "@",
       "&",
       "<",
       ">",
       "/",
       ":",
       "\n",
       "\t"
     ],
     "type": "text",
     "doc_value": true
   },
```

```
    "f2": {
      "doc_value": true,
      "type": "long"
    }
  },
  "storage": "pg",
  "ttl": 2,
  "index_mode": "v2",
  "line": {
    "caseSensitive": false,
    "token": [
      ",",
      " ",
      "\"",
      "\"",
      ";",
      "=",
      "(",
      ")",
      "[",
      "]",
      "{",
      "}",
      "?",
      "@",
      "&",
      "<",
      ">",
      "/",
      ":",
      "\n",
      "\t"
    ]
  }
}
```

- update_index
- delete_index
- get_index_config
- list_topics

8. Cursor management

- get_cursor
- get_cursor_time
- get_previous_cursor_time
- get_begin_cursor
- get_end_cursor

9. Logs write and consume

- put_logs
- Format of parameter:

```
{
"project": "dlq-test-cli-35144",
"logstore": "logstore1",
"topic": "topic1",
"source": "source1",
"logtags": [
  [
    "tag1",
    "v1"
  ],
  [
    "tag2",
    "v2"
  ]
],
"hashKey": "1231231234",
"logitems": [
  {
    "timestamp": 1510579341,
    "contents": [
      [
        "key1",
        "v1"
      ],
      [
        "key2",
        "v2"
      ]
    ]
  },
  {
    "timestamp": 1510579341,
    "contents": [
      [
        "key3",
        "v3"
      ],
      [
        "key4",
        "v4"
      ]
    ]
  }
]
}
```

- get_logs
- Format of parameter:

```
{
"topic": "",
"logstore": "logstore1",
"project": "dlq-test-cli-35144",
"toTime": "2018-01-01 11:11:11",
"offset": "0",
"query": "*",
"line": "10",
```

```
"fromTime": "2018-01-01 10:10:10",
"reverse": "true"
}
```

- It will fetch all data when `line` is passed as -1. But if have large volume of data exceeding 1GB, better to use `get_log_all`
- get_log_all
- this API is similar as `get_logs`, but it will fetch data iteratively and output them by chunk. It's used for large volume of data fetching.
- get_histograms
- pull_logs
- pull_log
- this API is similar as `pull_logs`, but it allow readable parameter and allow to fetch data iteratively and output them by chunk. It's used for large volume of data fetching.

10. Shipper management

- create_shipper
- Format of partial parameter:

```
{
"oss_bucket": "dlq-oss-test1",
"oss_prefix": "sls",
"oss_role_arn": "acs:ram::1234:role/aliyunlogdefaultrole",
"buffer_interval": 300,
"buffer_mb": 128,
"compress_type": "snappy"
}
```

- update_shipper
- delete_shipper
- get_shipper_config
- list_shipper
- get_shipper_tasks
- retry_shipper_tasks

11. Consumer group management

- create_consumer_group
- update_consumer_group
- delete_consumer_group
- list_consumer_group
- update_check_point
- get_check_point

1.6.6 Best Practice

- Create Logtail Config
- Duplicate project settings cross region
- Pull Logs

1.6.7 Troubleshooting

By default, CLI store erros or warnings at `~/aliyunlogcli.log`, it's also configurable via file ~/.aliyunlogcli, section `__loggging__` to adjust the logging level and location:

```
[__logging__]
filename=  # default: ~/aliyunlogcli.log
filemode=  # default: a, could also be: w, a
format=    # default: %(asctime)s %(levelname)s %(filename)s:%(lineno)d %(funcName)s
↪%(message)s
datefmt=   # default: "%Y-%m-%d %H:%M:%S", could be strftime() compitable date/time␣
↪formatting string
level=     # default: warn, could be: info, error, fatal, critical, debug
```

1.7 Other resources

1. Alicloud Log Service homepagehttps://www.alibabacloud.com/product/log-service
2. Alicloud Log Service dochttps://www.alibabacloud.com/help/product/28958.htm
3. Alicloud Log Python SDK doc: http://aliyun-log-python-sdk.readthedocs.io/
4. for any issues, please submit support tickets

CHAPTER 2

README in English

2.1

-
  -
  -
-
  -
  -
  -
  -
-
  -
  -
  -
-
  -
  -
  -
  -
  -

-
  -
  -
  -
  -
  -
    -
    -
    -
    -
    - *Logtail*
    - *Logtail*
    -
    -
    -
    -
    -
-
-
-

(SLS)WebSDKSLS(Command Line Interface - CLI)

CLI

- REST
-
-
-
-
- JMES
- Windows/LinuxMacPythonPy2.6+Py3.3+Pip

CLI

- Windows
- Mac OS
- Linux

Python 2.62.73.33.43.53.6PyPyPyPyPy3

CLI

```
> pip install -U aliyun-log-cli
```

CLI

```
> aliyun --help
```

SDKIDKeyEndpointLogClient

CLI, : , , .

-

```
> aliyun log create_project ..... --access-id=<value> --access-key=<value> --region-
↪endpoint=<value>
```

: logAKEndpoint()

-
- ALIYUN_LOG_CLI_ACCESSID
- ALIYUN_LOG_CLI_ACCESSKEY
- ALIYUN_LOG_CLI_ENDPOINT
-

AKEndpoint~/.aliyunlogcli, `main`

```
[main]
access-id=
access-key=
region-endpoint=
```

Configure.

```
> aliyun configure access_id access_key cn-beijing.log.aliyuncs.com
```

1. , ():

```
> aliyun configure access_id1 access_key1 cn-beijing.log.aliyuncs.com
> aliyun configure access_id2 access_key2 cn-hangzhou.log.aliyuncs.com test
```

AK:

```
[main]
access-id=access_id1
access-key=access_key1
region-endpoint=cn-beijing.log.aliyuncs.com

[test]
access-id=access_id2
access-key=access_key2
region-endpoint=cn-hangzhou.log.aliyuncs.com
```

2.

`--client-name=<value>`, :

```
> aliyun log create_project ..... --client-name=test
```

testAK.

3.

,:

```
> aliyun log copy_project --from_project="p1" --to_project="p1" --to_client=test
```

mainp1testp1

1.

```
> aliyun log get_logs --request="{\"topic\": \"\", \"logstore\": \"logstore1\", \
↪"project\": \"dlq-test-cli-123\", \"toTime\": \"2018-01-01 11:11:11\", \"offset\": \
↪"0\", \"query\": \"*\", \"line\": \"10\", \"fromTime\": \"2018-01-01 10:10:10\", \
↪"reverse\":\"false\"}"
```

2.

,, file://+:

```
> aliyun log get_logs --request="file://./get_logs.json"
```

get_logs.json,: \.

```
{
  "topic": "",
  "logstore": "logstore1",
  "project": "project1",
  "toTime": "2018-01-01 11:11:11",
  "offset": "0",
  "query": "*",
  "line": "10",
  "fromTime": "2018-01-01 10:10:10",
  "reverse": "true"
}
```

- ,
- , int, bool, string list,
- bool:
- true (), T, 1
- false (), F, 0
- [“s1”, “s2”]

1. Create, Update, Delete, , exit code=0.
2. Get/List, json
3. ,:

```
{
  "errorCode":"...",
  "errorMessage":"..."
}
```

JMES.

:

```
> aliyun log get_logs ...
```

:

```
[ {"__source__": "ip1", "key": "log1"}, {"__source__": "ip2", "key": "log2"} ]
```

:

```
> aliyun log get_logs ... --jmes-filter="join('\n', map(&to_string(@), @))"
```

:

```
{"__source__": "ip1", "key": "log1"}
{"__source__": "ip2", "key": "log2"}
```

>>. , , , json. Linux/Unix, |.

```
| python2 -c "from __future__ import print_function;import json;map(lambda x:␣
↪print(json.dumps(x).encode('utf8')), json.loads(raw_input()));"
or
| python3 -c "import json;list(map(lambda x: print(json.dumps(x)), json.
↪loads(input())));"
```

:

```
aliyun log get_log .... | | python2 -c "from __future__ import print_function;import␣
↪json;map(lambda x: print(json.dumps(x).encode('utf8')), json.loads(raw_input()));" >
↪> data.txt
```

```
1. aliyun log <subcommand> [parameters | global options]
2. aliyun configure <access_id> <access-key> <endpoint> [<client-name>]
3. aliyun [--help | --version]
```

CLIaliyunlog, aliyun, aliyunlog:

```
1. aliyunlog log <subcommand> [parameters | global options]
2. aliyunlog configure <access_id> <access-key> <endpoint>
3. aliyunlog [--help | --version]
```

Python SDK, aliyun.log.LogClient, . API, Python SDK API

:

```
def create_logstore(self, project_name, logstore_name, ttl=2, shard_count=30):
```

:

```
> aliyun log create_logstore
  --project_name=<value>
  --logstore_name=<value>
  [--ttl=<value>]
  [--shard_count=<value>]
```

:

```
[--access-id=<value>]
[--access-key=<value>]
```

```
[--region-endpoint=<value>]
[--client-name=<value>]
[--jmes-filter=<value>]
```

1.
2.
3.
4.
5. *Logtail*
6. *Logtail*
7.
8.
9.
10.
11.

1.

- list_project
- create_project
- get_project
- delete_project
- **copy_project**
- projectlogstore, logtail, machine groupindexproject.

```
> aliyun log copy_project --from_project="p1" --to_project="p1" --to_client="account2"
```

- : `to_client`aliyun configure, `main`.
- .

2.

- create_logstore
- delete_logstore
- get_logstore
- update_logstore
- list_logstore

3.

- list_shards
- split_shard
- merge_shard

4.

- create_machine_group

- :

```
{
 "machine_list": [
   "machine1",
   "machine2"
 ],
 "machine_type": "userdefined",
 "group_name": "group_name2",
 "group_type": "Armory",
 "group_attribute": {
   "externalName": "ex name",
   "groupTopic": "topic x"
 }
}
```

- delete_machine_group
- update_machine_group
- get_machine_group
- list_machine_group
- list_machines

5. Logtail

- create_logtail_config
- LogtailLogtail.
- update_logtail_config
- delete_logtail_config
- get_logtail_config
- list_logtail_config

6. Logtail

- apply_config_to_machine_group
- remove_config_to_machine_group
- get_machine_group_applied_configs
- get_config_applied_machine_groups

7.

- create_index
- :

```
{
 "keys": {
   "f1": {
     "caseSensitive": false,
     "token": [
       ",",
       " ",
       "\"",
       "\"",
```

```
        ";",
        "=",
        "(",
        ")",
        "[",
        "]",
        "{",
        "}",
        "?",
        "@",
        "&",
        "<",
        ">",
        "/",
        ":",
        "\n",
        "\t"
      ],
      "type": "text",
      "doc_value": true
    },
    "f2": {
      "doc_value": true,
      "type": "long"
    }
  },
  "storage": "pg",
  "ttl": 2,
  "index_mode": "v2",
  "line": {
    "caseSensitive": false,
    "token": [
      ",",
      " ",
      "\"",
      "\"",
      ";",
      "=",
      "(",
      ")",
      "[",
      "]",
      "{",
      "}",
      "?",
      "@",
      "&",
      "<",
      ">",
      "/",
      ":",
      "\n",
      "\t"
    ]
  }
}
```

- update_index

- delete_index
- get_index_config
- list_topics

8.

- get_cursor
- get_cursor_time
- get_previous_cursor_time
- get_begin_cursor
- get_end_cursor

9.

- put_logs
- :

```
{
"project": "dlq-test-cli-35144",
"logstore": "logstore1",
"topic": "topic1",
"source": "source1",
"logtags": [
  [
    "tag1",
    "v1"
  ],
  [
    "tag2",
    "v2"
  ]
],
"hashKey": "1231231234",
"logitems": [
  {
    "timestamp": 1510579341,
    "contents": [
      [
        "key1",
        "v1"
      ],
      [
        "key2",
        "v2"
      ]
    ]
  },
  {
    "timestamp": 1510579341,
    "contents": [
      [
        "key3",
        "v3"
      ],
      [
        "key4",
```

```
        "v4"
      ]
    ]
  }
]
}
```

- get_logs
- :

```
{
"topic": "",
"logstore": "logstore1",
"project": "dlq-test-cli-35144",
"toTime": "2018-01-01 11:11:11",
"offset": "0",
"query": "*",
"line": "10",
"fromTime": "2018-01-01 10:10:10",
"reverse": "true"
}
```

- `line`-1, . , 1GB, `get_log_all`
- get_log_all
- `get_logs`, , .
- get_histograms
- pull_logs
- pull_log
- `pull_logs`, , .

10.

- create_shipper
- :

```
{
"oss_bucket": "dlq-oss-test1",
"oss_prefix": "sls",
"oss_role_arn": "acs:ram::1234:role/aliyunlogdefaultrole",
"buffer_interval": 300,
"buffer_mb": 128,
"compress_type": "snappy"
}
```

- update_shipper
- delete_shipper
- get_shipper_config
- list_shipper
- get_shipper_tasks
- retry_shipper_tasks

11.

- create_consumer_group
- update_consumer_group
- delete_consumer_group
- list_consumer_group
- update_check_point
- get_check_point
- Logtail
-
-

CLI~/aliyunlogcli.log, ~/.aliyunlogcli__loggging__:

```
[__logging__]
filename=  # : ~/aliyunlogcli.log
filemode=  # : a, : w, a
format=    # : %(asctime)s %(levelname)s %(filename)s:%(lineno)d %(funcName)s
↪%(message)s
datefmt=   # : "%Y-%m-%d %H:%M:%S", strftime()/
level=     # : warn, info, error, fatal, critical, debug
```

1. http://www.aliyun.com/product/sls/
2. https://help.aliyun.com/product/28958.html
3. Python SDK: http://aliyun-log-python-sdk.readthedocs.io/
4.

CHAPTER 3

3.1 Logtail

3.1.1

Logtail, . Logtail, CLILogtail.

Logtail

Logtail:

- :
-
- JSON
-
-
- syslog
-
- NGNIX,
-

3.1.2

CLI.

CLI, , . , .

```
> aliyun configure AKID****123 AKKEY****123 cn-hangzhou.log.aliyuncs.com
```

:

- ,
- Endpoint,

3.1.3

1. logtail

, content. .

project1logtail:

```
> aliyun log create_logtail_config --project_name="project1" --config_detail="file://
↪simple_1.json"
```

simple_1.json:

```
{
  "configName": "simple_1",
  "inputDetail": {
    "logType": "common_reg_log",
    "logPath": "/user",
    "filePattern": "*.log"
  },
  "inputType": "file",
  "outputDetail": {
    "logstoreName": "logstore1"
  }
}
```

logstore1simple_1logtail. logPathfilePattern. /user.log.

:,: inputTypefile, inputDetail.logTypecommon_reg_log.

,.

2. JSONlogtail

JSONJSON.

project1JSONlogtail:

```
> aliyun log create_logtail_config --project_name="project1" --config_detail="file://
↪json_1.json"
```

json_1.json:

```
{
  "configName": "json_1",
  "inputDetail": {
    "logType": "json_log",
    "filePattern": "*.json",
    "logPath": "/json_1"
  },
  "inputType": "file",
  "outputDetail": {
    "logstoreName": "logstore1"
  }
}
```

logstore1json_1logtail. logPathfilePattern. /user.json.

:

1. . inputTypefile, inputDetail.logTypejson_log.
2. Logtailjson. :

```
{
  "from": "nanjing",
  "to": "shanghai",
  "people": "xiaoming",
  "travel_time": "2018-1-1 10:10:10"
}
```

from, topeopletravel_time.

, , . , , logtailtravel_time:

```
{
  "inputDetail": {
    "timeFormat": "%Y-%M-%D %h:%m:s",
    "timeKey": "travel_time"
  }
}
```

inputDetail.timeKeytravel_time, inputDetail.timeFormat%Y-%M-%D %h:%m:s, ,

3. logtail

, CSV, TSV.

3, :

```
2017-1-1 10:10:00&#&nanjing&#&shanghai&#&xiao ming
2017-1-1 20:10:00&#&beijing&#&hangzhou&#&xiao wang
```

&#&, , , .

project1logtail:

```
> aliyun log create_logtail_config --project_name="project1" --config_detail="file://
↪sep_1.json"
```

sep_1.json:

```
{
  "configName": "sep_1",
  "logSample": "2017-1-1 10:10:00&#&nanjing&#&shanghai&#&xiao ming",
  "inputDetail": {
    "logType": "delimiter_log",
    "logPath": "/user",
    "filePattern": "travel.log",
    "separator": "&#&",
    "key": [
      "travel_time",
      "from_city",
      "to_city",
      "people"
    ]
  },
  "inputType": "file",
  "outputDetail": {
    "logstoreName": "logstore1"
  }
}
```

logstore1json_1logtail. :

- ./usertravel.json.
- &#&,,travel_time, from_city, to_citypeople.
- ,logSample.
- .inputTypefile, inputDetail.logTypedelimiter_log.

,,,&|,.

```
2017-1-1 10:10:00&nanjing|&shanghai|xiao ming
2017-1-1 20:10:00|beijing&hangzhou|xiao wang
```

,.

,,.,, logtailtravel_time:

```
{
  "inputDetail": {
    "timeFormat": "%Y-%M-%D %h:%m:s",
    "timeKey": "travel_time"
  }
}
```

travel_time, %Y-%M-%D %h:%m:s,,

,separator:

	"
	\ \
	\/
	\b
	\f
	\n
	\r
	\t
	\v
\u 0000	

4. logtail

,..

NGNIX:

```
10.1.1.1 - - [2018-1-1 10:10:10] "GET / HTTP/1.1" 0.011 180 404 570 "-" "Mozilla/4.0␣
↪(compatible; MSIE 6.0; Windows NT 5.1; 360se)"
10.1.1.1 - - [2018-1-1 10:10:20] "GET / HTTP/1.1" 0.011 180 404 570 "-" "Mozilla/4.0␣
↪(compatible; MSIE 6.0; Windows NT 5.1; 360se)"
```

: IP - - [] "HTTP" HTTP "-" "" : (\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}) - - \[([^\]+])\] "([\w\s\/\.]+) ([\d\.]+) ([\d\.]+) ([\d\.]+) ([\d\.]+) "-" "[^\"]+"

project1logtail:

```
> aliyun log create_logtail_config --project_name="project1" --config_detail="file://
↪reg_1.json"
```

reg_1.json:

```
{
  "configName": "ngnix_1",
  "logSample": "10.1.1.1 - - [13/Mar/2016:10:00:10 +0800] \"GET / HTTP/1.1\" 0.011␣
↪180 404 570 \"-\" \"Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; 360se)\"",
  "inputDetail": {
    "logType": "common_reg_log",
    "logPath": "/ngnix",
    "filePattern": "*.log",
    "regex": "(\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}) - - \\[([^\\]+])\\] \
↪"([\\w\\s\/\\.]+) ([\\d\\.]+) ([\\d\\.]+) ([\\d\\.]+) ([\\d\\.]+) \"-\" \"[^\\\"]+\"
↪",
    "key": [
      "client_ip",
      "request_time",
      "method_type",
      "process_period",
      "request_bytes",
```

```
      "http_status",
      "response_bytes",
      "client_agent"
    ],
    "timeFormat": "%Y-%M-%D %h:%m:%s"
  },
  "inputType": "file",
  "outputDetail": {
    "logstoreName": "logstore1"
  }
}
```

logstore1ngnix_1logtail. :

- . /ngnix*.log.
- regex, key.
- logSample.
- . inputTypefile, inputDetail.logTypedelimiter_log.

, , . , NGNIX, logtailrequest_time:

```
{
  "inputDetail": {
    "timeKey": "request_time",
    "timeFormat": "%Y-%M-%D %h:%m:s"
  }
}
```

inputDetail.timeFormat%Y-%M-%D %h:%m:s, ,

: inputDetail.timeFormatinputDetail.timeKey, time.

5. sysloglogtail

syslog, . CLI, logtail. .

project1sysloglogtail:

```
> aliyun log create_logtail_config --project_name="project1" --config_detail="file://
↪syslog_1.json"
```

syslog_1.json:

```
{
  "configName": "syslog_1",
  "inputDetail": {
    "tag": "tag1"
  },
  "inputType": "syslog",
  "outputDetail": {
    "logstoreName": "logstore1"
  }
}
```

`logstore1syslog_1`logtail. :

- `tag`.
- . `inputTypesyslog`, `inputDetail.logType`.

syslog, CLI, logtail. . , syslog.

3.1.4

, Java. . **:** .

:

```
com.journaldev.log.LoggingExample::main::2017-1-1 01:42:43::Msg977
com.journaldev.log.LoggingExample::main::2017-1-1 03:42:43::Break at
  File "/Applications/PyCharm CE.app/Contents/helpers/pydev/pydevd.py", line 1599, in
↪<module>
    globals = debugger.run(setup['file'], None, None, is_module)
  File "/Applications/PyCharm CE.app/Contents/helpers/pydev/pydevd.py", line 1026, in␣
↪run
    pydev_imports.execfile(file, globals, locals)  # execute the script
  File "/Users/wjo1212/GithubProjects/aliyun-log-cli/aliyunlogcli/cli.py", line 5, in
↪<module>
    main()
com.journaldev.log.LoggingExample::main::2017-1-1 05:42:43::Msg978
```

`logBeginRegex`:

```
{
  "inputDetail": {
    "logBeginRegex": "com.journaldev.log.LoggingExample::main::.*"
  }
}
```

`inputDetail.logBeginRegexcom.journaldev.log.LoggingExample::main::.*`.

: `.*`. , :

```
{
  "inputDetail": {
    "logBeginRegex": "com.journaldev.log.LoggingExample::main::"
  }
}
```

Logtail

Logtail, , :

- (localStorage):

  - 1GB
  - (true)
- (enableRawLog)
  -
  - (false)
- Topic (topicFormat)
  - :
    - (none): .
    - (group_topic): Topic.
    - : Topic
  - ,
  - -(none)
- (fileEncoding)
  - utf8gbk
  - utf8
- (maxDepth)
  - 0-10000
  - 100
- (preserve)
  - - (true):
    - 30(false): 30
  - (true)
- (preserveDepth)
  - 30, , 1-3
- -
  - filterKeyfilterRegex.

, , , Topic, gbk, 200, 30, 3. from_cityto_city..

```
{
  "inputDetail": {
    "localStorage": false,
    "enableRawLog": true,
    "topicFormat": "/user.+/(\\w+).log",
    "fileEncoding": "gbk",
    "maxDepth": 200,
    "preserve": false,
    "preserveDepth": 3,
    "filterKey": [
      "from_city",
      "to_city"
    ],
    "filterRegex": [
```

```
      "nanjing|shanghai",
      "beijing|hangzhou"
    ]
  }
}
```

3.1.5

Logtail, , *apply_config_to_machine_group*.

3.2

3.2.1

(Region)(Project), . . CLI.

3.2.2

:

- (logstore),
-
- logtail

, , .

-
- logtail

: CLI, .

3.2.3

CLI.

1.

CLI, , . , .

2, , .

```
> aliyun configure AKID****123 AKKEY****123 cn-hangzhou.log.aliyuncs.com
> aliyun configure AKID****123 AKKEY****123 cn-beijing.log.aliyuncs.com bj
```

: , , . `main`, , .

:

- ,
- Endpoint,

2.

project1,:

```
> aliyun log copy_project --from_project="project1" --to_project="project1" --to_
↪client=bj
```

CLImainproject1,bjproject1.

3.

,.

bjproject1project2:

```
> aliyun log copy_project --from_project="project1" --to_project="project2" --copy_
↪machine_group=true --client-name=bj
```

: copy_machine_grouptrueLogtail.

3.3

3.4

3.5 JMES

CHAPTER 4

Mapped SDK API

4.1 Request and Config Class

GetHistogramsRequest([project, logstore, ...])	The request used to get histograms of a query from log.
GetLogsRequest([project, logstore, ...])	The request used to get logs by a query from log.
GetProjectLogsRequest([project, query])	The request used to get logs by a query from log cross multiple logstores.
ListTopicsRequest([project, logstore, ...])	The request used to get topics of a query from log.
ListLogstoresRequest([project])	The request used to list log store from log.
PutLogsRequest([project, logstore, topic, ...])	The request used to send data to log.
LogtailConfigGenerator	Generator of Logtial config
SeperatorFileConfigDetail(logstoreName, ...)	The logtail config for separator mode
SimpleFileConfigDetail(logstoreName, ...[, ...])	The logtail config for simple mode
FullRegFileConfigDetail(logstoreName, ...[, ...])	The logtail config for full regex mode
JsonFileConfigDetail(logstoreName, ...[, ...])	The logtail config for json mode
ApsaraFileConfigDetail(logstoreName, ...[, ...])	The logtail config for Apsara mode
SyslogConfigDetail(logstoreName, configName, tag)	The logtail config for syslog mode
MachineGroupDetail(group_name, machine_type, ...)	The machine group detail info
IndexConfig([ttl, line_config, ...])	The index config of a logstore
OssShipperConfig(oss_bucket, oss_prefix, ...)	A oss ship config
OdpsShipperConfig(odps_endpoint, ...[, ...])	Odps shipper config
ShipperTask(task_id, task_status, ...)	A shipper task

4.2 Project

list_project([offset, size])	list the project
create_project(project_name, project_des)	Create a project Unsuccessful opertaion will cause an LogException.
get_project(project_name)	get project
delete_project(project_name)	delete project
copy_project(from_project, to_project[, ...])	copy project, logstore, machine group and logtail config to target project,

4.3 Logstore

list_logstore(project_name[, ...])	list the logstore in a projectListLogStoreResponse
create_logstore(project_name, logstore_name, ...)	create log store
get_logstore(project_name, logstore_name)	get the logstore meta info
update_logstore(project_name, logstore_name, ttl)	update the logstore meta info
delete_logstore(project_name, logstore_name)	delete log store
list_topics(request)	List all topics in a logstore.

4.4 Index

create_index(project_name, logstore_name, ...)	create index for a logstore
update_index(project_name, logstore_name, ...)	update index for a logstore
delete_index(project_name, logstore_name)	delete index of a logstore
get_index_config(project_name, logstore_name)	get index config detail of a logstore

4.5 Logtail Config

create_logtail_config(project_name, ...)	create logtail config in a project
update_logtail_config(project_name, ...)	update logtail config in a project
delete_logtail_config(project_name, config_name)	delete logtail config in a project
get_logtail_config(project_name, config_name)	get logtail config in a project
list_logtail_config(project_name[, offset, size])	list logtail config name in a project

4.6 Machine Group

create_machine_group(project_name, group_detail)	create machine group in a project
delete_machine_group(project_name, group_name)	delete machine group in a project
update_machine_group(project_name, group_detail)	update machine group in a project
get_machine_group(project_name, group_name)	get machine group in a project
list_machine_group(project_name[, offset, size])	list machine group names in a project
list_machines(project_name, group_name[, ...])	list machines in a machine group

4.7 Apply Logtail Config

apply_config_to_machine_group(project_name, ...)	apply a logtail config to a machine group
remove_config_to_machine_group(project_name, ...)	remove a logtail config to a machine group
get_machine_group_applied_configs(...)	get the logtail config names applied in a machine group
get_config_applied_machine_groups(...)	get machine group names where the logtail config applies to

4.8 Shard

list_shards(project_name, logstore_name)	list the shard meta of a logstore
split_shard(project_name, logstore_name, ...)	split a readwrite shard into two shards
merge_shard(project_name, logstore_name, shardId)	split two adjacent readwrite hards into one shards

4.9 Cursor

get_cursor(project_name, logstore_name, ...)	Get cursor from log service for batch pull logs Unsuccessful opertaion will cause an LogException.
get_cursor_time(project_name, logstore_name, ...)	Get cursor time from log service Unsuccessful opertaion will cause an LogException.
get_previous_cursor_time(project_name, ...)	Get previous cursor time from log service.
get_begin_cursor(project_name, ...)	Get begin cursor from log service for batch pull logs Unsuccessful opertaion will cause an LogException.
get_end_cursor(project_name, logstore_name, ...)	Get end cursor from log service for batch pull logs Unsuccessful opertaion will cause an LogException.

4.10 Logs

put_logs(request)	Put logs to log service.
pull_logs(project_name, logstore_name, ...)	batch pull log data from log service
pull_log	
get_log(project, logstore, from_time, to_time)	Get logs from log service.
get_logs(request)	Get logs from log service.
get_log_all	
get_histograms(request)	Get histograms of requested query from log service.
get_project_logs(request)	Get logs from log service.

4.11 Consumer group

create_consumer_group(project, logstore, ...)	create consumer group
update_consumer_group(project, logstore, ...)	Update consumer group
delete_consumer_group(project, logstore, ...)	Delete consumer group

Continued on next page

Table 4.11 – continued from previous page

list_consumer_group(project, logstore)	List consumer group
update_check_point(project, logstore, ... [, ...])	Update check point
get_check_point(project, logstore, ... [, shard])	Get check point

4.12 Shipper

create_shipper(project_name, logstore_name, ...)	create odps/oss shipper
update_shipper(project_name, logstore_name, ...)	update odps/oss shipper
delete_shipper(project_name, logstore_name, ...)	delete odps/oss shipper
get_shipper_config(project_name, ...)	get odps/oss shipper
list_shipper(project_name, logstore_name)	list odps/oss shipper
get_shipper_tasks(project_name, ... [, ...])	get odps/oss shipper tasks in a certain time range
retry_shipper_tasks(project_name, ...)	retry failed tasks , only the failed task can be retried

4.13 Definitions

class aliyun.log.**LogClient**(*endpoint*, *accessKeyId*, *accessKey*, *securityToken=None*, *source=None*)

Construct the LogClient with endpoint, accessKeyId, accessKey.

Parameters

- **endpoint** (*string*) – log service host name, for example, http://ch-hangzhou.sls.aliyuncs.com
- **accessKeyId** (*string*) – aliyun accessKeyId
- **accessKey** (*string*) – aliyun accessKey

apply_config_to_machine_group(*project_name*, *config_name*, *group_name*)

apply a logtail config to a machine group Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **config_name** (*string*) – the logtail config name to apply
- **group_name** (*string*) – the machine group name

Returns ApplyConfigToMachineGroupResponse

Raise LogException

copy_project(*from_project*, *to_project*, *to_client=None*, *copy_machine_group=False*)

copy project, logstore, machine group and logtail config to target project, expecting the target project doesn't contain same named logstores as source project

Parameters

- **from_project** (*string*) – project name
- **to_project** (*string*) – project name
- **to_client** (LogClient) – logclient instance
- **copy_machine_group** (*bool*) – if copy machine group resources, False by default.

Returns None

create_consumer_group(*project*, *logstore*, *consumer_group*, *timeout*, *in_order=False*)
create consumer group

Parameters

- **project** (*string*) – project name
- **logstore** (*string*) – logstore name
- **consumer_group** (*string*) – consumer group name
- **timeout** (*int*) – time-out
- **in_order** (*bool*) – if consume in order, default is False

Returns CreateConsumerGroupResponse

create_index(*project_name*, *logstore_name*, *index_detail*)
create index for a logstore Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **index_detail** (IndexConfig) – the index config detail used to create index

Returns CreateIndexResponse

Raise LogException

create_logstore(*project_name*, *logstore_name*, *ttl*, *shard_count*)
create log store Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **ttl** (*int*) – the life cycle of log in the logstore in days
- **shard_count** (*int*) – the shard count of the logstore to create

Returns CreateLogStoreResponse

Raise LogException

create_logtail_config(*project_name*, *config_detail*)
create logtail config in a project Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **config_detail** (LogtailConfigGenerator *or* SeperatorFileConfigDetail *or* SimpleFileConfigDetail *or* FullRegFileConfigDetail *or* JsonFileConfigDetail *or* ApsaraFileConfigDetail *or* SyslogConfigDetail *or* *CommonRegLogConfigDetail*) – the logtail config detail info, use *LogtailConfigGenerator.from_json* to generate config: SeperatorFileConfigDetail or SimpleFileConfigDetail or FullRegFileConfigDetail or JsonFileConfigDetail or ApsaraFileConfigDetail or SyslogConfigDetail, Note: CommonRegLogConfigDetail is deprecated.

Returns CreateLogtailConfigResponse

Raise LogException

create_machine_group(*project_name*, *group_detail*)
create machine group in a project Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **group_detail** (MachineGroupDetail) – the machine group detail config

Returns CreateMachineGroupResponse

Raise LogException

create_project(*project_name*, *project_des*)
Create a project Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **project_des** (*string*) – the description of a project

Returns CreateProjectResponse

Raise LogException

create_shipper(*project_name*, *logstore_name*, *shipper_name*, *shipper_type*, *shipper_config*)
create odps/oss shipper for every type, it only allowed one shipper Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shipper_name** (*string*) – the shipper name
- **shipper_type** (*string*) – only support "odps" or "oss"
- **shipper_config** (OssShipperConfig *or* OdpsShipperConfig) – the detail shipper config, must be OssShipperConfig or OdpsShipperConfig type

Returns CreateShipperResponse

Raise LogException

delete_consumer_group(*project*, *logstore*, *consumer_group*)
Delete consumer group

Parameters

- **project** (*string*) – project name
- **logstore** (*string*) – logstore name
- **consumer_group** (*string*) – consumer group name

Returns None

delete_index(*project_name*, *logstore_name*)
delete index of a logstore Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name

Returns DeleteIndexResponse

Raise LogException

delete_logstore(*project_name*, *logstore_name*)
delete log store Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name

Returns DeleteLogStoreResponse

Raise LogException

delete_logtail_config(*project_name*, *config_name*)
delete logtail config in a project Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **config_name** (*string*) – the logtail config name

Returns DeleteLogtailConfigResponse

Raise LogException

delete_machine_group(*project_name*, *group_name*)
delete machine group in a project Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **group_name** (*string*) – the group name

Returns DeleteMachineGroupResponse

Raise LogException

delete_project(*project_name*)
delete project Unsuccessful opertaion will cause an LogException.

Parameters **project_name** (*string*) – the Project name

Returns DeleteProjectResponse

Raise LogException

delete_shard(*project_name*, *logstore_name*, *shardId*)
delete a readonly shard Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shardId** (*int*) – the read only shard id

Returns ListShardResponse

Raise LogException

delete_shipper(*project_name*, *logstore_name*, *shipper_name*)
delete odps/oss shipper Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shipper_name** (*string*) – the shipper name

Returns DeleteShipperResponse

Raise LogException

get_begin_cursor(*project_name*, *logstore_name*, *shard_id*)
Get begin cursor from log service for batch pull logs Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shard_id** (*int*) – the shard id

Returns GetLogsResponse

Raise LogException

get_check_point(*project*, *logstore*, *consumer_group*, *shard=-1*)
Get check point

Parameters

- **project** (*string*) – project name
- **logstore** (*string*) – logstore name
- **consumer_group** (*string*) – consumer group name
- **shard** (*int*) – shard id

Returns ConsumerGroupCheckPointResponse

get_check_point_fixed(*project*, *logstore*, *consumer_group*, *shard=-1*)
Get check point

Parameters

- **project** (*string*) – project name
- **logstore** (*string*) – logstore name
- **consumer_group** (*string*) – consumer group name
- **shard** (*int*) – shard id

Returns ConsumerGroupCheckPointResponse

get_config_applied_machine_groups(*project_name*, *config_name*)
get machine group names where the logtail config applies to Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **config_name** (*string*) – the logtail config name used to apply

Returns GetConfigAppliedMachineGroupsResponse

Raise LogException

get_cursor(*project_name*, *logstore_name*, *shard_id*, *start_time*)
Get cursor from log service for batch pull logs Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shard_id** (*int*) – the shard id
- **start_time** (*string/int*) – the start time of cursor, e.g 1441093445 or "begin"/"end", or "%Y-%m-%d %H:%M:%S"

Returns GetCursorResponse

Raise LogException

get_cursor_time(*project_name*, *logstore_name*, *shard_id*, *cursor*)
Get cursor time from log service Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shard_id** (*int*) – the shard id
- **cursor** (*string*) – the cursor to get its service receive time

Returns GetCursorTimeResponse

Raise LogException

get_end_cursor(*project_name*, *logstore_name*, *shard_id*)
Get end cursor from log service for batch pull logs Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shard_id** (*int*) – the shard id

Returns GetLogsResponse

Raise LogException

get_histograms(*request*)
Get histograms of requested query from log service. Unsuccessful opertaion will cause an LogException.

Parameters **request** (GetHistogramsRequest) – the GetHistograms request parameters class.

Returns GetHistogramsResponse

Raise LogException

get_index_config(*project_name*, *logstore_name*)
get index config detail of a logstore Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name

Returns GetIndexResponse

Raise LogException

get_log(*project*, *logstore*, *from_time*, *to_time*, *topic=None*, *query=None*, *reverse=False*, *offset=0*, *size=100*)

Get logs from log service. will retry when incomplete. Unsuccessful opertaion will cause an LogException.

Parameters

- **project** (*string*) – project name
- **logstore** (*string*) – logstore name
- **from_time** (*int/string*) – the begin timestamp or format of time in format "%Y-%m-%d %H:%M:%S" e.g. "2018-01-02 12:12:10"
- **to_time** (*int/string*) – the end timestamp or format of time in format "%Y-%m-%d %H:%M:%S" e.g. "2018-01-02 12:12:10"
- **topic** (*string*) – topic name of logs, could be None
- **query** (*string*) – user defined query, could be None
- **reverse** (*bool*) – if reverse is set to true, the query will return the latest logs first, default is false
- **offset** (*int*) – line offset of return logs
- **size** (*int*) – max line number of return logs, -1 means get all

Returns GetLogsResponse

Raise LogException

get_logs(*request*)

Get logs from log service. Unsuccessful opertaion will cause an LogException.

Parameters **request** (GetLogsRequest) – the GetLogs request parameters class.

Returns GetLogsResponse

Raise LogException

get_logstore(*project_name*, *logstore_name*)

get the logstore meta info Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name

Returns GetLogStoreResponse

Raise LogException

get_logtail_config(*project_name*, *config_name*)

get logtail config in a project Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **config_name** (*string*) – the logtail config name

Returns GetLogtailConfigResponse

Raise LogException

get_machine_group(*project_name*, *group_name*)
get machine group in a project Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **group_name** (*string*) – the group name to get

Returns GetMachineGroupResponse

Raise LogException

get_machine_group_applied_configs(*project_name*, *group_name*)
get the logtail config names applied in a machine group Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **group_name** (*string*) – the group name list

Returns GetMachineGroupAppliedConfigResponse

Raise LogException

get_previous_cursor_time(*project_name*, *logstore_name*, *shard_id*, *cursor*, *normalize=True*)
Get previous cursor time from log service. Note: normalize = true: if the cursor is out of range, it will be nornalized to nearest cursor Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shard_id** (*int*) – the shard id
- **cursor** (*string*) – the cursor to get its service receive time
- **normalize** (*bool*) – fix the cursor or not if it's out of scope

Returns GetCursorTimeResponse

Raise LogException

get_project(*project_name*)
get project Unsuccessful opertaion will cause an LogException.

Parameters **project_name** (*string*) – the Project name

Returns GetProjectResponse

Raise LogException

get_project_logs(*request*)
Get logs from log service. Unsuccessful opertaion will cause an LogException.

Parameters **request** (GetProjectLogsRequest) – the GetProjectLogs request parameters class.

Returns GetLogsResponse

Raise LogException

get_shipper_config(*project_name*, *logstore_name*, *shipper_name*)
get odps/oss shipper Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shipper_name** (*string*) – the shipper name

Returns GetShipperConfigResponse

Raise LogException

get_shipper_tasks(*project_name*, *logstore_name*, *shipper_name*, *start_time*, *end_time*, *status_type=''*, *offset=0*, *size=100*)
get odps/oss shipper tasks in a certain time range Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shipper_name** (*string*) – the shipper name
- **start_time** (*int*) – the start timestamp
- **end_time** (*int*) – the end timestamp
- **status_type** (*string*) – support one of ['', 'fail', 'success', 'running'] , if the status_type = '' , return all kinds of status type
- **offset** (*int*) – the begin task offset, -1 means all
- **size** (*int*) – the needed tasks count

Returns GetShipperTasksResponse

Raise LogException

heart_beat(*project*, *logstore*, *consumer_group*, *consumer*, *shards=None*)
Heatbeat consumer group

Parameters

- **project** (*string*) – project name
- **logstore** (*string*) – logstore name
- **consumer_group** (*string*) – consumer group name
- **consumer** (*string*) – consumer name
- **shards** (*int list*) – shard id list e.g. [0,1,2]

Returns None

list_consumer_group(*project*, *logstore*)
List consumer group

Parameters

- **project** (*string*) – project name
- **logstore** (*string*) – logstore name

Returns ListConsumerGroupResponse

list_logstore(*project_name*, *logstore_name_pattern=None*, *offset=0*, *size=100*)
list the logstore in a projectListLogStoreResponse Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name_pattern** (*string*) – the sub name logstore, used for the server to return logstore names contain this sub name
- **offset** (*int*) – the offset of all the matched names
- **size** (*int*) – the max return names count, -1 means all

Returns ListLogStoreResponse

Raise LogException

list_logstore_acl(*project_name*, *logstore_name*, *offset=0*, *size=100*)
list acl of a logstore Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **offset** (*int*) – the offset of all acl
- **size** (*int*) – the max return acl count

Returns ListAclResponse

Raise LogException

list_logstores(*request*)
List all logstores of requested project. Unsuccessful opertaion will cause an LogException.

Parameters **request** (ListLogstoresRequest) – the ListLogstores request parameters class.

Returns ListLogStoresResponse

Raise LogException

list_logtail_config(*project_name*, *offset=0*, *size=100*)
list logtail config name in a project Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **offset** (*int*) – the offset of all config names
- **size** (*int*) – the max return names count, -1 means all

Returns ListLogtailConfigResponse

Raise LogException

list_machine_group(*project_name*, *offset=0*, *size=100*)
list machine group names in a project Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **offset** (*int*) – the offset of all group name

- **size** (*int*) – the max return names count, -1 means all

Returns ListMachineGroupResponse

Raise LogException

list_machines(*project_name*, *group_name*, *offset=0*, *size=100*)
list machines in a machine group Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **group_name** (*string*) – the group name to list
- **offset** (*int*) – the offset of all group name
- **size** (*int*) – the max return names count, -1 means all

Returns ListMachinesResponse

Raise LogException

list_project(*offset=0*, *size=100*)
list the project Unsuccessful opertaion will cause an LogException.

Parameters

- **offset** (*int*) – the offset of all the matched names
- **size** (*int*) – the max return names count, -1 means return all data

Returns ListProjectResponse

Raise LogException

list_project_acl(*project_name*, *offset=0*, *size=100*)
list acl of a project Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **offset** (*int*) – the offset of all acl
- **size** (*int*) – the max return acl count

Returns ListAclResponse

Raise LogException

list_shards(*project_name*, *logstore_name*)
list the shard meta of a logstore Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name

Returns ListShardResponse

Raise LogException

list_shipper(*project_name*, *logstore_name*)
list odps/oss shipper Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name

- **logstore_name** (*string*) – the logstore name

Returns ListShipperResponse

Raise LogException

list_topics(*request*)
List all topics in a logstore. Unsuccessful opertaion will cause an LogException.

Parameters **request** (ListTopicsRequest) – the ListTopics request parameters class.

Returns ListTopicsResponse

Raise LogException

merge_shard(*project_name*, *logstore_name*, *shardId*)
split two adjacent readwrite hards into one shards Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shardId** (*int*) – the shard id of the left shard, server will determine the right adjacent shardId

Returns ListShardResponse

Raise LogException

pull_logs(*project_name*, *logstore_name*, *shard_id*, *cursor*, *count=1000*, *end_cursor=None*, *compress=False*)
batch pull log data from log service Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shard_id** (*int*) – the shard id
- **cursor** (*string*) – the start to cursor to get data
- **count** (*int*) – the required pull log package count, default 1000 packages
- **end_cursor** (*string*) – the end cursor position to get data
- **compress** (*boolean*) – if use zip compress for transfer data

Returns PullLogResponse

Raise LogException

put_logs(*request*)
Put logs to log service. Unsuccessful opertaion will cause an LogException.

Parameters **request** (PutLogsRequest) – the PutLogs request parameters class

Returns PutLogsResponse

Raise LogException

remove_config_to_machine_group(*project_name*, *config_name*, *group_name*)
remove a logtail config to a machine group Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name

- **config_name** (*string*) – the logtail config name to apply
- **group_name** (*string*) – the machine group name

Returns RemoveConfigToMachineGroupResponse

Raise LogException

retry_shipper_tasks(*project_name*, *logstore_name*, *shipper_name*, *task_list*)
retry failed tasks , only the failed task can be retried Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shipper_name** (*string*) – the shipper name
- **task_list** (*string array*) – the failed task_id list, e.g ['failed_task_id_1', 'failed_task_id_2',...], currently the max retry task count 10 every time

Returns RetryShipperTasksResponse

Raise LogException

set_source(*source*)
Set the source of the log client

Parameters **source** (*string*) – new source

Returns None

set_user_agent(*user_agent*)
set user agent

Parameters **user_agent** (*string*) – user agent

Returns None

split_shard(*project_name*, *logstore_name*, *shardId*, *split_hash*)
split a readwrite shard into two shards Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shardId** (*int*) – the shard id
- **split_hash** (*string*) – the internal hash between the shard begin and end hash

Returns ListShardResponse

Raise LogException

update_check_point(*project*, *logstore*, *consumer_group*, *shard*, *check_point*, *consumer=''*, *force_success=True*)
Update check point

Parameters

- **project** (*string*) – project name
- **logstore** (*string*) – logstore name
- **consumer_group** (*string*) – consumer group name
- **shard** (*int*) – shard id

- **check_point** (*string*) – checkpoint name
- **consumer** (*string*) – consumer name
- **force_success** (*bool*) – if force to succeed

Returns None

update_consumer_group(*project*, *logstore*, *consumer_group*, *timeout=None*, *in_order=None*)
Update consumer group

Parameters

- **project** (*string*) – project name
- **logstore** (*string*) – logstore name
- **consumer_group** (*string*) – consumer group name
- **timeout** (*int*) – timeout
- **in_order** (*bool*) – order

Returns None

update_index(*project_name*, *logstore_name*, *index_detail*)
update index for a logstore Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **index_detail** (IndexConfig) – the index config detail used to update index

Returns UpdateIndexResponse

Raise LogException

update_logstore(*project_name*, *logstore_name*, *ttl*, *shard_count=None*)
update the logstore meta info Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **ttl** (*int*) – the life cycle of log in the logstore in days
- **shard_count** (*int*) – deprecated, the shard count could only be updated by split & merge

Returns UpdateLogStoreResponse

Raise LogException

update_logstore_acl(*project_name*, *logstore_name*, *acl_action*, *acl_config*)
update acl of a logstore Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **acl_action** (*string*) – “grant” or “revoke”, grant or revoke the acl_config to/from a logstore

- **acl_config** (*acl_config.AclConfig*) – the detail acl config info

Returns UpdateAclResponse

Raise LogException

update_logtail_config(*project_name*, *config_detail*)
update logtail config in a project Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **config_detail** (LogtailConfigGenerator *or* SeperatorFileConfigDetail *or* SimpleFileConfigDetail *or* FullRegFileConfigDetail *or* JsonFileConfigDetail *or* ApsaraFileConfigDetail *or* SyslogConfigDetail *or* *CommonRegLogConfigDetail*) – the logtail config detail info, use *LogtailConfigGenerator.from_json* to generate config: SeperatorFileConfigDetail or SimpleFileConfigDetail or FullRegFileConfigDetail or JsonFileConfigDetail or ApsaraFileConfigDetail or SyslogConfigDetail

Returns UpdateLogtailConfigResponse

Raise LogException

update_machine_group(*project_name*, *group_detail*)
update machine group in a project Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **group_detail** (MachineGroupDetail) – the machine group detail config

Returns UpdateMachineGroupResponse

Raise LogException

update_project_acl(*project_name*, *acl_action*, *acl_config*)
update acl of a project Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **acl_action** (*string*) – "grant" or "revoke", grant or revoke the acl_config to/from a project
- **acl_config** (*acl_config.AclConfig*) – the detail acl config info

Returns UpdateAclResponse

Raise LogException

update_shipper(*project_name*, *logstore_name*, *shipper_name*, *shipper_type*, *shipper_config*)
update odps/oss shipper for every type, it only allowed one shipper Unsuccessful opertaion will cause an LogException.

Parameters

- **project_name** (*string*) – the Project name
- **logstore_name** (*string*) – the logstore name
- **shipper_name** (*string*) – the shipper name

- **shipper_type** (*string*) – only support "odps" or "oss" , the type must be same with the oringal shipper
- **shipper_config** (*OssShipperConfig or OdpsShipperConfig*) – the detail shipper config, must be OssShipperConfig or OdpsShipperConfig type

Returns UpdateShipperResponse

Raise LogException

class aliyun.log.**LogException**(*errorCode*, *errorMessage*, *requestId=''*, *resp_status=200*, *resp_header=''*, *resp_body=''*)

The Exception of the log request & response.

Parameters

- **errorCode** (*string*) – log service error code
- **errorMessage** (*string*) – detailed information for the exception
- **requestId** (*string*) – the request id of the response, '' is set if client error

class aliyun.log.**GetHistogramsRequest**(*project=None*, *logstore=None*, *fromTime=None*, *toTime=None*, *topic=None*, *query=None*)

The request used to get histograms of a query from log.

Parameters

- **project** (*string*) – project name
- **logstore** (*string*) – logstore name
- **fromTime** (*int*) – the begin time
- **toTime** (*int*) – the end time
- **topic** (*string*) – topic name of logs
- **query** (*string*) – user defined query

class aliyun.log.**GetLogsRequest**(*project=None*, *logstore=None*, *fromTime=None*, *toTime=None*, *topic=None*, *query=None*, *line=100*, *offset=0*, *reverse=False*)

The request used to get logs by a query from log.

Parameters

- **project** (*string*) – project name
- **logstore** (*string*) – logstore name
- **fromTime** (*int/string*) – the begin time, or format of time in format "%Y-%m-%d %H:%M:%S" e.g. "2018-01-02 12:12:10"
- **toTime** (*int/string*) – the end time, or format of time in format "%Y-%m-%d %H:%M:%S" e.g. "2018-01-02 12:12:10"
- **topic** (*string*) – topic name of logs
- **query** (*string*) – user defined query
- **line** (*int*) – max line number of return logs
- **offset** (*int*) – line offset of return logs
- **reverse** (*bool*) – if reverse is set to true, the query will return the latest logs first

class aliyun.log.**GetProjectLogsRequest**(*project=None*, *query=None*)
The request used to get logs by a query from log cross multiple logstores.

Parameters

- **project** (*string*) – project name
- **query** (*string*) – user defined query

class aliyun.log.**IndexConfig**(*ttl=1*, *line_config=None*, *key_config_list=None*, *all_keys_config=None*)
The index config of a logstore

Parameters

- **ttl** (*int*) – this parameter is deprecated, the ttl is same as logstore's ttl
- **line_config** (*IndexLineConfig*) – the index config of the whole log line
- **key_config_list** (*dict*) – dict (string => IndexKeyConfig), the index key configs of the keys
- **all_keys_config** (*IndexKeyConfig*) – the key config of all keys, the new create logstore should never user this param, it only used to compatible with old config

class aliyun.log.**ListTopicsRequest**(*project=None*, *logstore=None*, *token=None*, *line=None*)
The request used to get topics of a query from log.

Parameters

- **project** (*string*) – project name
- **logstore** (*string*) – logstore name
- **token** (*string*) – the start token to list topics
- **line** (*int*) – max topic counts to return

class aliyun.log.**ListLogstoresRequest**(*project=None*)
The request used to list log store from log.

Parameters **project** (*string*) – project name

class aliyun.log.**LogtailConfigGenerator**
Generator of Logtial config

class aliyun.log.**SeperatorFileConfigDetail**(*logstoreName*, *configName*, *logPath*, *filePattern*, *logSample*, *separator*, *key*, *timeKey=''*, *timeFormat=None*, *localStorage=None*, *enableRawLog=None*, *topicFormat=None*, *fileEncoding=None*, *maxDepth=None*, *preserve=None*, *preserveDepth=None*, *filterKey=None*, *filterRegex=None*, *createTime=None*, *modifyTime=None*, ***extended_items*)
The logtail config for separator mode

Parameters

- **logstoreName** (*string*) – the logstore name
- **configName** (*string*) – the config name
- **logPath** (*string*) – folder of log path /apsara/nuwa/
- **filePattern** (*string*) – file path, e.g. *.log, it will be /apsara/nuwa/.../*.log

- **logSample** (*string*) – log sample. e.g. shanghai|2000|least
- **separator** (*string*) – ' ' for tab, ' ' for space, 'l', up to 3 chars like "&&&" or "||" etc.
- **key** (*string list*) – keys to map the fields like ["city", "population", "location"]
- **timeKey** (*string*) – one key name in *key* to set the time or set it None to use system time.
- **timeFormat** (*string*) – whe timeKey is not None, set its format, refer to https://help.aliyun.com/document_detail/28980.html?spm=5176.2020520112.113.4.2243b18eHkxdNB
- **localStorage** (*bool*) – if use local cache 1GB when logtail is offline. default is True.
- **enableRawLog** (*bool*) – if upload raw data in content, default is False
- **topicFormat** (*string*) – "none", "group_topic" or regex to extract value from file path e.g. "/test/(w+).log" will extract each file as topic, default is "none"
- **fileEncoding** (*string*) – "utf8" or "gbk" so far
- **maxDepth** (*int*) – max depth of folder to scan, by default its 100, 0 means just scan the root folder
- **preserve** (*bool*) – if preserve time-out, by default is False, 30 min time-out if set it as True
- **preserveDepth** (*int*) – time-out folder depth. 1-3
- **filterKey** (*string list*) – only keep log which match the keys. e.g. ["city", "location"] will only scan files math the two fields
- **filterRegex** (*string list*) – matched value for filterKey, e.g. ["shanghai|beijing|nanjing", "east"] note, it's regex value list
- **createTime** (*int*) – timestamp of created, only useful when getting data from REST
- **modifyTime** (*int*) – timestamp of last modified time, only useful when getting data from REST
- **extended_items** (*dict*) – extended items

class aliyun.log.**SimpleFileConfigDetail**(*logstoreName*, *configName*, *logPath*, *filePattern*, *localStorage=None*, *enableRawLog=None*, *topicFormat=None*, *fileEncoding=None*, *maxDepth=None*, *preserve=None*, *preserveDepth=None*, *filterKey=None*, *filterRegex=None*, ***extended_items*)

The logtail config for simple mode

Parameters

- **logstoreName** (*string*) – the logstore name
- **configName** (*string*) – the config name
- **logPath** (*string*) – folder of log path /apsara/nuwa/
- **filePattern** (*string*) – file path, e.g. *.log, it will be /apsara/nuwa/.../*.log
- **localStorage** (*bool*) – if use local cache 1GB when logtail is offline. default is True.
- **enableRawLog** (*bool*) – if upload raw data in content, default is False
- **topicFormat** (*string*) – "none", "group_topic" or regex to extract value from file path e.g. "/test/(w+).log" will extract each file as topic, default is "none"

- **fileEncoding** (*string*) – "utf8" or "gbk" so far
- **maxDepth** (*int*) – max depth of folder to scan, by default its 100, 0 means just scan the root folder
- **preserve** (*bool*) – if preserve time-out, by default is False, 30 min time-out if set it as True
- **preserveDepth** (*int*) – time-out folder depth. 1-3
- **filterKey** (*string list*) – only keep log which match the keys. e.g. ["city", "location"] will only scan files math the two fields
- **filterRegex** (*string list*) – matched value for filterKey, e.g. ["shanghai|beijing|nanjing", "east"] note, it's regex value list
- **createTime** (*int*) – timestamp of created, only useful when getting data from REST
- **modifyTime** (*int*) – timestamp of last modified time, only useful when getting data from REST
- **extended_items** (*dict*) – extended items

class aliyun.log.**FullRegFileConfigDetail**(*logstoreName*, *configName*, *logPath*, *filePattern*, *logSample*, *logBeginRegex=None*, *regex=None*, *key=None*, *timeFormat=None*, *localStorage=None*, *enableRawLog=None*, *topicFormat=None*, *fileEncoding=None*, *maxDepth=None*, *preserve=None*, *preserveDepth=None*, *filterKey=None*, *filterRegex=None*, ***extended_items*)

The logtail config for full regex mode

Parameters

- **logstoreName** (*string*) – the logstore name
- **configName** (*string*) – the config name
- **logPath** (*string*) – folder of log path /apsara/nuwa/
- **filePattern** (*string*) – file path, e.g. *.log, it will be /apsara/nuwa/.../*.log
- **logSample** (*string*) – log sample. e.g. shanghai|2000|least
- **logBeginRegex** (*string*) – regex to match line, None means '.*', just single line mode.
- **regex** (*string*) – regex to extract fields form log. None means (.*), just capture whole line
- **key** (*string list*) – keys to map the fields like ["city", "population", "location"]. None means ["content"]
- **timeFormat** (*string*) – whe timeKey is not None, set its format, refer to https://help.aliyun.com/document_detail/28980.html?spm=5176.2020520112.113.4.2243b18eHkxdNB
- **localStorage** (*bool*) – if use local cache 1GB when logtail is offline. default is True.
- **enableRawLog** (*bool*) – if upload raw data in content, default is False
- **topicFormat** (*string*) – "none", "group_topic" or regex to extract value from file path e.g. "/test/(w+).log" will extract each file as topic, default is "none"
- **fileEncoding** (*string*) – "utf8" or "gbk" so far

- **maxDepth** (*int*) – max depth of folder to scan, by default its 100, 0 means just scan the root folder
- **preserve** (*bool*) – if preserve time-out, by default is False, 30 min time-out if set it as True
- **preserveDepth** (*int*) – time-out folder depth. 1-3
- **filterKey** (*string list*) – only keep log which match the keys. e.g. ["city", "location"] will only scan files math the two fields
- **filterRegex** (*string list*) – matched value for filterKey, e.g. ["shanghai|beijing|nanjing", "east"] note, it's regex value list
- **createTime** (*int*) – timestamp of created, only useful when getting data from REST
- **modifyTime** (*int*) – timestamp of last modified time, only useful when getting data from REST
- **extended_items** (*dict*) – extended items

class aliyun.log.**JsonFileConfigDetail**(*logstoreName*, *configName*, *logPath*, *filePattern*, *timeKey=''*, *timeFormat=None*, *localStorage=None*, *enableRawLog=None*, *topicFormat=None*, *fileEncoding=None*, *maxDepth=None*, *preserve=None*, *preserveDepth=None*, *filterKey=None*, *filterRegex=None*, *createTime=None*, *modifyTime=None*, ***extended_items*)

The logtail config for json mode

Parameters

- **logstoreName** (*string*) – the logstore name
- **configName** (*string*) – the config name
- **logPath** (*string*) – folder of log path /apsara/nuwa/
- **filePattern** (*string*) – file path, e.g. *.log, it will be /apsara/nuwa/.../*.log
- **timeKey** (*string*) – one key name in *key* to set the time or set it None to use system time.
- **timeFormat** (*string*) – whe timeKey is not None, set its format, refer to https://help.aliyun.com/document_detail/28980.html?spm=5176.2020520112.113.4.2243b18eHkxdNB
- **localStorage** (*bool*) – if use local cache 1GB when logtail is offline. default is True.
- **enableRawLog** (*bool*) – if upload raw data in content, default is False
- **topicFormat** (*string*) – "none", "group_topic" or regex to extract value from file path e.g. "/test/(w+).log" will extract each file as topic, default is "none"
- **fileEncoding** (*string*) – "utf8" or "gbk" so far
- **maxDepth** (*int*) – max depth of folder to scan, by default its 100, 0 means just scan the root folder
- **preserve** (*bool*) – if preserve time-out, by default is False, 30 min time-out if set it as True
- **preserveDepth** (*int*) – time-out folder depth. 1-3
- **filterKey** (*string list*) – only keep log which match the keys. e.g. ["city", "location"] will only scan files math the two fields

- **filterRegex** (*string list*) – matched value for filterKey, e.g. ["shanghai|beijing|nanjing", "east"] note, it's regex value list
- **createTime** (*int*) – timestamp of created, only useful when getting data from REST
- **modifyTime** (*int*) – timestamp of last modified time, only useful when getting data from REST
- **extended_items** (*dict*) – extended items

class aliyun.log.**ApsaraFileConfigDetail**(*logstoreName*, *configName*, *logPath*, *filePattern*, *logBeginRegex*, *localStorage=None*, *enableRawLog=None*, *topicFormat=None*, *fileEncoding=None*, *maxDepth=None*, *preserve=None*, *preserveDepth=None*, *filterKey=None*, *filterRegex=None*, *createTime=None*, *modifyTime=None*, ***extended_items*)

The logtail config for Apsara mode

Parameters

- **logstoreName** (*string*) – the logstore name
- **configName** (*string*) – the config name
- **logPath** (*string*) – folder of log path /apsara/nuwa/
- **filePattern** (*string*) – file path, e.g. *.log, it will be /apsara/nuwa/.../*.log
- **logBeginRegex** (*string*) – regex to match line, None means '.*', just single line mode.
- **localStorage** (*bool*) – if use local cache 1GB when logtail is offline. default is True.
- **enableRawLog** (*bool*) – if upload raw data in content, default is False
- **topicFormat** (*string*) – "none", "group_topic" or regex to extract value from file path e.g. "/test/(w+).log" will extract each file as topic, default is "none"
- **fileEncoding** (*string*) – "utf8" or "gbk" so far
- **maxDepth** (*int*) – max depth of folder to scan, by default its 100, 0 means just scan the root folder
- **preserve** (*bool*) – if preserve time-out, by default is False, 30 min time-out if set it as True
- **preserveDepth** (*int*) – time-out folder depth. 1-3
- **filterKey** (*string list*) – only keep log which match the keys. e.g. ["city", "location"] will only scan files math the two fields
- **filterRegex** (*string list*) – matched value for filterKey, e.g. ["shanghai|beijing|nanjing", "east"] note, it's regex value list
- **createTime** (*int*) – timestamp of created, only useful when getting data from REST
- **modifyTime** (*int*) – timestamp of last modified time, only useful when getting data from REST
- **extended_items** (*dict*) – extended items

class aliyun.log.**SyslogConfigDetail**(*logstoreName*, *configName*, *tag*, *localStorage=None*, *createTime=None*, *modifyTime=None*, ***extended_items*)

The logtail config for syslog mode

Parameters

- **logstoreName** (*string*) – the logstore name
- **configName** (*string*) – the config name
- **tag** (*string*) – tag for the log captured
- **localStorage** (*bool*) – if use local cache 1GB when logtail is offline. default is True.
- **createTime** (*int*) – timestamp of created, only useful when getting data from REST
- **modifyTime** (*int*) – timestamp of last modified time, only useful when getting data from REST
- **extended_items** (*dict*) – extended items

class aliyun.log.**MachineGroupDetail**(*group_name*, *machine_type*, *machine_list*, *group_type=''*, *group_attribute=None*)

The machine group detail info

Parameters

- **group_name** (*string*) – group name
- **machine_type** (*string*) – "ip" or "userdefined"
- **machine_list** (*string list*) – the list of machine ips or machine userdefined, e.g ["127.0.0.1", "127.0.0.2"]
- **group_type** (*string*) – the machine group type, "" or "Armory"
- **group_attribute** (*dict*) – the attributes in group, it contains two optional key : 1. "externalName": only used if the group_type is "Armory", its the Armory name 2. "groupTopic": group topic value

class aliyun.log.**PutLogsRequest**(*project=None*, *logstore=None*, *topic=None*, *source=None*, *logitems=None*, *hashKey=None*, *compress=False*, *logtags=None*)

The request used to send data to log.

Parameters

- **project** (*string*) – project name
- **logstore** (*string*) – logstore name
- **topic** (*string*) – topic name
- **source** (*string*) – source of the logs
- **logitems** (*list<LogItem>*) – log data
- **hashKey** (*String*) – put data with set hash, the data will be send to shard whose range contains the hashKey
- **compress** (*bool*) – if need to compress the logs
- **logtags** (*list*) – list of key:value tag pair , [(tag_key_1,tag_value_1) , (tag_key_2,tag_value_2)]

class aliyun.log.**OssShipperConfig**(*oss_bucket*, *oss_prefix*, *oss_role_arn*, *buffer_interval=300*, *buffer_mb=128*, *compress_type='snappy'*)

A oss ship config

Parameters

- **oss_bucket** (*string*) – the oss bucket name
- **oss_prefix** (*string*) – the the prefix path where to save the log

- **oss_role_arn** (*string*) – the ram arn used to get the temporary write permission to the oss bucket
- **buffer_interval** (*int*) – the time(seconds) to buffer before save to oss
- **buffer_mb** (*int*) – the data size(MB) to buffer before save to oss
- **compress_type** (*string*) – the compress type, only support 'snappy' or 'none'

class aliyun.log.**OdpsShipperConfig**(*odps_endpoint*, *odps_project*, *odps_table*, *log_fields_list*, *partition_column*, *partition_time_format*, *bufferInterval=1800*)

Odps shipper config

Parameters

- **odps_endpoint** (*string*) – the odps endpoint
- **odps_project** (*string*) – the odps project name
- **odps_table** (*string*) – the odps table name
- **log_fields_list** (*string array*) – the log field(keys in log) list mapping to the odps table column. e.g log_fields_list=['__time__', 'key_a', 'key_b'], the $log_time, $log_key_a, $log_key_b will mapping to odps table column No.1, No.2, No.3
- **partition_column** (*string array*) – the log fields mapping to odps table partition column
- **partition_time_format** (*string*) – the time format of __partition_time__, e.g yyyy_MM_dd_HH_mm

class aliyun.log.**ShipperTask**(*task_id*, *task_status*, *task_message*, *task_create_time*, *task_last_data_receive_time*, *task_finish_time*)

A shipper task

Parameters

- **task_id** (*string*) – the task id
- **task_status** (*string*) – one of ['success', 'running', 'fail']
- **task_message** (*string*) – the error message of task_status is 'fail'
- **task_create_time** (*int*) – the task create time (timestamp from 1970.1.1)
- **task_last_data_receive_time** (*int*) – last log data receive time (timestamp)
- **task_finish_time** (*int*) – the task finish time (timestamp)

CHAPTER 5

Indices and tables

- genindex
- search

Index

A

apply_config_to_machine_group() (aliyun.log.LogClient method), 40
ApsaraFileConfigDetail (class in aliyun.log), 60

C

copy_project() (aliyun.log.LogClient method), 40
create_consumer_group() (aliyun.log.LogClient method), 40
create_index() (aliyun.log.LogClient method), 41
create_logstore() (aliyun.log.LogClient method), 41
create_logtail_config() (aliyun.log.LogClient method), 41
create_machine_group() (aliyun.log.LogClient method), 41
create_project() (aliyun.log.LogClient method), 42
create_shipper() (aliyun.log.LogClient method), 42

D

delete_consumer_group() (aliyun.log.LogClient method), 42
delete_index() (aliyun.log.LogClient method), 42
delete_logstore() (aliyun.log.LogClient method), 43
delete_logtail_config() (aliyun.log.LogClient method), 43
delete_machine_group() (aliyun.log.LogClient method), 43
delete_project() (aliyun.log.LogClient method), 43
delete_shard() (aliyun.log.LogClient method), 43
delete_shipper() (aliyun.log.LogClient method), 43

F

FullRegFileConfigDetail (class in aliyun.log), 58

G

get_begin_cursor() (aliyun.log.LogClient method), 44
get_check_point() (aliyun.log.LogClient method), 44
get_check_point_fixed() (aliyun.log.LogClient method), 44
get_config_applied_machine_groups() (aliyun.log.LogClient method), 44
get_cursor() (aliyun.log.LogClient method), 44
get_cursor_time() (aliyun.log.LogClient method), 45
get_end_cursor() (aliyun.log.LogClient method), 45
get_histograms() (aliyun.log.LogClient method), 45
get_index_config() (aliyun.log.LogClient method), 45
get_log() (aliyun.log.LogClient method), 46
get_logs() (aliyun.log.LogClient method), 46
get_logstore() (aliyun.log.LogClient method), 46
get_logtail_config() (aliyun.log.LogClient method), 46
get_machine_group() (aliyun.log.LogClient method), 47
get_machine_group_applied_configs() (aliyun.log.LogClient method), 47
get_previous_cursor_time() (aliyun.log.LogClient method), 47
get_project() (aliyun.log.LogClient method), 47
get_project_logs() (aliyun.log.LogClient method), 47
get_shipper_config() (aliyun.log.LogClient method), 47
get_shipper_tasks() (aliyun.log.LogClient method), 48
GetHistogramsRequest (class in aliyun.log), 55
GetLogsRequest (class in aliyun.log), 55
GetProjectLogsRequest (class in aliyun.log), 55

H

heart_beat() (aliyun.log.LogClient method), 48

I

IndexConfig (class in aliyun.log), 56

J

JsonFileConfigDetail (class in aliyun.log), 59

L

list_consumer_group() (aliyun.log.LogClient method), 48
list_logstore() (aliyun.log.LogClient method), 48
list_logstore_acl() (aliyun.log.LogClient method), 49
list_logstores() (aliyun.log.LogClient method), 49
list_logtail_config() (aliyun.log.LogClient method), 49
list_machine_group() (aliyun.log.LogClient method), 49
list_machines() (aliyun.log.LogClient method), 50

list_project() (aliyun.log.LogClient method), 50
list_project_acl() (aliyun.log.LogClient method), 50
list_shards() (aliyun.log.LogClient method), 50
list_shipper() (aliyun.log.LogClient method), 50
list_topics() (aliyun.log.LogClient method), 51
ListLogstoresRequest (class in aliyun.log), 56
ListTopicsRequest (class in aliyun.log), 56
LogClient (class in aliyun.log), 40
LogException (class in aliyun.log), 55
LogtailConfigGenerator (class in aliyun.log), 56

M

MachineGroupDetail (class in aliyun.log), 61
merge_shard() (aliyun.log.LogClient method), 51

O

OdpsShipperConfig (class in aliyun.log), 62
OssShipperConfig (class in aliyun.log), 61

P

pull_logs() (aliyun.log.LogClient method), 51
put_logs() (aliyun.log.LogClient method), 51
PutLogsRequest (class in aliyun.log), 61

R

remove_config_to_machine_group() (aliyun.log.LogClient method), 51
retry_shipper_tasks() (aliyun.log.LogClient method), 52

S

SeperatorFileConfigDetail (class in aliyun.log), 56
set_source() (aliyun.log.LogClient method), 52
set_user_agent() (aliyun.log.LogClient method), 52
ShipperTask (class in aliyun.log), 62
SimpleFileConfigDetail (class in aliyun.log), 57
split_shard() (aliyun.log.LogClient method), 52
SyslogConfigDetail (class in aliyun.log), 60

U

update_check_point() (aliyun.log.LogClient method), 52
update_consumer_group() (aliyun.log.LogClient method), 53
update_index() (aliyun.log.LogClient method), 53
update_logstore() (aliyun.log.LogClient method), 53
update_logstore_acl() (aliyun.log.LogClient method), 53
update_logtail_config() (aliyun.log.LogClient method), 54
update_machine_group() (aliyun.log.LogClient method), 54
update_project_acl() (aliyun.log.LogClient method), 54
update_shipper() (aliyun.log.LogClient method), 54