# API使用规范

提供原生RESTful支持。您可以使用支持HTTP的编程语言进行开发。

## HTTP方法

|  |  |
| --- | --- |
| HTTP方法 | 描述 |
| GET | 获取资源信息。  所有的查询API以及读API均使用该方法。 |
| POST | 所有对资源的创建/修改/删除操作，例如创建、启动停止虚拟机，均使用该方法。 |

## 传参方式

URL（路径参数）、Query String、HTTP body三种方式均可用于传参。每种方式可以单独使用，也可以混合使用，具体使用哪种传参方式由具体API决定。

## HTTP Headers

当前API使用如下自定义HTTP Headers：

|  |  |
| --- | --- |
| 名称 | 描述 |
| X-AUTH-TOKEN | 试用access key 访问接口的认证信息，格式为 access token 加上请求校验信息，中间使用:分隔，例如  5sHF/5lt5kDxN4l5c7qEQC3kYEE=:o+jCIBOpXyqsMfJIPxnjvUvb2MY= |
| X-TASK\_UUID | 用来标记执行任务的UUID，这个在使用POST方法调用API接口时返回，用来跟踪任务 |
| X-TASK-POLL-URL | 在使用POST方法调用异步API接口时返回,提供轮询异步任务最终结果的请求地址 |
| X-WEB-CALLBACK | 对于异步API，可以通过X-WEB-CALLBACK HTTP Header指定一个回调URL用于接收API 返回。通过使用回调URL的方法，调用者可以避免使用轮询去查询一个异步API的执行结果 |

## HTTP返回码 (HTTP Status Code)

|  |  |
| --- | --- |
| 返回码 | 意义 |
| 200 | API请求接收成功 |
| 400 | API请求未包含必要的参数或包含了非法的参数。 |
| 404 | URL不存在，通常是指定了错误的API URL。 |
| 405 | API调用使用了错误的HTTP方法，例如在创建一个资源的时候用了GET方法而不是POST方法。 |
| 500 | 服务器端执行遇到一个内部错误。 |

# 同步API

所有使用GET方法的API都是同步API，HTTP Response中直接包含了API的结果。例如：

GET /cloudHost/query.do?offset=0&limit=10&currentPage=1&state=&searchtype=0&keyword=&labelId=

{

"code": 1,

"msg": null,

"id": null,

"userId": null,

"userToken": null,

"wsEndpoint": null,

"sessionToken": null,

"userName": null,

"roleName": null,

"userType": 0,

"projectIdList": null,

"basePath": null,

"projectId": null,

"projectName": null,

"guide": null,

"resourceIds": null,

"error": null,

"async": 0,

"serverTime": null,

"zoneId": null,

"total": 36,

"rows": [

{

"vmid": "0cedc6fa673643bf875624f8428b32f0",

"mirrorid": "d39aba3d56f54d0a8f33ed4409f7de0c",

"domainid": "49c009ae45f546038ca87658bffca11b",

"clusterid": "cfa1413f4d2e43b3952b8f481263b7fe",

"rootvolumeid": "24c206e71bc04a028b55da1a56e6761f",

"specid": "df032c1e2dfd4200a61f9b69b2412c2e",

"defaultl3networkid": "f7e0902c57e84628bfa8b8efae57e95c",

"cpunum": 4,

"cpuspeed": 0,

"memorysize": 5368709120,

"allocatorstrategy": 1,

"name": "测试告警--请不要操作",

"remark": "",

"type": "KVM",

"state": "running",

"platform": "Linux",

"creater": "3492a0f8b8d44673a68a797277fc5425",

"creationtime": "2021-09-07 13:52:07",

"modifier": "3492a0f8b8d44673a68a797277fc5425",

"modificationtime": "2021-09-21 08:58:22",

"hostid": "5c72736ac7864a4489c379b9513811db",

"oldhostid": "8112a306b9054ad496617ba4a1ca2bf1",

"consolepassword": "",

"sshkey": "",

"userdata": "",

"vmcpupining": null,

"vmconsolemode": 1,

"nextconsolemode": 1,

"usbredirect": false,

"allocatorstrategy2": 1,

"maxvmcount": null,

"strategymodel": 1,

"projectid": "3492a0f8b8d44673a68a797277fc5425",

"agid": "",

"consoleport": 5902,

"deleted": false,

"bootorder": null,

"vmlevel": 1,

"maxmemorysize": 5120,

"hostlevel": 1,

"migrate": true,

"autoscalinggroupid": null,

"ownertype": 1,

"rdpmode": null,

"servicetype": 1,

"owner": "Admin",

"mname": "nginx",

"biosmodel": "Legacy",

"mformat": "qcow2",

"projectname": null,

"domainname": "物流园西区A502",

"clustername": "A502-机架H6-UI重构",

"diskspecname": null,

"comspecname": "4c4g-2",

"pmname": "vm.k8s-node-33",

"hostip": "192.168.20.33",

"oldpmname": "vm.k8s-node-32",

"oldhostip": "192.168.20.32",

"defaultIp": null,

"currentNetworkUsedIp": null,

"vmnicInfList": null,

"memorysizeStr": "5.00 GB",

"unloadDisk": true,

"lableName": [ ],

"inBackup": false,

"createdBackup": false,

"rootVolumeMsType": 3

}]

}

# 异步API

所有不使用GET方法的API都为异步API。对于任务类API，HTTP Header 中返回任务的id和轮询任务结果的请求地址, body 中返回请求API本身的结果。例如：

POST

Status Code: 200

Header:

X-TASK-POLL-URL: /api-task/4ec5fd1eda604fbe8fcbef5241e56b5e

X-TASK-UUID: 4ec5fd1eda604fbe8fcbef5241e56b5e

Body:

{

"code": 1,

"msg": "success",

"id": "42e2c1b1bae242a1a519daf967afbe1a",

"userId": null,

"userToken": null,

"wsEndpoint": null,

"sessionToken": null,

"userName": null,

"roleName": null,

"userType": 0,

"projectIdList": null,

"basePath": null,

"projectId": null,

"projectName": null,

"guide": null,

"resourceIds": null,

"error": null,

"async": 1,

"serverTime": null,

"zoneId": null

}

# 使用access key调用API接口

# 获取access token

首先，调用方通过系统中分配的access key id 和access key secret 生成校验信息，可以采用下面例子中的python脚本。

命令格式 python get\_accesskeysign.py accesskeyid accesskeysecret

python get\_accesskeysign.py "sdjFdsyXZUmSjCf6mYIB" "sqwefYvswmnEV3X1KWNs1WfsdwefBIIphmI0rX9S"

args: Namespace(accesskey\_id='sdjFdsyXZUmSjCf6mYIB', accesskey\_secret='sqwefYvswmnEV3X1KWNs1WfsdwefBIIphmI0rX9S')

Signature: 8RfGvB4YnkrLqzTaL+cuhvlwYLs=

get\_accesskeysign.py

import base64

import argparse

from hashlib import sha1

parser = argparse.ArgumentParser(description='calculate access token sign.')

parser.add\_argument('accesskey\_id')

parser.add\_argument('accesskey\_secret')

args = parser.parse\_args()

print "args: %s" % args

str = args.accesskey\_id + args.accesskey\_secret

Signature = base64.b64encode(sha1(str.encode('utf-8')).digest())

print "Signature: %s" % Signature

之后通过请求接口获取access token

|  |
| --- |
| curl -X GET \  "http://192.168.100.185:38080/accessToken.do?accessKeyId=sdjFdsyXZUmSjCf6mYIB&signature=8RfGvB4YnkrLqzTaL%2BcuhvlwYLs%3D" |

返回结果

{

"accessToken": "5sHF/5lt5kDxN4l5c7qEQC3kYEE=",

"code": 1,

"data": null,

"error": null,

"expireTime": 1590151898,

"msg": null,

"projectIds": [],

"success": true,

"zoneIds": [

"a82d7717e24042c79c125147101dfe60",

"8baf5a89373946acaf05a04a13eb1e7e"

]

}

注意：

如果返回的信息中zoneIds或者projectIds为元素大于1的数组，需要在调用其他接口之前，先调用changeZone或者changeProject接口，先设置当前的一个区域或者项目

# 使用access token生成请求校验信息

可以采用下面get\_tokensign.py脚本生成每次调接口的校验信息。

命令格式 python get\_tokensign.py access\_token accesskey\_secret method(大写) uri（不需要包括查询参数部分）

python get\_tokensign.py "5sHF/5lt5kDxN4l5c7qEQC3kYEE=" "sqwefYvswmnEV3X1KWNs1WfsdwefBIIphmI0rX9S" GET /cloudHost/query.do

args: Namespace(access\_token='5sHF/5lt5kDxN4l5c7qEQC3kYEE=', accesskey\_secret='sqwefYvswmnEV3X1KWNs1WfsdwefBIIphmI0rX9S', method='GET', uri='/cloudHost/query.do')

Signature: DIrj3WV9A4XeHzfgiDtjW42M5fc=

Authoration 5sHF/5lt5kDxN4l5c7qEQC3kYEE=:DIrj3WV9A4XeHzfgiDtjW42M5fc=

get\_tokensign.py

import base64

import argparse

from hashlib import sha1

parser = argparse.ArgumentParser(description='calculate request sign.')

parser.add\_argument('access\_token')

parser.add\_argument('accesskey\_secret')

parser.add\_argument('method')

parser.add\_argument('uri')

args = parser.parse\_args()

print "args: %s" % args

str = args.accesskey\_secret + args.method + args.uri

signature = base64.b64encode(sha1(str.encode('utf-8')).digest())

print "Signature: %s" % signature

print "Authoration %s" % (args.access\_token + ":" + signature)

# 调用接口

|  |
| --- |
| curl -X GET \  -H "X-AUTH-TOKEN:5sHF/5lt5kDxN4l5c7qEQC3kYEE=:o+jCIBOpXyqsMfJIPxnjvUvb2MY=" \  "http://192.168.100.185:38080/cloudHost/query.do?order=asc&offset=0&limit=20&searchtype=0" |

返回结果：

{

"code": 1,

"msg": null,

"id": null,

"userId": null,

"userToken": null,

"wsEndpoint": null,

"sessionToken": null,

"userName": null,

"roleName": null,

"userType": 0,

"projectIdList": null,

"basePath": null,

"projectId": null,

"projectName": null,

"guide": null,

"resourceIds": null,

"error": null,

"async": 0,

"serverTime": null,

"zoneId": null,

"total": 36,

"rows": [

{

"vmid": "0cedc6fa673643bf875624f8428b32f0",

"mirrorid": "d39aba3d56f54d0a8f33ed4409f7de0c",

"domainid": "49c009ae45f546038ca87658bffca11b",

"clusterid": "cfa1413f4d2e43b3952b8f481263b7fe",

"rootvolumeid": "24c206e71bc04a028b55da1a56e6761f",

"specid": "df032c1e2dfd4200a61f9b69b2412c2e",

"defaultl3networkid": "f7e0902c57e84628bfa8b8efae57e95c",

"cpunum": 4,

"cpuspeed": 0,

"memorysize": 5368709120,

"allocatorstrategy": 1,

"name": "测试告警--请不要操作",

"remark": "",

"type": "KVM",

"state": "running",

"platform": "Linux",

"creater": "3492a0f8b8d44673a68a797277fc5425",

"creationtime": "2021-09-07 13:52:07",

"modifier": "3492a0f8b8d44673a68a797277fc5425",

"modificationtime": "2021-09-21 08:58:22",

"hostid": "5c72736ac7864a4489c379b9513811db",

"oldhostid": "8112a306b9054ad496617ba4a1ca2bf1",

"consolepassword": "",

"sshkey": "",

"userdata": "",

"vmcpupining": null,

"vmconsolemode": 1,

"nextconsolemode": 1,

"usbredirect": false,

"allocatorstrategy2": 1,

"maxvmcount": null,

"strategymodel": 1,

"projectid": "3492a0f8b8d44673a68a797277fc5425",

"agid": "",

"consoleport": 5902,

"deleted": false,

"bootorder": null,

"vmlevel": 1,

"maxmemorysize": 5120,

"hostlevel": 1,

"migrate": true,

"autoscalinggroupid": null,

"ownertype": 1,

"rdpmode": null,

"servicetype": 1,

"owner": "Admin",

"mname": "nginx",

"biosmodel": "Legacy",

"mformat": "qcow2",

"projectname": null,

"domainname": "物流园西区A502",

"clustername": "A502-机架H6-UI重构",

"diskspecname": null,

"comspecname": "4c4g-2",

"pmname": "vm.k8s-node-33",

"hostip": "192.168.20.33",

"oldpmname": "vm.k8s-node-32",

"oldhostip": "192.168.20.32",

"defaultIp": null,

"currentNetworkUsedIp": null,

"vmnicInfList": null,

"memorysizeStr": "5.00 GB",

"unloadDisk": true,

"lableName": [ ],

"inBackup": false,

"createdBackup": false,

"rootVolumeMsType": 3

}]

}

# 使用SDK调用API接口

## 环境准备

Java sdk 兼容版本jdk8

Java开发工具，可以选择eclipse或者Intellij IDEA

下载sdk包

导入sdk依赖，sdk工程的maven依赖如下

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.12</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>com.squareup.okhttp3</groupId>

<artifactId>okhttp</artifactId>

<version>3.14.9</version>

</dependency>

<dependency>

<groupId>com.alibaba</groupId>

<artifactId>fastjson</artifactId>

<version>1.2.62</version>

</dependency>

<dependency>

<groupId>commons-codec</groupId>

<artifactId>commons-codec</artifactId>

<version>1.9</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>servlet-api</artifactId>

<version>2.5</version>

<scope>provided</scope>

</dependency>

将sdk导入到调用工程中

<dependency>

<groupId>cn.com.nexwise</groupId>

<artifactId>java-sdk</artifactId>

<version>2.0.0</version>

</dependency>

调用示例

|  |
| --- |
| package cn.com.nexwise.sdk.demo;  import cn.com.nexwise.sdk.common.ApiClient;  import cn.com.nexwise.sdk.common.NWXConfig;  import cn.com.nexwise.sdk.common.dto.AccessTokenResult;  import cn.com.nexwise.sdk.common.dto.ApiRequest;  import cn.com.nexwise.sdk.common.dto.PagingDataWrapper;  import cn.com.nexwise.sdk.common.task.AccessTokenTask;  import cn.com.nexwise.sdk.vm.dto.QueryVmInfo;  import cn.com.nexwise.sdk.vm.dto.QueryVmReq;  import cn.com.nexwise.sdk.vm.task.QueryVmsTask;  /\*\*  \* 演示如何通过sdk调用查询云主机列表信息  \*  \*/  public class QueryVmDemo {    public static void main(String[] args) {    String hostname = "请输入管理节点访问地址";  int port = 38080; //请输入管理节点访问端口;    // sdk基础配置  ApiClient.configure(  new NWXConfig.Builder()  // .setShowLog(true) 可以配置是否输出调试日志  .setHostname(hostname)  .setPort(port)  .build()  );    // 获取access token  String accessKeyId = "请输入系统分配的access key id";  String accessKeySecret = "请输入系统分配的access key secret";  String accessToken = "请求返回的access token";// token 需要至少每隔1小时重新获取一次  ApiRequest tokenReq = new ApiRequest();  tokenReq.setAccessKeyId(accessKeyId);  tokenReq.setAccessKeySecret(accessKeySecret);    AccessTokenResult tokenRes = new AccessTokenTask(tokenReq).query();  if(tokenRes.checkSuccess()) {  accessToken = tokenRes.getAccessToken();  }    // 使用access token 访问API接口    QueryVmReq req = new QueryVmReq();  req.setAccessToken(accessToken);  req.setAccessKeySecret(accessKeySecret);  req.setLimit(1);  QueryVmsTask task = new QueryVmsTask(req);  PagingDataWrapper<QueryVmInfo> res = task.query();    // 使用返回结果处理...  }  } |

注意：access token 需要定时刷新，最好在一小时内重新获取一次。

# access token 相关接口

## 获取access token

通过系统中分配的access key id 和access key secret，获取access token。

### 请求地址

|  |
| --- |
| GET /accessToken.do |

### 请求头

无

### 请求参数

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| accessKeyId | varchar | access key id |
| signature | varchar | access key id 和access key secret 生成签名信息 |

### 返回值

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| code | integer | 状态码:1=成功,0=失败 |
| msg (可选) | varchar |  |
| accessToken | varchar | 后续调用API使用的access token |
| expireTime | long | access token |
| zoneIds | List | 有权限操作的区域id，用于区域切换 |
| projectIds | List | 有权限操作的项目id，用于项目切换 |

### 返回示例

{

"accessToken": "JMXRfVow7abezLrD39yUCZ7lJo4=",

"code": 1,

"expireTime": 1632292371,

"projectIds": [ ],

"zoneIds": [

"49c009ae45f546038ca87658bffca11b",

"66de3d76f6284c51a77a1d8e9cd3c95f",

"8baf5a89373946acaf05a04a13eb1e7e"

]

}

### Java SDK示例

String accessKeyId = "MzQwMjVlMWQ2Mzg3NGVi";

ApiRequest req = new ApiRequest();

req.setAccessKeyId(accessKeyId);

req.setAccessKeySecret(accessKeySecret);

AccessTokenResult res = new AccessTokenTask(req).query();

## 刷新access token

定时刷新access token，避免access token过期后，调用过期。

### 请求地址

|  |
| --- |
| GET /refreshAccessToken.do |

### 请求头

X-AUTH-TOKEN: 认证token

### 请求参数

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| token | varchar | 需要刷新的access token |

### 返回值

|  |  |  |
| --- | --- | --- |
| **字段** | **类型** | **说明** |
| code | integer | 状态码:1=成功,0=失败 |
| msg (可选) | varchar |  |
| accessToken | varchar | 后续调用API使用的access token |
| expireTime | long | access token |
| zoneIds | List | 有权限操作的区域id，用于区域切换 |
| projectIds | List | 有权限操作的项目id，用于项目切换 |

### 返回示例

{

"accessToken": "JMXRfVow7abezLrD39yUCZ7lJo4=",

"code": 1,

"expireTime": 1632292371,

"projectIds": [ ],

"zoneIds": [

"49c009ae45f546038ca87658bffca11b",

"66de3d76f6284c51a77a1d8e9cd3c95f",

"8baf5a89373946acaf05a04a13eb1e7e"

]

}

### Java SDK示例

RefreshAccessTokenReq req = new RefreshAccessTokenReq();

req.setAccessToken(accessToken);

req.setAccessKeySecret(accessKeySecret);

req.setToken(accessToken);

RefreshAccessTokenTask task = new RefreshAccessTokenTask(req);

AccessTokenResult res = task.query();