Alibaba Cloud Application Configuration Management

SDK Reference

Issue: 20190816

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. **You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.**
2. **No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.**
3. **The content of this document may be changed due to product version upgrades , adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.**
4. **This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults " and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity , applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use**

or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.

5. By law, all the content of the Alibaba Cloud website, including but not limited to works, products, images, archives, information, materials, website architecture, website graphic layout, and webpage design, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of the Alibaba Cloud website, product programs, or content shall be used, modified , reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates . The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates).
6. Please contact Alibaba Cloud directly if you discover any errors in this document.

Generic conventions

Table -1: Style conventions

Style	Description	Example
	This warning information indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	 Danger: Resetting will result in the loss of user configuration data.
	This warning information indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	 Warning: Restarting will cause business interruption. About 10 minutes are required to restore business.
	This indicates warning information, supplementary instructions, and other content that the user must understand.	 Notice: Take the necessary precautions to save exported data containing sensitive information.
	This indicates supplemental instructions, best practices, tips, and other content that is good to know for the user.	 Note: You can use Ctrl + A to select all files.
>	Multi-level menu cascade.	Settings > Network > Set network type
Bold	It is used for buttons, menus , page names, and other UI elements.	Click OK.
`Courier font`	It is used for commands.	Run the `cd / d C :/ windows` command to enter the Windows system folder.
Italics	It is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	It indicates that it is a optional value, and only one item can be selected.	`ipconfig` *[-all|-t]*

Style	Description	Example
{} or {a|b}	It indicates that it is a required value, and only one item can be selected.	`swich {stand | slave}`

Contents

1 SDK introduction

You can use ACM SDK to get and listen to configurations.

Available ACM SDKs include:

- ACM Java Native SDK: A Java native SDK for listening to and updating ACM configurations.
- Spring Cloud ACM: Java SDK that supports the Spring Cloud Config interface specification. This SDK won't be maintained any more. Therefore, we recommend that you use spring-cloud-starter-alibaba-nacos-config from Nacos instead.
- ACM Node.js SDK: A Node.js native SDK for listening to and updating ACM configurations.
- ACM C++ SDK: A C++ native SDK for listening to and updating ACM configurations.
- ACM Python (open-source): A Python native SDK for listening to and updating ACM configurations.
- ACM PHP (open-source): A PHP native SDK for listening to ACM configurations.
- Nacos Client: A Java native SDK for listening to and updating ACM configurations.

The functions and features of ACM SDKs are summarized in the following table:

Function / Language	Java Native	Java Spring Cloud	Python	Node.JS	C++	PHP	Nacos SDK
Get particular configurations	Supported	Supported	Supported	Supported	Supported	Supported	Supported
Listen to particular configurations	Supported	Supported	Supported	Supported	Supported	Unsupported	Supported
Write configurations	Supported	Unsupported	Unsupported	Unsupported	Unsupported	Supported	Supported

Function / Language	Java Native	Java Spring Cloud	Python	Node.JS	C++	PHP	Nacos SDK
Enumerate configurations under particular tenants	Supported	Unsupported	Unsupported	Unsupported	Unsupported	Unsupported	Unsupported
Support connecting server with Single IP solution	Supported	Unsupported	Supported	Unsupported	Unsupported	Unsupported	Supported
Support connecting server with multiple IP LB method*	Supported	Supported	Supported	Supported	Supported	Supported	Supported
Support user authentication with HmacSHA1 algorithm	Supported	Supported	Supported	Supported	Supported	Supported	Supported
Support local cache backup**	Supported	Supported	Supported	Supported	Supported	Unsupported	Supported
Support ECS instance RAM role authentication	Supported	Supported	Supported	Unsupported	Unsupported	Unsupported	Supported

Function / Language	Java Native	Java Spring Cloud	Python	Node.JS	C++	PHP	Nacos SDK
Open-source address	Planned	Planned	acm-sdk-python	Planned	Planned	acm-sdk-php	Nacos

- * Multiple IP LB method is a LoadBalance method based on multiple server IP addresses returned by ACM SDK from the address server, which boosts performance and achieves high availability.
- ** With local cache backup, ACM SDK can read from the local cache backup file saved when getting configurations last time, which avoids client downtime.
- *** `spring-cloud-starter-acm` won't be maintained any more. Therefore, we recommend that you use spring-cloud-starter-alibaba-nacos-config from Nacos instead.

2 ACM Java SDK

2.1 ACM Java Native SDK

2.1.1 Prerequisites

This topic explains how to use the ACM API with sample code.

Get the SDK

Enter the following POM configuration to establish API dependency on the SDK.

```
< dependency >
    < groupId > com . alibaba . edas . acm </ groupId >
    < artifactId > acm - sdk </ artifactId >
    < version > 1 . 0 . 8 </ version >
</ dependency >
<! -- Remove the following if logging implementa tion is
   available . -->
< dependency >
    < groupId > ch . qos . logback </ groupId >
    < artifactId > logback - classic </ artifactId >
    < version > 1 . 1 . 7 </ version >
</ dependency >
```

Code example

```
 import java . util . Properties ;
 import com . alibaba . edas . acm . ConfigServ ice ;
 import com . alibaba . edas . acm . exception . ConfigExce ption ;
 import com . alibaba . edas . acm . listener . ConfigChan
 geListener ;
 import com . alibaba . edas . acm . listener . Properties Listener
 ;
//  Sample code , for sample test only .
 public class ACMTest {
    //  Attribute / Switch
     private static String config = " DefaultVal ue ";
     private static Properties acmPropert ies = new
 Properties ();
     public static void main ( String [] args ) {
         try {
            //  Copy the correspond ing values from the
 namespaces page in the console .
             Properties properties = new Properties ();
             properties . put (" endpoint ", "$ endpoint ");
             properties . put (" namespace ", "$ namespace ");
            //  Access ACM with instance RAM role
            //  properties . put (" ramRoleNam e ", "$ ramRoleNam e
 ");
             properties . put (" accessKey ", "$ accessKey ");
             properties . put (" secretKey ", "$ secretKey ");
```

```
            //  If   it   is   an   encrypted   configurat  ion
, then   add   the   following   two   lines   for   automatic
decryption .
            // properties . put (" openKMSFil  ter ",  true );
            // properties . put (" regionId ", "$ regionId ");
             ConfigServ  ice . init ( properties );
            //  Actively   get   the   configurat  ion .
             String   content  =  ConfigServ  ice . getConfig ("${
dataId }", "${ group }",  6000 );
             System . out . println ( content );
            //  Add   listeners   to   the   configurat  ion   during
  initializa  tion ,  which   calls   back   a   notificati  on
when   the   configurat  ion   is   changed .
             ConfigServ  ice . addListene  r ("${ dataId }", "${ group
}",  new   ConfigChan  geListener () {
                 public   void   receiveCon  figInfo ( String
configInfo ) {
                     //  After   the   configurat  ion   is   updated
, the   latest   value   is   returned   to   the   user   via
this   callback   function .
                     //  Remember   not   to   make   blocking
  operations   in   callback   function .  Otherwise   the
notificati  on   thread   will   be   blocked .
                     config  =  configInfo ;
                     System . out . println ( configInfo );
                 }
            });
            /**
             * The   following   listener   can   be   used   if
  the   content   of   the   configurat  ion   value   is   in
properties  ( key = value )  format .  This   allows   you   to
manage   multiple   configurat  ion   items   in   one   configurat
ion .
             */
            /**
             ConfigServ  ice . addListene  r ("${ dataId }", "${ group
}",  new   Properties  Listener () {
                 @ Override
                  public   void   innerRecei  ve ( Properties
properties ) {
                     //  TODO   Auto - generated   method   stub
                      acmPropert  ies  =  properties ;
                      System . out . println ( properties );
                 }
            });
            **/
        }  catch  ( ConfigExce  ption   e ) {
             e . printStack  Trace ();
        }
        //  Keep   the   main   thread   alive   throughout   the
test ,  because   the   configurat  ion   subscripti  on   runs   in
  a   daemon   thread ,  which   exits   once   the   main   thread
exits .  The   following   code   is   not   required   in   a   real
  environmen  t .
         while  ( true ) {
             try  {
                 Thread . sleep ( 1000 );
            }  catch  ( Interrupte  dException   e ) {
                 e . printStack  Trace ();
            }
        }
    }
    //  Expose   the   configurat  ion   value   via   the   get
configurat  ion   API   for   external   use .
```

```
    public  static  String  getConfig () {
        return  config ;
    }
    //  Expose  the  configurat  ion  value  via  the  get
configurat  ion  API  for  external  use .
    public  static  Object  getPorpert  iesValue ( String  key )
{
        if  ( acmPropert  ies  ! =  null ) {
            return  acmPropert  ies . get ( key );
        }
        return  null ;
    }
}
```

Options of passing parameters

To help you get onboard, the sample code initializes the parameters with code . However, the real production process may involve different environments (namely different accounts, regions, or namespaces), and the parameters vary with environments, so you must use variables to pass the parameters. To facilitate the input parameter configuration and reduce the configuration cost, ACM provides multiple options of passing parameters.

 Note:
EDAS automatically injects for you during the publishing, and therefore you don’t need to take any action.

Initialization parameters	How to pass parameters
endpoint	From highest priority to lowest priority: 1. JVM parameters: `- Daddress . server . domain = xxx` 2. Environment variables: `address_se rver_domai n = xxx` 3. Code: see the preceding sample code
namespace	From highest priority to lowest priority: 1. JVM parameters: `- Dtenant . id = xxx` 2. Code: see the preceding sample code

Initialization parameters	How to pass parameters
ramRoleName	Note: The authorization priority is higher than the accesskey/ridgekey priority from high to low 1. JVM parameters: `- Dram . role . name = xxx` 2. Code: see the preceding sample code
accessKey/secretKey	From highest priority to lowest priority: 1. By files: store accessKey and secretKey in the format of Properties (which meets the requirement of the `public void java . io . Reader . Properties . load ( Reader reader )` method) in a file specified with `- Dspas . identity` . If not specified, then the `/ home / admin /. spas_key /& lt ; Applicatio nName >` file is used by default (the application name is specified with `- Dproject . name`) 2. Environment variable: `spas_acces sKey = xxx spas_secre tKey = xxx` 3. Code: see the preceding sample code

Related documents

- #unique_5

2.1.2 Get configurations

Description

It obtains configurations from ACM when the service starts.

```
public static String getConfig ( String dataId , String
group , long timeoutMs ) throws ConfigExce ption
```

Request parameters

Parameter	Parameter type	Description
dataId	String	Configuration ID. Use a naming rule such as `package . class` (for example `com . taobao . tc . refund . log . level`) to ensure global uniqueness. We recommend that you indicate business meaning of the configuration in the “class” section. All characters must be in lower case. Only English characters and four special characters (".", ":", "-", and "_") are allowed. It must not exceed 256 bytes.

Parameter	Parameter type	Description
group	String	Configuration group. We recommend that you use `product name : module name` (for example `ACM : Test`) to ensure the uniqueness. Only English characters and four special characters (".", ":", "-", and "_") are allowed. It must not exceed 128 bytes.
timeout	String	Length of configuration read time-out (in ms). Recommended value: 3000.

Return values

Parameter type	Description
String	Configuration value

Request example

```
try {
    // Initialize the configurat ion service . Retrieves
the following parameters in console with sample code .
    ConfigServ ice . init ("${ endpoint }", "${ namespace }", "${
accessKey }", "${ secretKey }");
    // Actively retrieves configurat ion
    String content = ConfigServ ice . getConfig ("${ dataId }",
"${ group }", 3000 );
    System . out . println ( content );
} catch ( ConfigExce ption e ) {
```

```
    //  TODO   Auto - generated   catch   block
     e . printStack  Trace ();
}
```

Exception

A `ConfigExce ption` exception is thrown in case of a configuration read time-out or a network error.

2.1.3 Listen for configurations

Description

If you want ACM to push configuration changes, you can use the ACM dynamic listener configuration interface.

```
public   static   void   addListene  r ( String   dataId ,
ConfigChan  geListener  Adapter   listener )
```

Request parameters

Parameter name	Parameter type	Description
dataId	string	Configuration ID. Use a naming rule like package .class (for example, com.taobao.tc.refund .log.level) to ensure global uniqueness. It is recommended to indicate business meaning of the configuration in the “class ” section. All characters must be in lowercase. Only English characters and four special characters (".", ":", "-", and "_") are allowed . It must not exceed 256 bytes.

Parameter name	Parameter type	Description
group	string	Configuration group. We recommend that you use product name: module name (for example ACM: Test) to guarantee the uniqueness. Only English characters and four special characters (".", ":", "-", and "_") are allowed. It must not exceed 128 bytes.
listener	ConfigChangeListener	Listener. Configuration changes go into the callback function of the listener.

Returned values

Parameter type	Description
string	Configuration value. This value is returned through the callback function during initialization or configuration modification.

Request example

```
// Initialize the configurat ion service , and the
 console automatica lly obtains the following parameters
 through the sample code .
 Configserv ice . init ("$ { endpoint }", "$ { namespace }", "$ {
 accesskey }", "$ { ridgekey }");
// Add listeners to the configurat ion during
 initializa tion , which calls back a notificati on of
 configurat ion changes .
 Configserv ice . addlistene r ("$ { dataid }", "$ { group }",
 new fig () {
     public void receiveCon figInfo ( String configInfo ) {
         // After the configurat ion is updated , the
   latest value is returned to the user by this
 callback function .
         System . out . println ( configInfo );
     }
});
// Keep the main thread alive throughout the test ,
 because the configurat ion subscripti on runs in a
 daemon thread , which exits once the main thread exits
 . The following code is not required in a real
 environmen t .
 while ( true ) {
     try {
         Thread . sleep ( 1000 );
```

```
    } catch ( Interrupte dException e ) {
        e . printStack Trace ();
    }
}
```

2.1.4 Publish configurations

Description

It publishes ACM configurations automatically with program to reduce operation and maintenance costs with automation.

Note:

It uses the same publishing interface to create or modify a configuration. If the specified configuration doesn’t exist, then it creates a configuration. If the specified configuration exists, then it updates the configuration.

```
public static boolean publishCon fig ( String dataId ,
String group , String content ) throws ConfigExce ption
```

Request parameters

Parameter	Parameter type	Description
dataId	String	Configuration ID. Use a naming rule such as `package . class` (for example `com . taobao . tc . refund . log . level`) to ensure global uniqueness. It is recommended to indicate business meaning of the configuration in the “class” section. All characters must be in lower case. Only English characters and four special characters (".", ":", "-", and "_") are allowed. It must not exceed 256 bytes.
group	String	Configuration group. We recommend that you use `product name : module name` (for example `ACM : Test`) to ensure the uniqueness. Only English characters and four special characters (".", ":", "-", and "_") are allowed. It must not exceed 128 bytes.
content	String	Configuration content. It must not exceed 100K bytes.

Returned values

Parameter type	Description
boolean	If the publishing is successful

Request example

```
try {
    // Initialize the configurat ion service . Retrieves
the following parameters in console with sample code .
    ConfigServ ice . init ("${ endpoint }", "${ namespace }", "${
accessKey }", "${ secretKey }");
    // Actively retrieves configurat ion
    boolean isPublishO k = ConfigServ ice . publishCon fig
("${ dataId }", "${ group }", "${ content }");
    System . out . println ( isPublishO k );
} catch ( ConfigExce ption e ) {
    // TODO Auto - generated catch block
    e . printStack Trace ();
}
```

Exception

A `ConfigExce ption` exception is thrown in case of a configuration read time-out or a network error.

2.1.5 Delete configurations

Description

It deletes ACM configurations automatically with program to reduce operation and maintenance costs with automation.

Note:

If the specified configuration exists, then it deletes the configuration. If the specified configuration doesn’t exist, then it returns a successful message.

```
public static boolean removeConf ig ( String dataId ,
String group ) throws ConfigExce ption
```

Request parameters

Parameter	Parameter Type	Description
dataId	String	Configuration ID
group	String	Configuration group

Returned values

Parameter type	Description
boolean	If the deletion is successful

Request example

```
try {
    // Initialize the configurat ion service . Retrieves
the following parameters in console with sample code .
    ConfigServ ice . init ("${ endpoint }", "${ namespace }", "${
accessKey }", "${ secretKey }");
    // Actively retrieves configurat ion
    boolean isRemoveOk = ConfigServ ice . removeConf ig ("${
dataId }", "${ group }");
    System . out . println ( isRemoveOk );
} catch ( ConfigExce ption e ) {
    // TODO Auto - generated catch block
    e . printStack Trace ();
}
```

Exception

A `ConfigExce ption` exception is thrown in case of a configuration read time-out or a network error.

2.2 Spring Cloud ACM

2.2.1 Prerequisites

spring-cloud-starter-acm won't be maintained any more. Therefore, we recommend that you use spring-cloud-starter-alibaba-nacos-config from Nacos instead.

Steps for using the Spring Cloud ACM SDK are as follows.

1. Add Maven dependency.

```
< dependency >
    < groupId > com . alibaba . cloud </ groupId >
    < artifactId > spring - cloud - starter - acm </ artifactId >
    < version > 1 . 0 . 8 </ version >
</ dependency >
```

2. Configure the application name and the application group.

   Configure `applicatio n . properties` in Spring Boot, and configure `spring . applicatio n . group` and `spring . applicatio n . name` .

```
spring . applicatio n . group = com . alibaba . cloud . acm
```

```
spring . applicatio  n . name = sample - app
```

3. **Configure the ACM environment and authentication information.**

   Edit the `applicatio  n . properties` **file in Spring Boot, and configure** `alibaba . acm . endpoint` **,** `alibaba . acm . namespace` **,** `alibaba . acm . accessKey` **, and** `alibaba . acm . secretKey` **:**

```
spring . applicatio  n . group = com . alibaba . cloud . acm
spring . applicatio  n . name = sample - app
alibaba . acm . endpoint = xxx

#  Namespace   ID
alibaba . acm . namespace = xxx

#  Access   ACM   with   instance   RAM   role
#  alibaba . acm . ramRoleNam  e = xxx

alibaba . acm . accessKey = xxx
alibaba . acm . secretKey = xxx

#  If   it   is   an   encrypted   configurat  ion ,  then   add
 the   following   two   lines   for   automatic   decryption .
#  alibaba . acm . openKMSFil  ter = true
#  Regionid   can   be   obtained   by   the   Zone   ID   in   the
   namespace   details
#  alibaba . acm . regionId = xxx

#  If   group   is   not   DEFAULT_GR  OUP ,  then   set   alibaba .
 acm . group   manually .
#  alibaba . acm . group = xxx

#  Options   include   properties ,  yaml ,  and   yml ,  where
 properties   is   the   default  ( only   version   1 . 0 . 8   and
   above )
# alibaba . acm . file - extension = properties
```

4. **Add application configuration in the ACM console.**

   Log on to the ACM console and create a new configuration under the correspond ing namespace.

   - **Write the Data ID in the following format:**

     `${ spring . applicatio  n . group }:${ spring . applicatio  n . name }.{ alibaba . acm . file - extension }`

     For example: `com . alibaba . cloud . acm : sample - app . properties`

   - **Select** `Properties` **for the configuration format, and put the specific key-value pairs in the configuration body:**

```
user . id  =  001
user . name  =  juven2
```

```
user . age = 88
```

Notes

- `spring - cloud - starter - acm 1 . 0 . 7` and higher version now supports `Spring Boot 2 . x` .
- `spring - cloud - starter - acm 1 . 0 . 8` and higher version now supports `YAML` .
- We recommend that you use `2 . 0 . 1 . RELEASE` and higher version of `Spring Boot 2 . x` . `2 . 0 . 0 . RELEASE` has a bug that prevents it from reading old data.
- To download the complete sample code, click: spring-cloud-acm-sample.zip.

Related documents

- #unique_5

2.2.2 Inject configuration

Use the Spring MVC annotation to inject the configurations and reduce configuration management costs.

The `@ Value` can be used directly to inject the configurations:

```
@ Component
 class SampleRunn er implements Applicatio nRunner {

    @ Value ("${ user . id }")
     String userId ;

    @ Value ("${ user . name }")
     String userName ;

    @ Value ("${ user . age }")
     int userAge ;

    @ Override
     public void run ( Applicatio nArguments args ){
         System . out . println ( userId );
         System . out . println ( userName );
         System . out . println ( userAge );
    }

}
```

Note:

If the same key is configured in `applicatio n . properties` of the Spring Boot application and `${ spring . applicatio n . group }:${ spring .`

`applicatio n . name }. properties` of ACM at the same time, the value in ACM overrides the default value of the application.

2.2.3 Spring Boot Integration

Health check

Spring Cloud ACM incorporates the Health Check of Spring Boot. Access the health endpoint to see if the Spring Boot application is properly connected to the ACM server:

```
{
  " status ": " UP ",
  " acm ": {
    " status ": " UP ",
    " dataIds ": [
      " com . alibaba . cloud . acm : sample - app . properties "
    ]
  },
  " diskSpace ": {
    " status ": " UP ",
    " total ":  1000240963  584 ,
    " free ":  8588274237  44 ,
    " threshold ":  10485760
  },
  " refreshSco  pe ": {
    " status ": " UP "
  }
}
```

@RefreshScope

Spring Cloud ACM also supports the Refresh Scope feature of Spring Cloud.

The Bean marked with the annotation `@ RefreshSco pe` automatically listens for the changes of the ACM server and updates the configuration at run time. Sample code:

```
@ RestContro  ller
@ RequestMap  ping ("/ sample ")
@ RefreshSco  pe
 class   SampleCont  roller  {

    @ Value ("${ user . name }")
     String   userName ;

    @ RequestMap  ping ("/ acm ")
     public   String   simple () {
         return  " Hello   Spring   Cloud   ACM !" + "  Hello  " +
 userName  + "!" ;
    }
```

```
}
```

The Bean marked with the annotation `@ Configurat ionPropert ies` is subject to Refresh Scope by default.

ACM Endpoint

Spring Cloud ACM has a built-in endpoint named `acm`, which can be accessed by accessing /acm under the management.port (8080 by default) of Spring Boot applications, for example: http://localhost:8080/acm

```
{
  " runtime ": {
    " sources ": [
      {
        " dataId ": " com . alibaba . cloud . acm : sample - app .
 properties ",
        " lastSynced ": " 2017 - 10 - 10   10 : 46 : 27 "
      }
    ],
    " refreshHis  tory ": [
      {
        " timestamp ": " 2017 - 10 - 10   10 : 46 : 24 ",
        " dataId ": " com . alibaba . cloud . acm : sample - app .
 properties ",
        " md5 ": " 8692ae986e  c7bc345b3f  0f4de602ff  13 "
      }
    ]
  },
  " config ": {
    " group ": " DEFAULT_GR  OUP ",
    " timeOut ":  3000 ,
    " endpoit ": " xxx ",
    " namespace ": " xxx ",
    " accessKey ": " xxx ",
    " secretKey ": " xxx "
  }
}
```

Considerations

- When using Spring Boot 2.x, the ACM Endpoint path is `/ actuator / acm`.
- When using Spring Boot 2.x, you must add configuration `management . endpoints . web . exposure . include =*` before you can access ACM Endpoint.

2.3 Nacos SDK

2.3.1 Nacos Client

This topic explains how to use Nacos Client SDK.

Get the SDK

To! -- Remove the following if logging implementation is available. -- get Nacos Client SDK, add the following configuration to the pom.xml file in the Maven project.

```
< dependency >
    < groupId > com . alibaba . nacos </ groupId >
    < artifactId > nacos - client </ artifactId >
    < version >${ latest . version }</ version >
</ dependency >
<>
< dependency >
    < groupId > ch . qos . logback </ groupId >
    < artifactId > logback - classic </ artifactId >
    < version > 1 . 2 . 3 </ version >
</ dependency >
```

Example

Run the following code in your project for configuration management.

```
 import   com . alibaba . nacos . api . NacosFacto  ry ;
 import   com . alibaba . nacos . api . PropertyKe  yConst ;
 import   com . alibaba . nacos . api . config . ConfigServ  ice ;
 import   com . alibaba . nacos . api . exception . NacosExcep  tion ;
 import   com . alibaba . nacos . client . config . listener . impl .
 Properties  Listener ;
 import   org . slf4j . Logger ;
 import   org . slf4j . LoggerFact  ory ;
 import   java . util . Properties ;
/**
 *  Sample   code ,  for   sample   test   only .
 */
 public   class   NacosExamp  le  {
     private   static   final   Logger   LOGGER  =  LoggerFact  ory .
 getLogger ( NacosExamp  le . class );
     public   static   void   main ( String []  args )  throws
 NacosExcep  tion  {
         Properties   properties  =  new   Properties ();
         String   dataId  = " com . alibaba . nacos . example ";
         String   group  = " DEFAULT_GR  OUP ";
         String   content  = " connectTim  eoutInMill  s = 5000 ";
        //  Copy   endpoint   and   namespace   from   the   namespace
   details   on   the   namespaces   page   of   the   console
         properties . put ( PropertyKe  yConst . ENDPOINT , "${
 endpoint }");
         properties . put ( PropertyKe  yConst . NAMESPACE , "${
 namespace }");
        //  We   recommend   that   you   use   the   accessKey   and
   secretKey   of   the   RAM   account .
         properties . put ( PropertyKe  yConst . ACCESS_KEY , "${
 accessKey }");
         properties . put ( PropertyKe  yConst . SECRET_KEY , "${
 secretKey }");
         ConfigServ  ice   configServ  ice  =  NacosFacto  ry .
 createConf  igService ( properties );
```

```
        //  Publish   configurat  ion
         boolean   publishCon  fig  =  configServ  ice . publishCon
fig ( dataId ,  group ,  content );
         LOGGER . info (" publishCon  fig : {}",  publishCon  fig );
         wait2Sync ();
        //  Query   configurat  ion
         String   config  =  configServ  ice . getConfig ( dataId ,
group ,  5000 );
         LOGGER . info (" getConfig : {}",  config );
        //  Listen   for   configurat  ion   changes
         configServ  ice . addListene  r ( dataId ,  group ,  new
Properties  Listener () {
            @ Override
             public   void   innerRecei  ve ( Properties   properties
) {
                 LOGGER . info (" innerRecei  ve : {}",  properties );
            }
        });
        //  Update   configurat  ion
         boolean   updateConf  ig  =  configServ  ice . publishCon
fig ( dataId ,  group , " connectTim  eoutInMill  s = 3000 ");
         LOGGER . info (" updateConf  ig : {}",  updateConf  ig );
         wait2Sync ();
        //  Delete   configurat  ion
         boolean   removeConf  ig  =  configServ  ice . removeConf  ig
( dataId ,  group );
         LOGGER . info (" removeConf  ig : {}",  removeConf  ig );
         wait2Sync ();
         config  =  configServ  ice . getConfig ( dataId ,  group ,
5000 );
         LOGGER . info (" getConfig : {}",  config );
    }
     private   static   void   wait2Sync () {
         try  {
             Thread . sleep ( 3000 );
        }  catch  ( Interrupte  dException   e ) {
             //  ignore
        }
    }
}
```

Options of passing parameters

To help you get onboard, the sample code initializes the parameters with code. However, the real production process may involve different environments (namely different accounts, regions, or namespaces), and the parameters vary with environments, so you must use variables to pass the parameters. To facilitate the input parameter configuration and reduce the configuration cost, ACM provides multiple options of passing parameters.

Note:

EDAS automatically injects for you during the publishing, and therefore you don’t need to take any action.

Initialization parameters	How to pass parameters
endpoint	From highest priority to lowest priority: 1. By JVM parameters: `- Daddress . server . domain = xxx` 2. By environmental variables: `address_se rver_domai n = xxx` 3. By code: see the preceding sample code
namespace	From highest priority to lowest priority: 1. By JVM parameters:`- Dtenant . id = xxx` 2. By code: see the preceding sample code
accessKey/secretKey	From highest priority to lowest priority: 1. By files: store accessKey and secretKey in the format of Properties (which meets the requirement of the `public void java . io . Reader . Properties . load ( Reader reader )` method) in a file specified with `- Dspas . identity` . If not specified, then the `/ home / admin /. spas_key /< Applicatio nName >` file is used by default (the application name is specified with `- Dproject . name`) 2. Environment variable: `spas_acces sKey = xxx spas_secre tKey = xxx` 3. By code: see the preceding sample code

2.3.2 Nacos Spring

This topic explains how to use Nacos Spring SDK.

Get the SDK

To get Nacos Spring SDK, add the following configuration to the pom.xml file in the Maven project.

```
< dependency >
    < groupId > com . alibaba . nacos </ groupId >
    < artifactId > nacos - spring - context </ artifactId >
    < version >${ latest . version }</ version >
</ dependency >
```

Example

1. Add `@ EnableNaco sConfig` annotation to enable the configuration management service of Nacos Spring. Use `@ NacosPrope rtySource` to load the configuration source with the `dataId` of `com . alibaba . nacos . example` , and turn on automatic update.

```
@ Configurat ion
// The endpoint and namespace can be found in the
   namespace details . We recommend that you use the
 accessKey and secretKey of the RAM account .
@ EnableNaco sConfig ( globalProp erties = @ NacosPrope rties (
 endpoint = "${ endpoint }", namespace = "${ namespace }",
 accessKey = "${ accessKey }", secretKey = "${ secretKey }"))
@ NacosPrope rtySource ( dataId = " com . alibaba . nacos .
 example ", autoRefres hed = true )
 public class NacosConfi guration {
}
```

2. Set the value of the properties with the `@ NacosValue` annotation of Nacos.

```
 Controller
@ RequestMap ping (" config ")
 public class ConfigCont roller {

    @ NacosValue ("${ connectTim eoutInMill s : 5000 }",
 autoRefres hed = true )
     private int connectTim eoutInMill s ;

    @ RequestMap ping ( value = "/ get ", method = GET )
    @ ResponseBo dy
     public int get () {
         return connectTim eoutInMill s ;
    }
```

```
}
```

3. Open the ACM console and create a new configuration under the corresponding namespace.
   - Data ID: `com . alibaba . nacos . example`
   - Select `Properties` for the configuration format, and put the specific key-value pairs in the configuration body:

     ```
     connectTim  eoutInMill  s = 3000
     ```

   For the complete sample code, see nacos-spring-config-example.

Related documents

- Nacos spring Project

2.3.3 Nacos Spring Boot

This topic explains how to use nacos-config-spring-boot-starter.

Get the Starter

To get the nacos-config-spring-boot-starter, add the following configuration to the pom.xml file in the Maven project.

```
< dependency >
    < groupId > com . alibaba . boot </ groupId >
    < artifactId > nacos - config - spring - boot - starter </
 artifactId >
    < version >${ latest . version }</ version >
</ dependency >
```

Example

1. Configure the connection in application.properties.

   ```
   #  The   value   of   endpoint   and   namespace   can   be   found
      in   namespace   details .
    nacos . config . endpoint =${ endpoint }
    nacos . config . namespace =${ namespace }
   #  We   recommend   that   you   use   the   accessKey   and
    secretKey   of   the   RAM   account
    nacos . config . access - key =${ accessKey }
    nacos . config . secret - key =${ secretKey }
   ```

2. Use `@ NacosPrope  rtySource` to load the configuration source with the `dataId` of `com . alibaba . nacos . example`, and turn on automatic update.

   ```
   @ SpringBoot  Applicatio  n
   @ NacosPrope  rtySource ( dataId  = " com . alibaba . nacos .
    example ",  autoRefres  hed  =  true )
    public   class   NacosConfi  gApplicati  on  {
   ```

```
    public   static   void   main ( String []  args ) {
        SpringAppl  ication . run ( NacosConfi  gApplicati  on .
class ,  args );
    }
}
```

3. Set the value of the properties with the @NacosValue annotation of Nacos.

```
@ Controller
@ RequestMap  ping (" config ")
 public   class   ConfigCont  roller  {

    @ NacosValue ("${ connectTim  eoutInMill  s : 5000 }",
autoRefres  hed  =  true )
     private   int   connectTim  eoutInMill  s ;

    @ RequestMap  ping ( value  = "/ get ",  method  =  GET )
    @ ResponseBo  dy
     public   int   get () {
        return   connectTim  eoutInMill  s ;
    }
}
```

4. Open the ACM console and create a new configuration under the corresponding namespace.

   - Data ID: `com . alibaba . nacos . example`
   - Select `Properties` for the configuration format, and put the specific key-value pairs in the configuration body:

     ```
     connectTim  eoutInMill  s = 3000
     ```

   For the complete sample code, see nacos-spring-boot-config-example.

Related documents

- Nacos Config Spring Boot

2.3.4 Nacos Spring Cloud

This topic explains how to integrate spring-cloud-starter-alibaba-nacos-config.

Get the Starter

Add the following configuration to the pom.xml file of the Maven project to obtain spring-cloud-starter-alibaba-nacos-config.

```
< dependency >
    < groupId > org . springfram  ework . cloud </ groupId >
    < artifactId > spring - cloud - starter - alibaba - nacos - config
 </ artifactId >
    < version >${ latest . version }</ version >
```

```
</ dependency >
```

Example

1. Configure the connection and configuration source in bootstrap.properties. The `dataId` of the configuration source is `com . alibaba . nacos . example . properties` .

```
#  The   value   of   endpoint   and   namespace   can   be   found
   in   namespace   details .
 spring . cloud . nacos . config . endpoint =${ endpoint }
 spring . cloud . nacos . config . namespace =${ namespace }
#  We   recommend   that   you   use   the   accessKey   and
 secretKey   of   the   RAM   account
 spring . cloud . nacos . config . access - key =${ accessKey }
 spring . cloud . nacos . config . secret - key =${ secretKey }
 spring . applicatio  n . name = com . alibaba . nacos . example
#  Specify   the   configurat  ion   extension   name ,  including
 properties ,  yaml ,  and   yml ,  where   properties   is   the
 default   value .
 spring . cloud . nacos . config . file - extension = properties
```

2. Sets the property value with Spring's @Value annotation. Enable automatic update of configurations with Spring Cloud native annotation @RefreshScope.

```
@ RestContro  ller
@ RequestMap  ping ("/ config ")
@ RefreshSco  pe
 public   class   ConfigCont  roller  {

    @ Value ("${ connectTim  eoutInMill  s : 5000 }")
     private   int   connectTim  eoutInMill  s ;

     public   void   setConnect  TimeoutInM  ills ( int
 connectTim  eoutInMill  s ) {
         this . connectTim  eoutInMill  s  =  connectTim
 eoutInMill  s ;
    }

    @ RequestMap  ping ( value  = "/ get ",  method  =  GET )
    @ ResponseBo  dy
     public   int   get () {
         return   connectTim  eoutInMill  s ;
    }
```

```
}
```

3. Open the ACM console and create a new configuration under the corresponding namespace.
   - Data ID: `com . alibaba . nacos . example . properties`
   - Select `Properties` for the configuration format, and put the specific key-value pairs in the configuration body:

     ```
     connectTim  eoutInMill  s = 3000
     ```

   For the complete sample code, see nacos-spring-cloud-config-example.

Related documents

- Spring Cloud Alibaba Nacos Config

3 ACM Node.js SDK

Install the ACM Client for Node.js

ACM Client for Node.js makes it possible to help front-end developers release the front and back ends independently, and separate development from operations, which improves the development efficiency dramatically.

Enter the following command to install the client:

```
npm i acm - client -- save
```

Sample code

Run the following code in your project for configuration management.

```
const ACMClient = require (' acm - client ');
const co = require (' co ');
const acm = new ACMClient ({
  endpoint : ' acm . aliyun . com ', // Can be found in the
  ACM console
  namespace : '***************', // Can be found in the
ACM console
  accessKey : '***************', // Can be found in the
ACM console
  secretKey : '***************', // Can be found in the
ACM console
  requestTim eout : 6000 , // Length of request time - out
, 6s by default .
});
// Actively pull the configurat ion .
co ( function *() {
  const content = yield acm . getConfig (' test ', ' DEFAULT_GR
OUP ');
  console . log (' getConfig = ', content );
});
// Listening for data updating
acm . subscribe ({
  dataId : ' test ',
  group : ' DEFAULT_GR OUP ',
}, content => {
  console . log ( content );
});
```

Error events handling

```
acm . on (' error ', function ( err ) {
 // Logs can be recorded here centrally
 // If error events are not listened for , all
exceptions get printed to stderr .
```

```
});
```

API description

- API for getting configurations

  Used to get the configuration from ACM when the service starts.

  ```
  function *  getConfig ( dataId ,  group )
  ```

- Request parameters

Parameter name	Parameter type	Description
dataId	string	Configuration ID. Use a naming rule like package .class (for example, com.taobao.tc.refund .log.level) to ensure global uniqueness. It is recommended to indicate business meaning of the configuration in the “class” section. All characters must be in lowercase. Only English characters and four special characters (".", ":", "-", and "_") are allowed . It must not exceed 256 bytes.
group	string	Configuration group. We recommend that you use product name: module name (for example ACM: Test) to guarantee the uniqueness. Only English characters and four special characters (".", ":", "-", and "_") are allowed. It must not exceed 128 bytes.

- Return values

Parameter type	Description
string	Configuration value

- API for configuration listening

  If you want ACM to push configuration changes, you can use the ACM dynamic configuration listening interface.

```
function subscribe ( info , listener )
```

- Request parameters

Parameter name	Parameter type	Description
info	Object	Info.dataId: Configuration ID. info.group: Configuration group
listener	Function	Listener. Configuration changes go into the callback function of the listener.

- API for canceling configuration listening

  Cancels configuration listening when the service starts.

```
function unSubscrib e ( info , [ listener ])
```

- Request parameters

Parameter name	Parameter type	Description
info	Object	info.dataId: Configuration ID. Info.group: Configuration groupp|
listener	Function	Callback function. (Optional, all listener functions are removed if this parameter is not passed in.)

Node.js project link

https://www.npmjs.com/package/acm-client

Feedback

- @hustxiaoc

4 ACM C++ SDK

This topic explains how to use ACM C++ SDK. ACM C++ SDK only supports Linux platform.

Install ACM C++ SDK

1. Download SDK dependency package: ACM C++ SDK
2. Extract the downloaded package to create the following directory structure:
   - example/
   - include/
   - lib/

   The preceding directories and files serve the following purposes:
   - example: acm.cpp demonstrates how to use SDK. Makefile compiles and manages the example directory.
   - include: the header files to be included in your own program.
   - lib: The directory contains the 64-bit static library and dynamic library.
3. The environment variable LD_LIBRARY_PATH specifies the ACM dynamic library search path, which is also the path for installation.

```
export LD_LIBRARY _PATH =/ usr / lib64 :/ usr / lib :/ lib :/
lib64 :../ lib
```

4. Go to the example directory and modify the start parameter of the acm.cpp file and the dataId and group parameters. Run the make command for compiling. Run the following sample code:

```
cd example // Enter the example directory
vim acm . cpp // Modify the acm . cpp file
make // Compile the sample code
./ acm // Run the sample code
```

Sample code

Run the following code in your project for configuration management.

```
# include " ACM . h "
using namespace std ;
using namespace acm ;
// Define the listener .
class MyListener : public ManagerLis tener
{
public :
```

```
    MyListener ( const   std :: string & amp ;  data_id ,  const
 std :: string & amp ;  group ): data_id_ ( data_id ), group_ ( group
 ){}
    virtual  ~ MyListener ()
   {}
    virtual   void   getExecuto  r ()
   {
        printf (" data_id :% s   group :% s   getExecuto  r \ n ",
 data_id_ . c_str (),  group_ . c_str ());
   }
   // Callback   during   configurat  ion   modificati  on
    virtual   void   receiveCon  figInfo (  std :: string  & amp ;
 configInfo )
   {
        printf (" data_id :% s   group :% s   configInfo :\ n % s \ n
 ",  data_id_ . c_str (),  group_ . c_str (),  configInfo . c_str ());
        config_  =  configInfo ;
   }
 private :
    std :: string   data_id_ ;
    std :: string   group_ ;
    std :: string   config_ ;
};
 int   main () {
    ACM :: init (" acm . aliyun . com ", //  endpoint :  Can   be
 found   in   the   ACM   console .
             "***************", //  namespace :  Can   be   found
 in   the   ACM   console .
             "***************", //  accessKey :  Can   be   found
 in   the   ACM   console .
             "***************"); //  secretKey :  Can   be   found
 in   the   ACM   console .
   // The   dataId   and   group   of   the   configurat  ion   that
  is   listened   for
    std :: string   dataId  = "${ dataId }";
    std :: string   group  = "${ group }";
    std :: string   content ;
   // Actively   pull   the   configurat  ion
    ACM :: getConfig ( dataId ,  group ,  5000 ,  content );
    printf (" get   ok   config  % s \ n ",  content . c_str ());
    MyListener *  listener  =  new   MyListener ( dataId ,  group );
   // Listen   for   configurat  ion   changes
    ACM :: addListene  r ( dataId ,  group ,  listener );
    printf (" add   listener   ok  % s  % s \ n ",  dataId . c_str
 (),  group . c_str ());
    do  {
        printf (" input   q   to   quit \ n ");
   }  while  ( getchar () ! = ' q ');
    return   0 ;
}
```

Interface description

- **Interface for initialization**

  Set endpoint, nameSpace, accessKey, and secretKey.

```
 static   void   init ( const   char *  endpoint ,
                  const   char *  nameSpace ,
                  char *  accessKey ,
```

```
                    char *  secretKey );
```

Parameters are described as follows:

Parameter name	Parameter type	Description
endpoint	const char*	ACM domain name, which is found in the console
nameSpace	const char*	Namespace, which is found in the console
accessKey	char*	accessKey, which is found in the console
secretKey	char*	secretKey, which is found in the console

- Interface for getting configurations

  Obtains configurations from ACM when the service starts.

```
static   bool   getConfig ( const   std :: string  & amp ; dataId ,
                      const   std :: string  & amp ; group ,
                      int   timeoutMs ,
                      std :: string  & amp ; content );
```

Parameters are described as follows:

Parameter name	Parameter type	Description
dataId	const string	Configuration ID. Use a naming rule like package .class (for example, com.taobao.tc.refund .log.level) to ensure global uniqueness. It is recommended to indicate business meaning of the configuration in the “class” section. All characters must be in lower case. Only English characters and four special characters (".", ":", "-", and "_") are allowed . It must not exceed 256 bytes.

Parameter name	Parameter type	Description
group	const string	Configuration group. We recommend that you use product name: module name (for example ACM: Test) to guarantee the uniqueness. Only English characters and four special characters (".", ":", "-", and "_") are allowed. It must not exceed 128 bytes.
timeoutMs	int	Time-out period
content	string	Returned configuration content

- Listening to configuration interface

  If you want ACM to push configuration changes, you can use the ACM dynamic configuration listening interface.

```
static   void   addListene  r ( const   std :: string  & amp ;
dataId ,
                         const   std :: string  & amp ; group ,
                         ManagerLis  tener *  listener );
```

  Parameters are described as follows:

Parameter name	Parameter type	Description
dataId	const string	dataId
group	const string	group
listener	ManagerListener	Listener. The listener callback function will be run when configuration is modified.

- Cancel configuration listening interface

  Cancels configuration listening when the service starts.

```
static   void   removeList  ener ( const   std :: string  & amp ;
dataId ,
                            const   std :: string  & amp ; group ,
```

```
                                ManagerLis  tener *  listener );
```

Parameters are described as follows:

Parameter name	Parameter type	Description
dataId	const string	dataId
group	const string	group
listener	Manager listener	Listener to be canceled

5 ACM PHP SDK

ACM provides PHP SDK for managing application configurations with ACM for PHP programs.

ACM PHP SDK is now open-sourced. For instructions, see Github.

6 ACM Python SDK

ACM provides Python SDK for managing application configurations with ACM for Python program.

ACM Python SDK is now open-sourced. For instructions, see Github.