Log Service

SDK Reference

SDK Reference

Introduction

Overview

The Log Service provides software development kits (SDKs) in multiple languages (Java, .NET, Python, PHP, and C). You can select one according to your need to improve the efficiency of the service.

Based on Log Service APIs, the Log Service SDKs provide the same capabilities as Log Service APIs. For details about the Log Service APIs, refer to API reference.

Similar to the Log Service APIs, you need an active Alibaba Cloud access key (consisting of an access key ID and access key secret) to use SDKs. For details, refer to Access key.

To use the Log Service SDKs, you need to know the Log Service endpoints in various Alibaba Cloud regions. For how to specify the root endpoint in an SDK, refer to SDK configuration.

Though the implementation details of the Log Service SDKs vary with different languages, the SDKs can be considered as Log Service APIs encapsulated in different languages and implement the same basic functions, including:

- Unified encapsulation of the Log Service APIs, removing your need to construct specific API requests and parse responses. The interfaces in various languages are similar, facilitating your switch between different languages.
- Digital signature logic for the Log Service APIs, greatly reducing the complexity of using APIs as you can ignore details of the API signature logic.
- Encapsulation of logs collected to the Log Service in the ProtoBuffer format, allowing you to write logs without concern over the ProtoBuffer format.
- Implementation of the compression method defined in the Log Service APIs, removing the need to focus on compression details. The SDKs in some languages allow you to determine whether to write logs in compression mode. (By default, the compression mode applies.)
- Unified error handling mechanism, allowing you to handle request errors by using languages you are familiar with.
- Currently, the SDKs in all languages only support synchronous requests.

The following table lists the download URLs, usage instructions, and complete programming references of the SDKs in different languages.

SDK language	Relevant document	Source code
Java	Quick start and Interface reference	GitHub
.NET	Quick start and Interface reference	GitHub
PHP	Quick start and Interface reference	GitHub
Node.js		GitHub
Java	Quick start and Interface reference	GitHub
C	For details, refer to README	GitHub
GO	For details, refer to README	GitHub
iOS	Usage instructions	GitHub
Android	Usage instructions	GitHub

Configuration

Like using APIs to interact with the Log Service, you need to specify basic configurations when using SDKs. Currently, the SDKs in all languages define a Client class as the portal class. Basic configurations are specified during the portal construction process. The basic configurations include:

- Endpoint: portal through which the client accesses the Log Service
- Alibaba Cloud access key (consisting of an access key ID and access key secret): used by the client to access the Log Service

The usage instructions of the two configuration items are described below.

Endpoint:

When using an SDK, specify the region of the Log Service project you want to access (for example, “China East 1 (Hangzhou)” and “China North 1 (Qingdao)”). Then select the matched Log Service endpoint for client initialization. The endpoint is defined in the same way as the endpoints of APIs.

1. When you select an endpoint for the client, ensure that the region of the project you want to access is the same as the region of the selected endpoint. If they are different, the SDK cannot access the project.
2. Because the endpoint can only be specified during the construction of a client instance, if you need to access a project in a different region, you must use a different

endpoint to construct a different client instance.
3. Currently, the endpoints of all APIs support only HTTP.
4. If you apply an SDK on Alibaba Cloud ECS VMs, you can use an intranet endpoint to avoid public bandwidth overhead. For details, refer to Endpoint.

Access key:

As described in the Access key section, all requests that interact with the Log Service must undergo security verification. An access key is a critical factor in request security verification and is created by pairing an access key ID and access key secret. During client construction, the access key ID and access key secret must be specified to create an access key. Before using an SDK, log on to the Alibaba Cloud Console and obtain or create an applicable access key on the key management page. Note the following points during client construction:

- If you have multiple access keys under your Alibaba Cloud account, ensure that the access key ID and access key secret specified during client construction are in pair; otherwise, the created access key cannot pass the security verification required by the Log Service.
- The specified access key must be “active”; otherwise, the request is denied by the Log Service. You can log on to the Alibaba Cloud Console to view the access key status on the key management page.

Example:

Access a project in the “China East 1 (Hangzhou)” region with an “active” access key. The access key is as follows:

```
AccessKeyId = "bq2sjzesjmo86kq35behupbq"
AccessKeySecret = "4fdO2fTDDnZPU/L7CHNdemB2Nsk="
```

The corresponding client instance is instantiated as follows:

Java:

```
String endpoint = "cn-hangzhou.sls.aliyuncs.com";       //Log Service endpoint in the "China East 1 (Hangzhou)" region.
String accessKeyId = "bq2sjzesjmo86kq35behupbq"; //Access key ID of the access key.
String accessKeySecret = "4fdO2fTDDnZPU/L7CHNdemB2Nsk=";//Access key secret of the access key.

Client client = new Client(endpoint, accessKeyId, acccessKeySecret);

//use client to operate log service project.....
```

.NET(C#):

```
String endpoint = "cn-hangzhou.sls.aliyuncs.com";       //Log Service endpoint in the "China East 1 (Hangzhou)" region.
```

```
String accessKeyId = "bq2sjzesjmo86kq35behupbq"; //Access key ID of the access key.
String accessKeySecret = "4fdO2fTDDnZPU/L7CHNdemB2Nsk=";//Access key secret of the access key.

SLSClient client = new SLSClient(endpoint, accessKeyId, accessKeySecret);

//use client to operate sls project.....
```

PHP:

```
$endpoint = 'cn-hangzhou.sls.aliyuncs.com'; //Log Service endpoint in the "China East 1 (Hangzhou)" region.
$accessKeyId = 'bq2sjzesjmo86kq35behupbq'; //Access key ID of the access key.
$accessKey = '4fdO2fTDDnZPU/L7CHNdemB2Nsk=';//Access key secret of the access key.

$client = new Aliyun_Sls_Client($endpoint, $accessKeyId, $accessKey);

//use client to operate sls project.....
```

Python:

```
# Log Service endpoint in the "China East 1 (Hangzhou)" region.
endpoint = 'cn-hangzhou.sls.aliyuncs.com'
# Access key ID of the access key.
accessKeyId = 'bq2sjzesjmo86kq35behupbq'
# Access key secret of the access key.
accessKey = '4fdO2fTDDnZPU/L7CHNdemB2Nsk='

client = LogClient(endpoint, accessKeyId, accessKey)

#use client to operate log project.....
```

Error processing

Possible SDK errors are classified as follows:

- Errors returned by the Log Service. This type of errors is returned by the Log Service and handled by SDKs. For details about this error type, refer to the common error codes of the Log Service APIs and the description of each API.
- Network errors that occur when SDKs send requests to the Log Service. This type of errors includes network interruptions and Log Service return timeout.
- Errors that are produced by SDKs and related to platforms or languages, for example, memory overflow.

Currently, the SDKs in various languages handle errors by throwing exceptions. The specific principles are as follows:

- The first and second types of errors are encapsulated as the LogException class and thrown

to users by SDKs.
- The third type of errors is not handled by SDKs, but is thrown to users as the platform- and language-specific Native Exception class.

LogException:

The LogException class is defined by SDKs to handle the logical errors of the Log Service. It inherits the basic exception classes from each language and provides the following exception information:

- Error code: indicates the error type. For the errors returned by the Log Service, the API-returned error codes are the same. For network errors occurred to SDK requests, the error code is “RequestError”. For more details, refer to the complete API reference for each language.
- Error message: indicates the message that comes with an error. For the errors returned by the Log Service, the API-returned error messages are the same. For network errors occurred to SDK requests, the error message is “request is failed.” For more details, refer to the complete API reference for each language.
- Request ID: indicates the service request ID corresponding to the current error. This ID is valid only when the Log Service returns an error message, otherwise, it is an empty string. When a request error occurs, you can provide the request ID to the Log Service team for problem locating.

Request failure and retry:

During SDK access to the Log Service, the access request may fail due to temporary network interruptions, transmission delay, and slow processing at the service end. Currently, these errors are directly thrown as exceptions and the Log Service does not implement any retry logic internally. You need to define the handling logic (whether to retry the request or directly report an error) when using an SDK.

Example:

Assume that you want to access the project named **big-game** in the “China East 1 (Hangzhou)” region and retry the request for the specified number of times in the case of a network exception. The code snippets in various languages are as follows:

Java:

```
//Other code.....

String accessId = "your_access_id"; //TODO： Use your Alibaba Cloud access key ID.
String accessKey = "your_access_key"; //TODO： Use your Alibaba Cloud access key secret.
String project = "big-game";
String endpoint = "cn-hangzhou.sls.aliyuncs.com";

int max_retries = 3;
```

```
/*
* Construct a client
*/
Client client = new Client(accessId, accessKey, endpoint);
ListLogStoresRequest lsRequest = new ListLogStoresRequest(project);

for (int i = 0; i < max_retries; i++)
{
try
{
ListLogStoresResponse res = client.ListLogStores(lsRequest)

//TODO: Processing the returned response.....
}
catch(LogException ex)
{
if (e.GetErrorCode() == "RequestError")
{
if ( i == max_retries - 1)
{
System.out.println("request is still failed after all retries.");
break;
}
else
System.out.println("request error happens, retry it!");
}
else
{
System.out.println("error code :" + e.GetErrorCode());
System.out.println("error message :" + e.GetErrorMessage());
System.out.println("error requestId :" + e.GetRequestId());
break;
}
}
catch(...)
{
System.out.println("unrecoverable exception when listing logstores.");
break;
}
}

//Other code.....
```

.NET(C#)：

```
//Other code.....

String accessId = "your_access_id"; //TODO： Use your Alibaba Cloud access key ID.
String accessKey = "your_access_key"; //TODO： Use your Alibaba Cloud access key secret.
String project = "big-game";
String endpoint = "cn-hangzhou.sls.aliyuncs.com";

int max_retries = 3;

//Construct a client
```

```
SLSClient client = new SLSClient(endpoint, accessId, accessKey);
ListLogstoresRequest request = new ListLogstoresRequest();
request.Project = project;

for (int i = 0; i < max_retries; i++)
{
try
{
ListLogstoresResponse response = client.ListLogstores(request);

//TODO: Processing the returned response.....
}
catch(LogException ex)
{
if (e.errorCode == "SLSRequestError")
{
if ( i == max_retries - 1)
{
Console.Writeline( “request is still failed after all retries.” );
break;
}
else
{
Console.Writeline("request error happens, retry it!");
}
}
else
{
Console.Writeline("error code :" + e.errorCode;
Console.Writeline("error message :" + e.Message;
Console.Writeline("error requestId :" + e.RequestId;
break;
}
}
catch(...)
{
Console.Writeline("unrecoverable exception when listing logstores.");
break;
}
}

//Other code.....
```

PHP：

```
<?php

//Other code.....

$endpoint = 'cn-hangzhou.sls.aliyuncs.com';
$accessId = 'your_access_id'; // TODO： Use your Alibaba Cloud access key ID.
$accessKey = 'your_access_key'; //TODO： Use your Alibaba Cloud access key secret.
$maxRetries = 3;
```

```
// Construct a Log Service client
$client = new Aliyun_Sls_Client($endpoint, $accessId, $accessKey);

$project = 'big-game';
$request = new Aliyun_Sls_Models_ListLogstoresRequest($project);

for($i = 0; $i < $maxRetries; ++$i)
{
try
{
$response = $client->ListLogstores($request);

//TODO: Processing the returned response.....
}
catch (Aliyun_Sls_Exception $e)
{
if ($e->getErrorCode()=='RequestError')
{
if ($i+1 == $maxRetries)
{
echo "error code :" . $e->getErrorCode() . PHP_EOL;
echo "error message :" . $e->getErrorMessage() . PHP_EOL;
break;
}
echo 'request error happens, retry it!' . PHP_EOL;
}
else
{
echo "error code :" . $e->getErrorCode() . PHP_EOL;
echo "error message :" . $e->getErrorMessage() . PHP_EOL;
echo "error requestId :" . $e->getRequestId() . PHP_EOL;
break;
}
}
catch (Exception $ex)
{
echo 'unrecoverable exception when listing logstores.' . PHP_EOL;
var_dump($ex);
break;
}
}

//Other code.....
```

Python：

```
//Other code.....

endpoint = 'cn-hangzhou.sls.aliyuncs.com'
accessId = 'your_access_id' # TODO： Use your Alibaba Cloud access key ID.
accessKey = 'your_access_key' # TODO： Use your Alibaba Cloud access key secret.
maxRetries = 3

# Construct a client
```

```
client = Client(endpoint, accessId, accessKey)

project = 'big-game'
lsRequest = ListLogstoresRequest(project)

for i in xrange(maxRetries):
try:
res = client.ListLogstores(lsRequest)
# TODO: Processing the returned response.....
except LogException as e:
if e.getErrorCode() == "RequestError":
if i+1 == maxRetries:
print "error code :" + e.getErrorCode()
print "error message :" + e.getErrorMessage()
break
else:
print "request error happens, retry it!"
else:
print "error code :" + e.getErrorCode()
print "error message :" + e.getErrorMessage()
print "error requestId :" + e.getRequestId()
break
except Exception as e:
print 'unrecoverable exception when listing logstores.'
break

//Other code.....
```

Interface regulations

Though SDKs in different languages are implemented differently, all their APIs comply with the request-response principle and are called in the following process:

1. Construct a request instance by using request parameters.
2. Call the corresponding interface in the SDK and input the request instance.
3. Encapsulate the results returned by the SDK interface into a response instance and return the instance to the user.

The code snippets below explain how to obtain the names of all LogStores under a project based on the preceding process.

Java

```
// Other code.....

String accessId = "your_access_id"; //TODO： Use your Alibaba Cloud access key ID.
String accessKey = "your_access_key"; //TODO： Use your Alibaba Cloud access key secret.
String project = "your_project"; //TODO: Use your project name.
```

```
String endpoint = "region_endpoint";//TODO： Use the endpoint corresponding to the region where your project is located.

//Construct a client instance.
Client client = new Client(endpoint, accessId, accessKey);

//Use the request parameter "project" to initialize a ListLogstores request class.
ListLogStoresRequest lsRequest = new ListLogStoresRequest(project);

//Use the request instance to call the ListLogstores interface. The return parameter is the corresponding response instance.
ListLogStoresResponse res = client.ListLogStores(lsRequest);

//Access the response instance to retrieve the request results
ArrayList<String> names = res.GetLogStores();

// Other code.....
```

.NET (C#)

```
// Other code.....

String accessId = "your_access_id"; //TODO： Use your Alibaba Cloud access key ID.
String accessKey = "your_access_key"; //TODO： Use your Alibaba Cloud access key secret.
String project = "your_project"; //TODO: Use your project name.
String endpoint = "region_endpoint";//TODO： Use the endpoint corresponding to the region where your project is located.

//Construct a client instance.
SLSClient client = new SLSClient(endpoint, accessId, accessKey);

//Use the request parameter "project" to initialize a ListLogstores request class.
ListLogStoresRequest lsRequest = new ListLogStoresRequest();
lsRequest.Project = project;

//Use the request instance to call the ListLogstores interface. The return parameter is the corresponding response instance.
ListLogStoresResponse res = client.ListLogStores(lsRequest);

//Access the response instance to retrieve the request results
List<String> names = res.Logstores;

// Other code.....
```

PHP

```
// Other code.....

$accessId = "your_access_id"; //TODO： Use your Alibaba Cloud access key ID.
$accessKey = "your_access_key"; //TODO： Use your Alibaba Cloud access key secret.
$project = "your_project"; //TODO: Use your project name.
```

```
$endpoint = "region_endpoint";//TODO： Use the endpoint corresponding to the region where your project is located.

//Construct a Log Service client instance.
$client = new Aliyun_Sls_Client($endpoint, $accessId, $accessKey);

//Use the request parameter "project" to initialize a ListLogstores request class.
$request = new Aliyun_Sls_Models_ListLogstoresRequest($project);

//Use the request instance to call the ListLogstores interface. The return parameter is the corresponding response instance.
$response = $client->listLogstores($request);

//Access the response instance to retrieve the request results
$names = $response->getLogstores();

// Other code......
```

Python

```
// Other code......

accessId = 'your_access_id'; //TODO： Use your Alibaba Cloud access key ID.
accessKey = 'your_access_key'; //TODO： Use your Alibaba Cloud access key secret.
project = 'your_project'; //TODO: Use your project name.
endpoint = 'region_endpoint';//TODO： Use the endpoint corresponding to the region where your project is located.

# Construct a client
client = LogClient(endpoint, accessId, accessKey)

# Use the request parameter "project" to initialize a ListLogstores request class.
lsRequest = ListLogstoresRequest(project)

# Use the request instance to call the ListLogstores interface. The return parameter is the corresponding response instance.
res = client.list_logstores(lsRequest)

# Access the response instance to retrieve the request results
names = res.get_logstores();

// Other code......
```

The SDKs provide multiple sets of interfaces similar to ListLogStores and define the corresponding request and response classes. In addition to the basic request-response interfaces, the SDKs in different languages provide secondary interfaces encapsulated with these basic interfaces, removing the need to construct requests and parse the final response. For details about the secondary interfaces, refer to the API reference of each SDK.

Java

Quick start

Follow the steps below to start using the Log Service Java SDK quickly.

Step 1 Create an Alibaba Cloud account

An Alibaba Cloud account is required for accessing the Log Service. If you do not have an Alibaba Cloud account, follow the steps below to create one.

1. Visit the Alibaba Cloud website and click **Register**.
2. Follow on-screen prompts to complete registration and perform real-name authentication.

> To better use Alibaba Cloud services, we suggest you complete real-name authentication as soon as possible; otherwise, some Alibaba Cloud services are not available for use.

Step 2 Obtain an Alibaba Cloud access key

Apply for an Alibaba Cloud access key before using the Log Service Java SDK.

Log on to the Alibaba Cloud key management page. Select an SDK-specific access key (consisting of an access key ID and access key secret). If you do not have any, create one and ensure that the access key is “active”. For how to create an access key, refer to Create an access key.

This access key will be used in the following steps. It must be kept confidential. You can refer to SDK configuration to learn more about how to use the access key in the SDK.

Step 3 Create a Log Service project and LogStore

Currently, the Log Service Java SDK does not support project and LogStore management. You need to create a LogStore on the console before using the SDK.

For details about how to create a project and LogStore, refer to Create a project and Create a LogStore.

> **NOTE:**
>
> - Ensure that the same Alibaba Cloud account is used to obtain the access key and create the project and LogStore.
> - For details about Log Service projects and LogStores, refer to the Log Service core concepts.

- A project name must be globally unique in the Log Service, and a LogStore name must be unique under the corresponding project.
- The region of a project is unchangeable once the project is created. Project migration between different regions of Alibaba Cloud is not supported currently.

Install the Java development environment

Currently, the Log Service Java SDK supports the J2SE 6.0 Java runtime environment and later versions. You can download the installation package at the Java website and follow instructions to install the Java development environment.

Install the Log Service Java SDK

Install the Log Service Java SDK after you build the Java development environment. Currently, two installation methods are available.

Apache Maven is recommended for obtaining the latest SDK version. You can add the following configurations to your Maven project:

```
<dependency>
<groupId>com.aliyun.openservices</groupId>
<artifactId>aliyun-log</artifactId>
<version>0.6.6</version>
</dependency>

java doc
<dependency>
<groupId>com.aliyun.openservices</groupId>
<artifactId>aliyun-log</artifactId>
<version>0.6.6</version>
<classifier>javadoc</classifier>
</dependency>
```

You can download the Java SDK package and then directly reference the local package in your Java project.

1. Click this link to download the Java SDK package. (Version updates are provided periodically. Use Maven to obtain the latest version.)
2. Decompress the downloaded package to the specified directory. The Java SDK does not require installation.
3. Add all .jar packages (including third-party dependent packages) in the SDK package to your Java project. (For detailed instructions, refer to the corresponding IDE document.)

Start a new Java project

Now you can start using the Java SDK. To interact with the Log Service and obtain the relevant output, run the following sample code in a text editor or Java IDE:

```
package sdksample;


import java.util.ArrayList;
import java.util.List;
import java.util.Vector;
import java.util.Date;

import com.aliyun.openservices.log.Client;
import com.aliyun.openservices.log.common.*;
import com.aliyun.openservices.log.exception.*;
import com.aliyun.openservices.log.request.*;
import com.aliyun.openservices.log.response.*;
import com.aliyun.openservices.log.common.LogContent;
import com.aliyun.openservices.log.common.LogGroupData;
import com.aliyun.openservices.log.common.LogItem;
import com.aliyun.openservices.log.common.Consts.CursorMode;

public class sdksample {

public static void main(String args[]) throws LogException,
InterruptedException {
String endpoint = "<log_service_endpoint>"; // Select the endpoint that matches the region of the project created
above
// Endpoint
String accessKeyId = "<your_access_key_id>"; // Use your Alibaba Cloud access key ID
String accessKeySecret = "<your_access_key_secret>"; // Use your Alibaba Cloud access key secret
String project = "<project_name>"; // Name of the project created above
String logstore = "<logstore_name>"; // Name of the LogStore created above


//Construct a client instance
Client client = new Client(endpoint, accessKeyId, accessKeySecret);

//List the names of all LogStores under the current project
int offset = 0;
int size = 100;
String logStoreSubName = "";
ListLogStoresRequest req1 = new ListLogStoresRequest(project, offset,
size, logStoreSubName);
ArrayList<String> logStores = client.ListLogStores(req1).GetLogStores();
System.out.println("ListLogs:" + logStores.toString() + "\n");

//Write logs
String topic = "";
String source = "";
// Send 10 packets consecutively, with each packet containing 10 logs
for (int i = 0; i < 10; i++) {
Vector<LogItem> logGroup = new Vector<LogItem>();
for (int j = 0; j < 10; j++) {
LogItem logItem = new LogItem(
(int) (new Date().getTime() / 1000));
logItem.PushBack("index", String.valueOf(i * 10 + j));
logGroup.add(logItem);
}
```

```
PutLogsRequest req2 = new PutLogsRequest(project, logstore, topic,
source, logGroup);
client.PutLogs(req2);
/*
You can specify the shard to which data is sent by setting the shard hashkey. Then data is written to the shard that
corresponds to the range with the hashkey. Refer to the following API:
public PutLogsResponse PutLogs(
String project,
String logStore,
String topic,
List<LogItem> logItems,
String source,
String shardHash // Write data to the shard based on the hashkey, which may be MD5(ip) or MD5(id)
) throws LogException;
*/
}

// Read the data written to Shard 0 during the past minute.
int shard_id = 0;
long curTimeInSec = System.currentTimeMillis() / 1000;
GetCursorResponse cursorRes = client.GetCursor(project, logstore,
shard_id, curTimeInSec - 60);
String beginCursor = cursorRes.GetCursor();

cursorRes = client.GetCursor(project, logstore, shard_id,
CursorMode.END);
String endCursor = cursorRes.GetCursor();

String curCursor = beginCursor;
while (curCursor.equals(endCursor) == false) {
int loggroup_count = 2; // Read two loggroups at a time
BatchGetLogResponse logDataRes = client.BatchGetLog(project,
logstore, shard_id, loggroup_count, curCursor, endCursor);

List<LogGroupData> logGroups = logDataRes.GetLogGroups();
for (LogGroupData logGroup : logGroups) {
System.out.println("Source:" + logGroup.GetSource());
System.out.println("Topic:" + logGroup.GetTopic());
for (LogItem log : logGroup.GetAllLogs()) {
System.out.println("LogTime:" + log.GetTime());
List<LogContent> contents = log.GetLogContents();
for (LogContent content : contents) {
System.out.println(content.GetKey() + ":"
+ content.GetValue());
}
}
}
String next_cursor = logDataRes.GetNextCursor();
System.out.println("The Next cursor:" + next_cursor);
curCursor = next_cursor;
}

// ！！！ Important: The next interface is called only when the indexing function is enabled.！！

//Wait 1 minute until logs are queryable
try {
```

```
Thread.sleep(60 * 1000);
} catch (InterruptedException e) {
e.printStackTrace();
}

//Query log distribution
String query = "index";
int from = (int) (new Date().getTime() / 1000 - 300);
int to = (int) (new Date().getTime() / 1000);
GetHistogramsResponse res3 = null;
while (true) {
GetHistogramsRequest req3 = new GetHistogramsRequest(project,
logstore, topic, query, from, to);
res3 = client.GetHistograms(req3);
if (res3 != null && res3.IsCompleted()) // If IsCompleted() returns "true", the query results are accurate; if "false" is
returned, query is performed again.
{
break;
}
Thread.sleep(200);
}

System.out.println("Total count of logs is " + res3.GetTotalCount());
for (Histogram ht : res3.GetHistograms()) {
System.out.printf("from %d, to %d, count %d.\n", ht.GetFrom(),
ht.GetTo(), ht.GetCount());
}

//Query log data
long total_log_lines = res3.GetTotalCount();
int log_offset = 0;
int log_line = 10;
while (log_offset <= total_log_lines) {
GetLogsResponse res4 = null;
// Read 10 lines of logs at a time for each log offset. If the read operation fails, it is retried three times at most.
for (int retry_time = 0; retry_time < 3; retry_time++) {
GetLogsRequest req4 = new GetLogsRequest(project, logstore,
from, to, topic, query, log_offset, log_line, false);
res4 = client.GetLogs(req4);
if (res4 != null && res4.IsCompleted()) {
break;
}
Thread.sleep(200);
}
System.out.println("Read log count:"
+ String.valueOf(res4.GetCount()));
log_offset += log_line;
}

}
}
```

Version list

The Log Service Java SDK allows you to operate the Alibaba Cloud Log Service conveniently using Java programs. Currently, the SDK supports J2SE 6.0 and later versions. The complete version list for this SDK is as follows:

SDK version	Download URL
0.6.1	Download
0.6.0	Download
0.4.1	Download
0.4.0	Download

.NET

Quick start

Follow the steps below to start using the .NET SDK quickly.

Step 1 Create an Alibaba Cloud account

An Alibaba Cloud account is required for accessing the Log Service. If you do not have an Alibaba Cloud account, follow the steps below to create one.

1. Visit the Alibaba Cloud website and click **Register**.
2. Follow on-screen prompts to complete registration and perform real-name authentication.

> To better use Alibaba Cloud services, we suggest you complete real-name authentication as soon as possible; otherwise, some Alibaba Cloud services are not available for use.

Step 2 Obtain an Alibaba Cloud access key

Apply for an Alibaba Cloud access key before using the Log Service Java SDK.

Log on to the Alibaba Cloud key management page. Select an SDK-specific access key (consisting of an access key ID and access key secret). If you do not have any, create one and ensure that the access key is “active”. For how to create an access key, refer to Create an access key.

This access key will be used in the following steps. It must be kept confidential. You can refer to SDK configuration to learn more about how to use the access key in the SDK.

Step 3 Create a Log Service project and LogStore

Currently, the Log Service Java SDK does not support project and LogStore management. You need to create a LogStore on the console before using the SDK.

For details about how to create a project and LogStore, refer to Create a project and Create a LogStore.

NOTE:

- Ensure that the same Alibaba Cloud account is used to obtain the access key and create the project and LogStore.
- For details about Log Service projects and LogStores, refer to the Log Service core concepts.
- A project name must be globally unique in the Log Service, and a LogStore name must be unique under the corresponding project.
- The region of a project is unchangeable once the project is created. Project migration between different regions of Alibaba Cloud is not supported currently.

Step 4 Install the .NET development environment

Currently, the Log Service SDK supports the .NET 3.5 and .NET 4.0/4.5 runtime environments. The following environments are recommended for Log Service SDK development:

- Microsoft .NET Framework 3.5, 4.0, and 4.5 (The actual version depends on the target environment required by your program.)
- Visual Studio 2010 and later versions

Step 5 Download and install the Log Service .NET SDK

Install the Log Service .NET SDK after you build the .NET development environment. The steps are as follows:

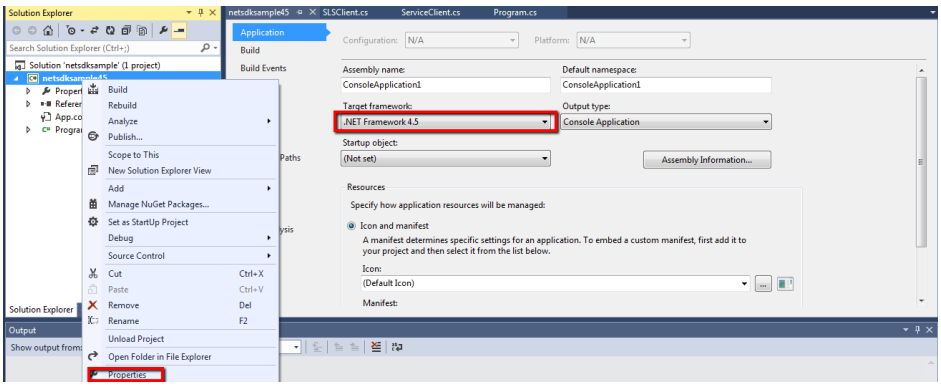
1. Download the .NET SDK package.
    - Download from GitHub:https://github.com/aliyun/aliyun-log-csharp-sdk
    - Historical version download: Visit this link to download the latest version of the Log Service .NET SDK package.
2. Decompress the downloaded package to the specified directory. The Log Service .NET SDK does not require installation. You can follow the steps below to use it directly in your Visual Studio project.

Step 5 Start a new Log Service .NET project

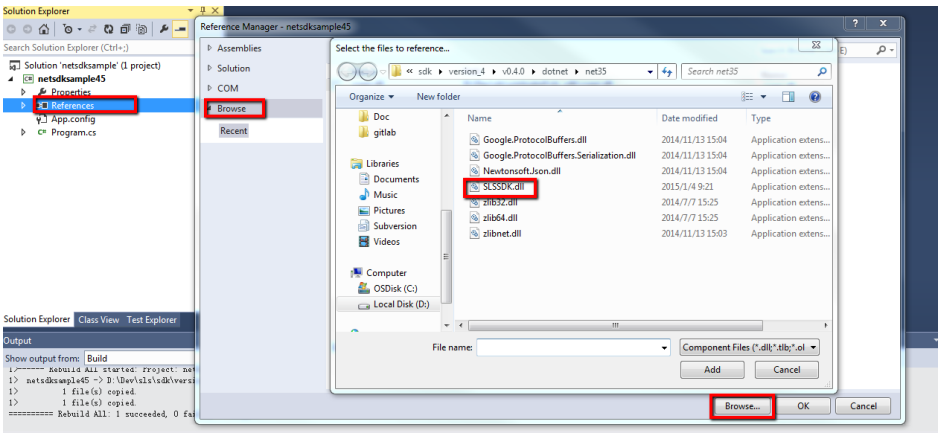
Create a .NET project after you build the .NET development environment and install the Log Service .NET SDK. A C# Console program (netsdksample45) is used as an example to illustrate the project creation process. (Projects are created in a similar way using other .NET languages and program types.)

Start Visual Studio and create a C# Console application program. Name it netsdksample45.

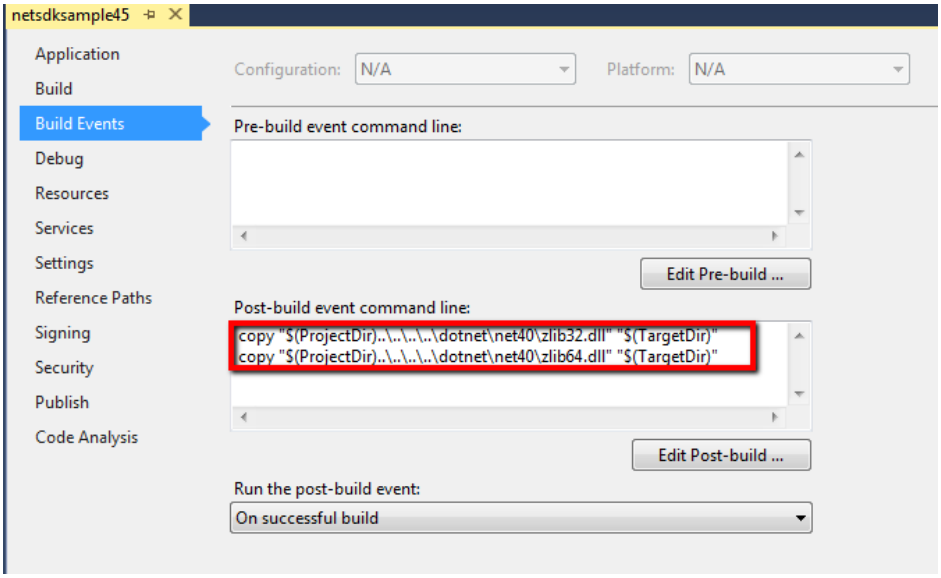
Right-click the new project and select **Properties** to go to the project property page. Select the **Application** tab, click the **Target Framework** drop-down box, and select the applicable .NET Framework target version (as shown below). Currently, the Log Service SDK only supports the .NET Framework 3.5, 4.0, and 4.5 versions. Select one of these versions. The .NET Framework 4.5 is selected here.



Right-click the new project's **References** subdirectory, select **Add Reference...**, and add the SLSSDK.dll file of the downloaded SDK (as shown below) as a project reference. Ensure that the SLSSDK.dll file version is correct. If your program's target platform is .NET Framework 4.0 or 4.5, select the file in the net40 subdirectory of the downloaded SDK package. If your program's target platform is .NET Framework 3.5, select the file in the net35 subdirectory of the downloaded SDK package.



Go to the **Properties** tab and add Build Events to copy the module on which the .NET SDK depends to the program output path, as shown below.



NOTE:

- The Log Service .NET SDK depends on the zlib library to perform compression. To enable your program to run on a 32- or 64-bit platform, ensure that the two files shown above are copied to the program output path and published with your final application program.
- The actual source path of the zlib library may be different depending on the path on your machine. Refer to the command shown above and make proper changes.

Use the Log Service SDK interface to access the Alibaba Cloud Log Service. Run the following sample code to interact with the Log Service and obtain the relevant output.

```
using System;
using System.Collections.Generic;

using Aliyun.Api.SLS;
using Aliyun.Api.SLS.Request;
using Aliyun.Api.SLS.Response;
using Aliyun.Api.SLS.Data;

namespace netsdksample45
{
class Program
{
static void Main(string[] args)
{
```

```
String endpoint = "<endpoint>"; //Select the Log Service endpoint that matches the region where the project is
located
String accessKeyId = "<your_access_key_id>"; //Use your Alibaba Cloud access key ID
String accessKeySecret = "<your_access_key_secret>"; //Use your Alibaba Cloud access key secret
String project = "<project_name>"; //Name of the project created above
String logstore = "<logstore_name>"; //Name of the LogStore created above

//Construct a client instance
SLSClient client = new SLSClient(endpoint, accessKeyId, accessKeySecret);

//List the names of all LogStores under the current project
ListLogstoresResponse res1 = client.ListLogstores(new ListLogstoresRequest(project));
Console.WriteLine("Totoal logstore number is " + res1.Count);
foreach (String name in res1.Logstores)
{
Console.WriteLine(name);
}

DateTime unixTimestampZeroPoint = new DateTime(1970, 01, 01, 0, 0, 0,DateTimeKind.Utc);

//Write logs
List<LogItem> logs = new List<LogItem>();
for (int i = 0; i < 5; i++)
{
LogItem item = new LogItem();
item.Time = (uint)((DateTime.UtcNow - unixTimestampZeroPoint).TotalSeconds);
item.PushBack("index", i.ToString());
logs.Add(item);
}

String topic = String.Empty; //Select the applicable log topic
String source = "localhost"; //Select the applicable log source (for example, IP address)
PutLogsResponse res4 = client.PutLogs(new PutLogsRequest(project, logstore, topic, source, logs));
Console.WriteLine("Request ID for PutLogs: " + res4.GetRequestId());

//Wait 1 minute until logs are queryable
System.Threading.Thread.Sleep(60 * 1000);

//Query log distribution
DateTime fromStamp = DateTime.UtcNow - new TimeSpan(0, 10, 0);
DateTime toStamp = DateTime.UtcNow;
uint from = (uint)((fromStamp - unixTimestampZeroPoint).TotalSeconds);
uint to = (uint)((toStamp - unixTimestampZeroPoint).TotalSeconds);
GetHistogramsResponse res2 = null;
do
{
res2 = client.GetHistograms(new GetHistogramsRequest(project, logstore, from, to));

} while ((res2 != null) && (!res2.IsCompleted()));

Console.WriteLine("Total count of logs is " + res2.TotalCount);
foreach (Histogram ht in res2.Histograms)
{
Console.WriteLine(String.Format("from {0}, to {1}, count {2}.", ht.From, ht.To, ht.Count));
}
```

```
//Query log data
GetLogsResponse res3 = null;
do
{
res3 = client.GetLogs(new GetLogsRequest(project, logstore, from, to, String.Empty, String.Empty, 5, 0, true));
} while ((res3 != null) && (!res3.IsCompleted()));

Console.WriteLine("Have gotten logs...");

foreach(QueriedLog log in res3.Logs)
{
Console.WriteLine("----{0}, {1}---", log.Time, log.Source);
for(int i = 0; i < log.Contents.Count; i++)
{
Console.WriteLine("{0} --> {1}", log.Contents[i].Key, log.Contents[i].Value);
}
}
}
}
}
```

Version list

The Log Service .NET SDK allows you to conveniently operate the Log Service on the .NET platform in Windows. Currently, the SDK supports the .NET Framework 3.5, 4.0, and 4.5 versions. SDK files vary with different .NET Framework versions, but the interfaces and functions are the same.

SDK GitHub address:

https://github.com/aliyun/aliyun-log-csharp-sdk

Version history:

SDK version	Download URL
0.4.1	Download

PHP

Quick start

Follow the steps below to start using the Log Service PHP SDK quickly.

Step 1 Create an Alibaba Cloud account

An Alibaba Cloud account is required for accessing the Log Service. If you do not have an Alibaba Cloud account, follow the steps below to create one.

1. Visit the Alibaba Cloud website and click **Register**.
2. Follow on-screen prompts to complete registration and perform real-name authentication.

> To better use Alibaba Cloud services, we suggest you complete real-name authentication as soon as possible; otherwise, some Alibaba Cloud services are not available for use.

Step 2 Obtain an Alibaba Cloud access key

Apply for an Alibaba Cloud access key before using the Log Service Java SDK.

Log on to the Alibaba Cloud key management page. Select an SDK-specific access key (consisting of an access key ID and access key secret). If you do not have any, create one and ensure that the access key is “active”. For how to create an access key, refer to Create an access key.

This access key will be used in the following steps. It must be kept confidential. You can refer to SDK configuration to learn more about how to use the access key in the SDK.

Step 3 Create a Log Service project and LogStore

Currently, the Log Service Java SDK does not support project and LogStore management. You need to create a LogStore on the console before using the SDK.

For details about how to create a project and LogStore, refer to Create a project and Create a LogStore.

> NOTE:
>
> - Ensure that the same Alibaba Cloud account is used to obtain the access key and create the project and LogStore.
> - For details about Log Service projects and LogStores, refer to the Log Service core concepts.
> - A project name must be globally unique in the Log Service, and a LogStore name must be unique under the corresponding project.
> - The region of a project is unchangeable once the project is created. Project migration between different regions of Alibaba Cloud is not supported currently.

Step 4 Install the PHP development environment

The PHP SDK supports PHP 5.2.1 and later versions. You can install any of these versions locally and build the corresponding PHP development environment.

Step 5 Download and install the PHP SDK

Install the PHP SDK after you build the PHP development environment. The steps are as follows:

1. Download the latest PHP SDK package from GitHub.
2. Decompress the downloaded package to the specified directory. The PHP SDK does not require installation. In addition to the SDK code, the SDK has a set of third-party dependent packages and an autoloader class for simplified use. You can follow the steps below to use the SDK directly in your PHP project.

Step 6 Start a new PHP project

Now you can start using the PHP SDK. To interact with the Log Service and obtain the relevant output, run the following sample code in a text editor or PHP IDE:

```
<?php

/* Use the autoloader class to automatically load all required PHP modules. Specify the proper path of the file containing the autoloader class*/
require_once realpath(dirname(__FILE__) . '/../Log_Autoload.php');

$endpoint = 'cn-hangzhou.sls.aliyuncs.com'; // Select the endpoint that matches the region of the project created above
$accessKeyId = 'your_access_key_id'; // Use your Alibaba Cloud access key ID
$accessKey = 'your_access_key'; // Use your Alibaba Cloud access key secret
$project = 'your_project'; // Name of the project created above
$logstore = 'your_logstore'; // Name of the LogStore created above

$client = new Aliyun_Log_Client($endpoint, $accessKeyId, $accessKey);

#List the names of all LogStores under the current project
$req1 = new Aliyun_Log_Models_ListLogstoresRequest($project);
$res1 = $client->listLogstores($req1);
var_dump($res1);


#Create a LogStore
$req2 = new Aliyun_Log_Models_CreateLogstoreRequest($project,$logstore,3,2);
$res2 = $client -> createLogstore($req2);

#Wait until the LogStore takes effect
sleep(60);


#Write logs
$topic = "";
$source = "";
$logitems = array();
for ($i = 0; $i < 5; $i++)
{
$contents = array('index1'=> strval($i));
$logItem = new Aliyun_Log_Models_LogItem();
$logItem->setTime(time());
```

```
$logItem->setContents($contents);
array_push($logitems, $logItem);
}
$req2 = new Aliyun_Log_Models_PutLogsRequest($project, $logstore, $topic, $source, $logitems);

$res2 = $client->putLogs($req2);
var_dump($res2);


# Drag data immediately
# Traverse shard IDs
$listShardRequest = new Aliyun_Log_Models_ListShardsRequest($project,$logstore);

$listShardResponse = $client -> listShards($listShardRequest);

foreach($listShardResponse-> getShardIds() as $shardId)

{

#Obtain the cursor corresponding to each shard ID

$getCursorRequest = new Aliyun_Log_Models_GetCursorRequest($project,$logstore,$shardId,null, time() - 60);
$response = $client -> getCursor($getCursorRequest);
$cursor = $response-> getCursor();
$count = 100;
while(true)
{
#Read data starting from the cursor
$batchGetDataRequest = new
Aliyun_Log_Models_BatchGetLogsRequest($project,$logstore,$shardId,$count,$cursor);
var_dump($batchGetDataRequest);
$response = $client -> batchGetLogs($batchGetDataRequest);
if($cursor == $response -> getNextCursor())
{
break;
}
$logGroupList = $response -> getLogGroupList();
foreach($logGroupList as $logGroup)
{
print ($logGroup->getCategory());

foreach($logGroup -> getLogsArray() as $log)
{
foreach($log -> getContentsArray() as $content)
{
print($content-> getKey().":".$content->getValue()."\t");
}
print("\n");
}
}
$cursor = $response -> getNextCursor();
}
}


#Wait 1 minute until logs are queryable
```

```
sleep(60);

#Query log distribution (NOTE: Ensure that indexes are created before you query logs. The PHP SDK does not provide this interface, so you need to create it on the console.)
$topic = "";
$query='';
$from = time()-3600;
$to = time();

$res3 = NULL;
while (is_null($res3) || (! $res3->isCompleted()))
{
$req3 = new Aliyun_Log_Models_GetHistogramsRequest($project, $logstore, $from, $to, $topic, $query);
$res3 = $client->getHistograms($req3);
}

var_dump($res3);


#Query log data
$res4 = NULL;

while (is_null($res4) || (! $res4->isCompleted()))
{
$req4 = new Aliyun_Log_Models_GetLogsRequest($project, $logstore, $from, $to, $topic, $query, 5, 0, False);
$res4 = $client->getLogs($req4);
}
var_dump($res4);
```

Python

Quick start

Follow the steps below to start using the Log Service Python SDK quickly.

Create an Alibaba Cloud account

An Alibaba Cloud account is required for accessing the Log Service. If you do not have an Alibaba Cloud account, follow the steps below to create one.

1. Access the Alibaba Cloud website and click **Registe**.
2. Follow the on-screen prompts to complete registration and perform real-name authentication.

To better use Alibaba Cloud services, we suggest you complete real-name authentication as soon as possible; otherwise, some Alibaba Cloud services are not available for use.

Obtain an Alibaba Cloud access key

Before using the SDK, apply for an Alibaba Cloud access key.

1. Log on to the Alibaba Cloud Console.
2. Go to the Alibaba Cloud key management page.
3. Select an SDK-specific access key. If you do not have any, create one and ensure that the access key is “active”.

This access key will be used in the following steps. It must be kept confidential. You can refer to SDK configuration to learn more about how to use the access key in the SDK.

Create a Log Service project and LogStore

Currently, the SDK does not support project and LogStore management. You need to create a LogStore on the console before using the SDK.

1. Log on to the Alibaba Cloud Console.
2. Click “Log Service” to go to the project management page.
3. Click “Create Project”. The “Create Project” dialog box is displayed.
4. Follow the on-screen prompts to specify the project name, notes, and region. Then confirm the creation of the project. The “Project Management” page shows the project after it is successfully created.
5. Click “Manage” corresponding to the new project. Go to the management page for this project and click the “LogStore Management” tab.
6. Click “Create LogStore”. The “Create LogStore” dialog box is displayed.
7. Follow the on-screen prompts to specify the LogStore name and log consumption method. Then confirm the creation of the LogStore. The “LogStore Management” tab shows the LogStore after it is successfully created.

    1. Ensure that the same Alibaba Cloud account is used to obtain the access key and create the project and LogStore.

        i. For details about Log Service projects and LogStores, refer to the Log Service core concepts.

    1. A project name must be globally unique in the Log Service, and a LogStore name must be unique under the corresponding project.
    2. The region of a project is unchangeable once the project is created. Project migration between different regions of Alibaba Cloud is not supported currently.

Install the Python environment

The Python SDK is a pure Python library and supports all Python operating systems, including Linux, Mac OS X, and Windows. Install Python as follows:

Download and install the latest Python 2 installation package.

Currently, the Python SDK supports the Python 2.6 and 2.7 environments. You can run python -V to query the current Python version.

Download and install the Python package management tool pip.

After PIP is installed, run pip -V to check whether the installation is successful and query the PIP version.

Install the dependent libraries of the Python SDK

The Python SDK depends on a set of third-party Python libraries. Before using this SDK, install the following dependent libraries:

GoogleProtocol Buffer: The Python SDK depends on the Protocol Buffer protocol when writing logs to the Log Service. The installation is as follows:

```
pip install protobuf
```

PIP needs to connect to the Python Package Index website to install Protobuf. Ensure that you can access this website on your machine. If installation fails due to network problems, you can manually download and install Protobuf from Protocol Buffer’ s GitHub website. (For details about the installation process, refer to the Readme document contained in the downloaded package.)

Python-Requests: The Python SDK depends on the Python-Requests class for HTTP communication. The installation process is as follows:

```
pip install requests
```

SimpleJSON: The Python SDK depends on SimpleJSON to process the JSON results returned by APIs. The installation process is as follows:

```
pip install simplejson
```

Install the Python SDK

Download the Python SDK after you install the Python environment. The steps are as follows:

1. Download the latest Python SDK package from GitHub.
2. Decompress the downloaded package to the specified directory.
3. Run the following command in the preceding directory to install the Python SDK:

```
python setup.py install
```

Start a Python program

Now you can start using the Python SDK. To interact with the Log Service and obtain the relevant output, run the following sample code in a text editor or Python IDE:

```
#!/usr/bin/env python
#encoding: utf-8

import time

from aliyun.log.logitem import LogItem
from aliyun.log.logclient import LogClient
from aliyun.log.getlogsrequest import GetLogsRequest
from aliyun.log.putlogsrequest import PutLogsRequest
from aliyun.log.listtopicsrequest import ListTopicsRequest
from aliyun.log.listlogstoresrequest import ListLogstoresRequest
from aliyun.log.gethistogramsrequest import GetHistogramsRequest

def main():
endpoint = 'cn-hangzhou.sls.aliyuncs.com' # Select the endpoint that matches the region of the project created above
accessKeyId = 'your_access_key_id' # Use your Alibaba Cloud access key ID
accessKey = 'your_access_key' # Use your Alibaba Cloud access key secret
project = 'your_project' # Name of the project created above
logstore = 'your_logstore' # Name of the LogStore created above

# Construct a client
client = LogClient(endpoint, accessKeyId, accessKey)

# List all LogStores
req1 = ListLogstoresRequest(project)
res1 = client.list_logstores(req1)
res1.log_print()

topic = ""
source = ""
# Send 10 packets consecutively, with each packet containing 10 logs
for i in range(10) :
logitemList = [] # LogItem list
for j in range(10):
```

```
contents = [('index', str(i * 10 + j))]
logItem = LogItem()
logItem.set_time(int(time.time()))
logItem.set_contents(contents)
logitemList.append(logItem)
req2 = PutLogsRequest(project, logstore, topic, source, logitemList)
res2 = client.put_logs(req2)
res2.log_print()


# List all shards and read the data written during the past minute
listShardRes = client.list_shards(project,logstore);
for shard in listShardRes.get_shards_info():
shard_id = shard["shardID"];
start_time = (int)(time.time() - 60)
end_time = start_time + 60
res = client.get_cursor(project, logstore, shard_id, start_time)
res.log_print()
start_cursor = res.get_cursor()
res = client.get_cursor(project, logstore, shard_id, end_time)
end_cursor = res.get_cursor()
while True :
loggroup_count = 100 # Read 100 packets each time
res = client.pull_logs(project, logstore, shard_id, start_cursor, loggroup_count, end_cursor)
res.log_print()
next_cursor = res.get_next_cursor()
if next_cursor == start_cursor :
break
start_cursor = next_cursor


# Important: The following interface can be used to query data only when the indexing function is enabled.
time.sleep(60)

topic = ""
query = "index"
From = int(time.time()) - 600
To = int(time.time())
res3 = None
# Query the number of logs that match the query criteria during the past 10 minutes. Retry if not all execution
results are correct.
while (res3 is None) or (not res3.is_completed()):
req3 = GetHistogramsRequest(project, logstore, From, To, topic, query)
res3 = client.get_histograms(req3)

res3.log_print()
# Obtain the number of logs that match the query criteria
total_log_count = res3.get_total_count()

log_line = 10
# Read 10 logs each time until all log data is queried. Retry three times if not all query results are accurate during
each query operation.
for offset in range(0, total_log_count, log_line) :
res4 = None
for retry_time in range(0, 3) :
req4 = GetLogsRequest(project, logstore, From, To, topic, query, log_line, offset, False)
```

```
res4 = client.get_logs(req4)
if res4 != None and res4.is_completed():
break
time.sleep(1)
if res4 != None:
res4.log_print()


listShardRes = client.list_shards(project,logstore);
shard = listShardRes.get_shards_info()[0]
#Split shards
if shard["status"] == "readwrite":
shard_id = shard["shardID"]
inclusiveBeginKey = shard["inclusiveBeginKey"]
midKey = inclusiveBeginKey[:-1]+str(((int)(inclusiveBeginKey[-1:]))+1)
client.split_shard(project,logstore,shard_id,midKey)


#Merge shards
shard = listShardRes.get_shards_info()[1]
if shard["status"] == "readwrite":
shard_id = shard["shardID"]
client.merge_shard(project,logstore,shard_id)

#Delete shards
shard = listShardRes.get_shards_info()[-1]
if shard["status"] == "readonly":
shard_id = shard["shardID"]
client.delete_shard(project,logstore,shard_id)



if __name__=='__main__':
main()
```

Android

Quick start

Used to collect user data on an Android platform, the Alibaba Cloud Log Service Android SDK currently provides the following function:

- log writing

Usage

.jar package

1. Download log-sdk.jar
2. Move the .jar package to the “android project libs” folder (create one if the folder does not exist).
3. Right-click log-skd.jar and select “Add as library”.
4. Add the network access permission by inserting the following content into AndroidManifest.xml:

```
<uses-permission android:name="android.permission.INTERNET" />
```

5. The Log Service Android SDK depends on the Android SDK of Alibaba fastjson for data formatting. The latest fastjson-1.1.54.android version is recommended.

GitHub

https://github.com/aliyun/aliyun-log-android-sdk

Maven

```
<dependency>
<groupId>com.aliyun.openservices</groupId>
<artifactId>aliyun-log-android-sdk</artifactId>
<version>0.3.0</version>
</dependency>
```

Example

```
import com.aliyun.logsdk.*;

/**
* Test the log writing function
* endPoint: access point (The http prefix is not required, for example, cn-hangzhou.log.aliyuncs.com.)
* The access key ID and access key secret form the access key of your Alibaba Cloud account or subaccount.
*/

/**
Use the endpoint, access key ID and access key secret to construct the Log Service client.
@endPoint: service portal. For details, refer to https://help.aliyun.com/document_detail/29008.html.
*/
final LOGClient myClient = new LOGClient("$log_service_endpoint", "your_access_id",
"$you_access_key","$project_name");

/* Create LogGroup */
final LogGroup logGroup = new LogGroup("mockTopic", "mockSource" );
```

```
for (int i = 0 ; i < 10 ; i++) {
/* Store 10 logs */
Log log = new Log();
log.PutContent("level", "info");
log.PutContent("message", "message" + String.valueOf(i) );
logGroup.PutLog(log);
}

/* Send logs in the background to avoid blocking the main thread*/
Thread t=new Thread(){public void run(){
try {
myClient.PostLog(logGroup,"logstore_name");
System.out.println("send ok");
} catch (LogException e) {
e.printStackTrace();
}
}};t.start();
```

Debugging

Refer to the console output.

C

Quick start

The Alibaba Cloud Log Service C SDK is used to access logs from various platforms, such as MIPS and OpenWrt systems. The C SDK supports RAM temporary identity (STS).

The C SDK uses cURL as a network library. The apr/apr-util library is intended for memory management and cross-platform interconnection. You only need to simply compile the source code to use the C SDK.

GitHub project address and more details

Example

```
#include "aos_log.h"
#include "aos_util.h"
#include "aos_string.h"
```

```
#include "aos_status.h"
#include "log_auth.h"
#include "log_util.h"
#include "log_api.h"
#include "log_config.h"

void log_post_logs_sample()
{
aos_pool_t *p = NULL;
aos_status_t *s = NULL;
//Create a memory pool
aos_pool_create(&p, NULL);
//Create log data
cJSON *root = cJSON_CreateObject();
cJSON_AddStringToObject(root, "__source__", "127.0.0.1");
cJSON_AddStringToObject(root, "__topic__", "topic");
cJSON *logs = cJSON_CreateArray();
cJSON_AddItemToObject(root, "__logs__", logs);
cJSON *log1 = cJSON_CreateObject();
cJSON_AddStringToObject(log1, "level", "error1");
cJSON_AddStringToObject(log1, "message", "c sdk test message1");
cJSON_AddNumberToObject(log1, "__time__", apr_time_now()/1000000);
cJSON *log2 = cJSON_CreateObject();
cJSON_AddStringToObject(log2, "level", "error2");
cJSON_AddStringToObject(log2, "message", "c sdk test message2");
cJSON_AddNumberToObject(log2, "__time__", apr_time_now()/1000000);
cJSON_AddItemToArray(logs, log1);
cJSON_AddItemToArray(logs, log2);
//Call the interface to send data
s = log_post_logs(p, LOG_ENDPOINT, ACCESS_KEY_ID, ACCESS_KEY_SECRET, PROJECT_NAME, LOGSTORE_NAME,
root);
if (aos_status_is_ok(s)) {
printf("post logs succeeded\n");
} else {
printf("put logs failed\n");
}
//Delete JSON resources
cJSON_Delete(root);
//Destroy the memory pool
aos_pool_destroy(p);
}

int main(int argc, char *argv[])
{
//Initialize HTTP I/O resources
if (aos_http_io_initialize("linux-x86_64", 0) != AOSE_OK) {
exit(1);
}
log_post_logs_sample();
//Release HTTP I/O resources
aos_http_io_deinitialize();
return 0;
}
```

The JSON format is as follows:

```
{
"__source__":"you host ip address",
# Optional
"__topic__":"you topic",
#Log data, JSON array
"__logs__":
[
{
# Unix timestamp of the log, in seconds
"__time__":1464949581,
#Other fields of the log
"key1":"value1",
"key2":"value2"
},
{
"__time__":1464949581,
"key1":"value11",
"key2":"value22"
}
]
}
```

This is a Golang SDK for Alibaba Cloud Log Service.

Loghub Go SDK

Install Instruction

Third Dependencies

```
go get github.com/cloudflare/golz4
go get github.com/golang/glog
go get github.com/gogo/protobuf/proto
go get github.com/stretchr/testify/suite
```

LogHub Golang SDK

```
go get github.com/aliyun/aliyun-log-go-sdk
```

Example

Write and Read LogHub

```
package main

import (
"fmt"
"time"
"strconv"
"math/rand"
"github.com/aliyun/aliyun-log-go-sdk/example/util"
"github.com/gogo/protobuf/proto"
sls "github.com/aliyun/aliyun-log-go-sdk"
)

func main() {

fmt.Println("loghub sample begin")
begin_time := uint32(time.Now().Unix())
rand.Seed(int64(begin_time))
logstore_name := "test"
logstore, err := util.Project.GetLogStore(logstore_name)
if logstore == nil {
fmt.Printf("GetLogStore fail, err:%v\n", err)
err = util.Project.CreateLogStore(logstore_name, 1, 2)
if err != nil {
fmt.Printf("CreateLogStore fail, err: ", err)
return
}
fmt.Println("CreateLogStore success")
} else {
fmt.Printf("GetLogStore success, name: %s, ttl: %d, shardCount: %d, createTime: %d, lastModifyTime: %d\n",
logstore.Name, logstore.TTL, logstore.ShardCount, logstore.CreateTime, logstore.LastModifyTime)
}
// put logs to logstore
for loggroupIdx := 0; loggroupIdx < 2; loggroupIdx++ {
logs := []*sls.Log {}
for logIdx := 0; logIdx < 100; logIdx++ {
content := []*sls.LogContent {}
for colIdx := 0; colIdx < 10; colIdx++ {
content = append(content, &sls.LogContent {
Key: proto.String(fmt.Sprintf("col_%d", colIdx)),
Value: proto.String(fmt.Sprintf("loggroup idx: %d, log idx: %d, col idx: %d, value: %d", loggroupIdx, logIdx, colIdx,
rand.Intn(10000000))),
})
}
log := &sls.Log{
Time: proto.Uint32(uint32(time.Now().Unix())),
Contents: content,
}
logs = append(logs, log)
}
loggroup := &sls.LogGroup {
Topic: proto.String(""),
Source: proto.String("10.230.201.117"),
Logs: logs,
}
// PostLogStoreLogs API Ref: https://www.alibabacloud.com/help/doc-detail/29026.htm
err = logstore.PutLogs(loggroup)
```

```
if err == nil {
fmt.Println("PutLogs success")
} else {
fmt.Printf("PutLogs fail, err: %s\n", err)
}
time.Sleep(1000 * time.Millisecond)
}
// pull logs from logstore
shards, err := logstore.ListShards()
for _, sh := range shards {
if sh == 0 {
// GetCursor API Ref: https://www.alibabacloud.com/help/doc-detail/29024.htm
begin_cursor, _ := logstore.GetCursor(sh, "begin")
end_cursor, _ := logstore.GetCursor(sh, "end")
// PullLogs API Ref: https://www.alibabacloud.com/help/doc-detail/29025.htm
loggrouplist, next_cursor, _ := logstore.PullLogs(sh, begin_cursor, end_cursor, 100)
fmt.Printf("shard: %d, begin_cursor: %s, end_cursor: %s, next_cursor: %s\n", sh, begin_cursor, end_cursor,
next_cursor)
for _, loggroup := range loggrouplist.LogGroups {
for _, log := range loggroup.Logs {
for _, content := range log.Contents {
fmt.Printf("key:%s, value:%s\n", content.GetKey(), content.GetValue())
}
}
}
} else {
begin_cursor, _ := logstore.GetCursor(sh, strconv.Itoa(int(begin_time) + 2))
for {
loggrouplist, next_cursor, _ := logstore.PullLogs(sh, begin_cursor, "", 2)
fmt.Printf("shard: %d, begin_cursor: %s, next_cursor: %s, len(loggrouplist.LogGroups): %d\n", sh, begin_cursor,
next_cursor, len(loggrouplist.LogGroups))
if len(loggrouplist.LogGroups) == 0 {
// means no more data in this shard, you can break out or sleep to wait new data
break
} else {
for _, loggroup := range loggrouplist.LogGroups {
for _, log := range loggroup.Logs {
for _, content := range log.Contents {
fmt.Printf("key:%s, value:%s\n", content.GetKey(), content.GetValue())
}
}
}
begin_cursor = next_cursor
}
}
}
}
fmt.Println("loghub sample end")
}
```

Use Index on LogHub

```
package main
```

```
import (
"fmt"
"time"
"math/rand"
"github.com/gogo/protobuf/proto"
sls "github.com/aliyun/aliyun-log-go-sdk"
"github.com/aliyun/aliyun-log-go-sdk/example/util"
)

func main() {

fmt.Println("loghub sample begin")
logstore_name := "test"
util.Project.DeleteLogStore(logstore_name)
time.Sleep(15 * 1000 * time.Millisecond)
err := util.Project.CreateLogStore(logstore_name, 1, 2)
if err != nil {
fmt.Printf("CreateLogStore fail, err: ", err)
return
}
time.Sleep(15 * 1000 * time.Millisecond)
fmt.Println("CreateLogStore success")
logstore, err := util.Project.GetLogStore(logstore_name)
if err != nil {
fmt.Printf("GetLogStore fail, err: ", err)
return
}
fmt.Printf("GetLogStore success, name: %s, ttl: %d, shardCount: %d, createTime: %d, lastModifyTime: %d\n",
logstore.Name, logstore.TTL, logstore.ShardCount, logstore.CreateTime, logstore.LastModifyTime)
indexKeys := map[string]sls.IndexKey {
"col_0": sls.IndexKey {
Token: []string{" "},
CaseSensitive: false,
Type: "long",
},
"col_1": sls.IndexKey {
Token: []string{",", ":", " "},
CaseSensitive: false,
Type: "text",
},
}
index := sls.Index {
TTL: 7,
Keys: indexKeys,
Line: &sls.IndexLine {
Token: []string{",", ":", " "},
CaseSensitive: false,
IncludeKeys: []string{},
ExcludeKeys: []string{},
},
}
err = logstore.CreateIndex(index)
if err != nil {
fmt.Printf("CreateIndex fail, err: ", err)
return
}
```

```
fmt.Println("CreateIndex success")
time.Sleep(30 * 1000 * time.Millisecond)
begin_time := uint32(time.Now().Unix())
rand.Seed(int64(begin_time))
// put logs to logstore
for loggroupIdx := 0; loggroupIdx < 10; loggroupIdx++ {
logs := []*sls.Log {}
for logIdx := 0; logIdx < 100; logIdx++ {
content := []*sls.LogContent {}
for colIdx := 0; colIdx < 10; colIdx++ {
if colIdx == 0 {
content = append(content, &sls.LogContent {
Key: proto.String(fmt.Sprintf("col_%d", colIdx)),
Value: proto.String(fmt.Sprintf("%d", rand.Intn(10000000))),
})
} else
{
content = append(content, &sls.LogContent {
Key: proto.String(fmt.Sprintf("col_%d", colIdx)),
Value: proto.String(fmt.Sprintf("loggroup idx: %d, log idx: %d, col idx: %d, value: %d", loggroupIdx, logIdx, colIdx,
rand.Intn(10000000))),
})
}
}
log := &sls.Log{
Time: proto.Uint32(uint32(time.Now().Unix())),
Contents: content,
}
logs = append(logs, log)
}
loggroup := &sls.LogGroup {
Topic: proto.String(""),
Source: proto.String("10.230.201.117"),
Logs: logs,
}
// PutLogs API Ref: https://www.alibabacloud.com/help/doc-detail/29026.htm
err = logstore.PutLogs(loggroup)
if err == nil {
fmt.Println("PutLogs success")
} else {
fmt.Printf("PutLogs fail, err: %s\n", err)
}
time.Sleep(1000 * time.Millisecond)
}
end_time := uint32(time.Now().Unix())
time.Sleep(15 * 1000 * time.Millisecond)
// search logs from index on logstore
totalCount := int64(0)
for {
// GetHistograms API Ref: https://www.alibabacloud.com/help/doc-detail/29030.htm
ghResp, err := logstore.GetHistograms("", int64(begin_time), int64(end_time), "col_0 > 1000000")
if err != nil {
fmt.Printf("GetHistograms fail, err: %v\n", err)
time.Sleep(10 * time.Millisecond)
continue
}
```

```
fmt.Printf("complete: %s, count: %d, histograms: %v\n", ghResp.Progress, ghResp.Count, ghResp.Histograms)
totalCount += ghResp.Count
if ghResp.Progress == "Complete" {
break
}
}
offset := int64(0)
// get logs repeatedly with (offset, lines) parameters to get complete result
for offset < totalCount {
// GetLogs API Ref: https://www.alibabacloud.com/help/doc-detail/29029.htm
glResp, err := logstore.GetLogs("", int64(begin_time), int64(end_time), "col_0 > 1000000", 100, offset, false)
if err != nil {
fmt.Printf("GetLogs fail, err: %v\n", err)
time.Sleep(10 * time.Millisecond)
continue
}
fmt.Printf("Progress:%s, Count:%d, offset: %d\n", glResp.Progress, glResp.Count, offset)
offset += glResp.Count
if glResp.Count > 0 {
fmt.Printf("logs: %v\n", glResp.Logs)
}
if glResp.Progress == "Complete" && glResp.Count == 0 {
break
}
}
fmt.Println("index sample end")
}
```

Create Config for Logtail

```

package main

import (
"fmt"
"os"

sls "github.com/aliyun/aliyun-log-go-sdk"
"github.com/aliyun/aliyun-log-go-sdk/example/util"
)

var projectName = "another-project"
var logstore = "demo-store"

func main() {
// log config sample
testConf := "test-conf"
testService := "demo-service"
exist, err := checkConfigExist(testConf)
if err != nil {
fmt.Println("check conf exist fail:", err)
os.Exit(1)
}
if exist {
deleteConfig(testConf)
```

```
}
err = createConfig(testConf, projectName, logstore, testService)
if err != nil {
fmt.Println("create config fail:", err)
os.Exit(1)
}
fmt.Println("create config success")

updateConfig(testConf)
getConfig(testConf)

exist, err = checkConfigExist(testConf)
if err != nil {
os.Exit(1)
}
if !exist {
fmt.Println("config:" + testConf + " should be exist")
os.Exit(1)
}

deleteConfig(testConf)

exist, err = checkConfigExist(testConf)
if err != nil {
os.Exit(1)
}
if exist {
fmt.Println("config:" + testConf + " should not be exist")
os.Exit(1)
}
fmt.Println("log config sample end")

}

func checkConfigExist(confName string) (exist bool, err error) {
exist, err = util.Project.CheckConfigExist(confName)
if err != nil {
return false, err
}
return exist, nil
}

func deleteConfig(confName string) (err error) {
err = util.Project.DeleteConfig(confName)
if err != nil {
return err
}
return nil
}

func updateConfig(configName string) (err error) {
config, _ := util.Project.GetConfig(configName)
config.InputDetail.FilePattern = "*.log"
err = util.Project.UpdateConfig(config)
if err != nil {
return err
```

```
}
return nil
}
func getConfig(configName string) (err error) {
_, err = util.Project.GetConfig(configName)
if err != nil {
return err
}
return nil
}
func createConfig(configName string, projectName string, logstore string, serviceName string) (err error) {
keys := []string{"message"}
inputDetail := sls.InputDetail{
LogType: "common_reg_log",
LogPath: "/var/log/lambda/" + serviceName,
FilePattern: "*.LOG",
TopicFormat: "/var/log/lambda/([^/]*)/.*",
LocalStorage: true,
TimeFormat: "",
LogBeginRegex: ".*",
Regex: "(.*)",
Keys: keys,
FilterKeys: make([]string, 1),
FilterRegex: make([]string, 1),
}
outputDetail := sls.OutputDetail{
ProjectName: projectName,
LogStoreName: logstore,
}
config := &sls.LogConfig{
Name: configName,
InputType: "file",
OutputType: "LogService",
InputDetail: inputDetail,
OutputDetail: outputDetail,
}
err = util.Project.CreateConfig(config)
if err != nil {
return err
}
return nil
}
```

Create Machine Group for Logtail

```
package main

import (
"fmt"
"os"

sls "github.com/aliyun/aliyun-log-go-sdk"
"github.com/aliyun/aliyun-log-go-sdk/example/util"
)
```

```
func main() {
// machine group example
projectName := util.Project.Name
logstore := "test-logstore"
testConf := "test-conf"
testMachineGroup := "test-mg"
testService := "demo-service"
exist, err := checkMachineGroupExist(testMachineGroup)
if err != nil {
fmt.Println("check machine group fail:", err)
os.Exit(1)
}
if exist {
util.Project.DeleteMachineGroup(testMachineGroup)
}

err = createMachineGroup(testMachineGroup)
if err != nil {
fmt.Println("create machine group:" + testMachineGroup + " fail")
fmt.Println(err)
os.Exit(1)
}

err = getMachineGroup(testMachineGroup)
if err != nil {
fmt.Println("get machine group:" + testMachineGroup + " fail")
fmt.Println(err)
os.Exit(1)
}

exist, err = checkMachineGroupExist(testMachineGroup)
if err != nil {
fmt.Println("check machine group exist fail:")
fmt.Println(err)
os.Exit(1)
}
if !exist {
fmt.Println("machine group:" + testMachineGroup + " should be exist")
os.Exit(1)
}

exist, err = util.Project.CheckConfigExist(testConf)
if err != nil {
fmt.Println("check config exist fail:", err)
os.Exit(1)
}
if exist {
util.Project.DeleteConfig(testConf)
}

err = createLogConfig(testConf, projectName, logstore, testService)
if err != nil {
fmt.Println("create config fail:")
fmt.Println(err)
os.Exit(1)
```

```
}

err = applyConfToMachineGroup(testConf, testMachineGroup)
if err != nil {
fmt.Println("apply config to machine group fail:")
fmt.Println(err)
os.Exit(1)
}

err = deleteConfig(testConf)
if err != nil {
fmt.Println("delete config fail:")
fmt.Println(err)
os.Exit(1)
}

err = deleteMachineGroup(testMachineGroup)
if err != nil {
fmt.Println("delte machine group fail:")
fmt.Println(err)
os.Exit(1)
}

exist, err = checkMachineGroupExist(testMachineGroup)
if err != nil {
fmt.Println("check machine group exist fail:")
fmt.Println(err)
os.Exit(1)
}
if exist {
fmt.Println("machine group:" + testMachineGroup + " should not be exist")
}
fmt.Println("machine group sample end")
}

func applyConfToMachineGroup(confName string, mgname string) (err error) {
err = util.Project.ApplyConfigToMachineGroup(confName, mgname)
if err != nil {
return err
}
return nil
}

func createLogConfig(configName string, projectName, logstore string, serviceName string) (err error) {
logPath := "/var/log/lambda/" + serviceName
filePattern := "*.LOG"
timeFormat := "%Y/%m/%d %H:%M:%S"
key := make([]string, 1)
filterKey := make([]string, 1)
filterRegex := make([]string, 1)
topicFormat := "/var/log/lambda/([^/]*)/.*" // topicFormat is right
inputDetail := sls.InputDetail{
LogType: "common_reg_log",
LogPath: logPath,
FilePattern: filePattern,
LocalStorage: true,
```

```
TimeFormat: timeFormat,
LogBeginRegex: "",
Regex: "",
Keys: key,
FilterKeys: filterKey,
FilterRegex: filterRegex,
TopicFormat: topicFormat,
}
outputDetail := sls.OutputDetail{
ProjectName: projectName,
LogStoreName: logstore,
}
config := &sls.LogConfig{
Name: configName,
InputType: "file",
OutputType: "LogService",
InputDetail: inputDetail,
OutputDetail: outputDetail,
}
err = util.Project.CreateConfig(config)
if err != nil {
return err
}
return nil
}

func checkMachineGroupExist(groupName string) (exist bool, err error) {
exist, err = util.Project.CheckMachineGroupExist(groupName)
if err != nil {
return false, err
}
return exist, nil
}
func getMachineGroup(groupName string) (err error) {
_, err = util.Project.GetMachineGroup(groupName)
if err != nil {
return err
}
return nil
}

func deleteMachineGroup(groupName string) (err error) {
err = util.Project.DeleteMachineGroup(groupName)
if err != nil {
return err
}
return nil
}

func createMachineGroup(groupName string) (err error) {
attribute := sls.MachinGroupAttribute{
ExternalName: "",
TopicName: "",
}
machineList := []string{"mac-user-defined-id-value"}
var machineGroup = &sls.MachineGroup{
```

```
Name: groupName,
MachineIDType: "userdefined",
MachineIDList: machineList,
Attribute: attribute,
}
err = util.Project.CreateMachineGroup(machineGroup)
if err != nil {
return err
}
return nil
}

func deleteConfig(confName string) (err error) {
err = util.Project.DeleteConfig(confName)
if err != nil {
return err
}
return nil
}
```

iOS

Quick start

The Alibaba Cloud Log Service SDKs are implemented based on APIs and provide the following function:

- log writing

GitHub address:

- Swift: https://github.com/aliyun/aliyun-log-ios-sdk
- Object-C: https://github.com/lujiajing1126/AliyunLogObjc (Third-party SDKs are verified in the testing and production environments. You can post issues to the author, who will give you a swift reply.)

Swift:

```
 /*
Use the endpoint, access key ID and access key secret to construct the Log Service client.
@endPoint: service portal. For details, refer to https://help.aliyun.com/document_detail/29008.html.
*/
let myClient = try! LOGClient(endPoint: "",
```

```
accessKeyID: "",
accessKeySecret: "",
projectName:"")

/* Create LogGroup */
let logGroup = try! LogGroup(topic: "mTopic",source: "mSource")

/* Store a log */
let log1 = Log()
try! log1.PutContent("K11", value: "V11")
try! log1.PutContent("K12", value: "V12")
try! log1.PutContent("K13", value: "V13")
logGroup.PutLog(log1)

/* Store a log */
let log2 = Log()
try! log2.PutContent("K21", value: "V21")
try! log2.PutContent("K22", value: "V22")
try! log2.PutContent("K23", value: "V23")
logGroup.PutLog(log2)

/* Send the log */
myClient.PostLog(logGroup,logStoreName: ""){ response, error in

// handle response however you want
if error?.domain == NSURLErrorDomain && error?.code == NSURLErrorTimedOut {
print("timed out") // note, `response` is likely `nil` if it timed out
}
}
```

Objective-C:

Refer to GitHub