

Alibaba Cloud Elastic Compute Service SDK Reference

Issue: 20190507

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 .
<code>Courier font</code>	It is used for commands.	Run the <code>cd / d C :/ windows</code> command to enter the Windows system folder.
<i>Italics</i>	It is used for parameters and variables.	<code>bae log list --instanceid <i>Instance_ID</i></code>
[] or [a b]	It indicates that it is an optional value, and only one item can be selected.	<code>ipconfig [-all -t]</code>

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

Contents

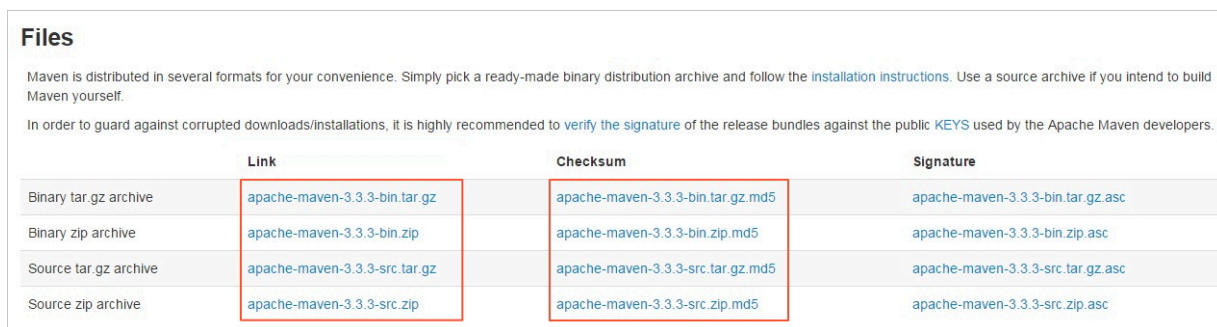
Legal disclaimer.....	I
Generic conventions.....	I
1 Install the new Java SDK.....	1
2 Use API to run ECS.....	5
2.1 Create multiple instances at a time.....	5
2.2 Create an ECS instance.....	9
2.3 Manage instances.....	15
2.4 Release an instance.....	19
2.5 Renew an instance.....	24
2.6 Use APIs to manage preemptible instances.....	31
2.7 Query available resources for configuration changes.....	39

1 Install the new Java SDK

Earlier versions of Java SDKs were downloadable as standalone packages, while the new Alibaba Cloud Java SDKs are distributed through Maven. All the Java SDKs of Alibaba Cloud are located in the same Maven repository for easy management.

Download Maven

Go to the [official Maven download page](#), and download the Maven software that corresponds to your operating system. You can verify the downloaded files by checking the Checksum files. The following screenshot is from the Windows 7 64-bit operating system, and the development environment is Eclipse Luna.



The screenshot shows the 'Files' section of the Maven download page. It contains a table with three columns: Link, Checksum, and Signature. The table lists four file formats: Binary tar.gz archive, Binary zip archive, Source tar.gz archive, and Source zip archive. The Link and Checksum columns are highlighted with red boxes in the original image.

	Link	Checksum	Signature
Binary tar.gz archive	apache-maven-3.3.3-bin.tar.gz	apache-maven-3.3.3-bin.tar.gz.md5	apache-maven-3.3.3-bin.tar.gz.asc
Binary zip archive	apache-maven-3.3.3-bin.zip	apache-maven-3.3.3-bin.zip.md5	apache-maven-3.3.3-bin.zip.asc
Source tar.gz archive	apache-maven-3.3.3-src.tar.gz	apache-maven-3.3.3-src.tar.gz.md5	apache-maven-3.3.3-src.tar.gz.asc
Source zip archive	apache-maven-3.3.3-src.zip	apache-maven-3.3.3-src.zip.md5	apache-maven-3.3.3-src.zip.asc

Install Java SDKs

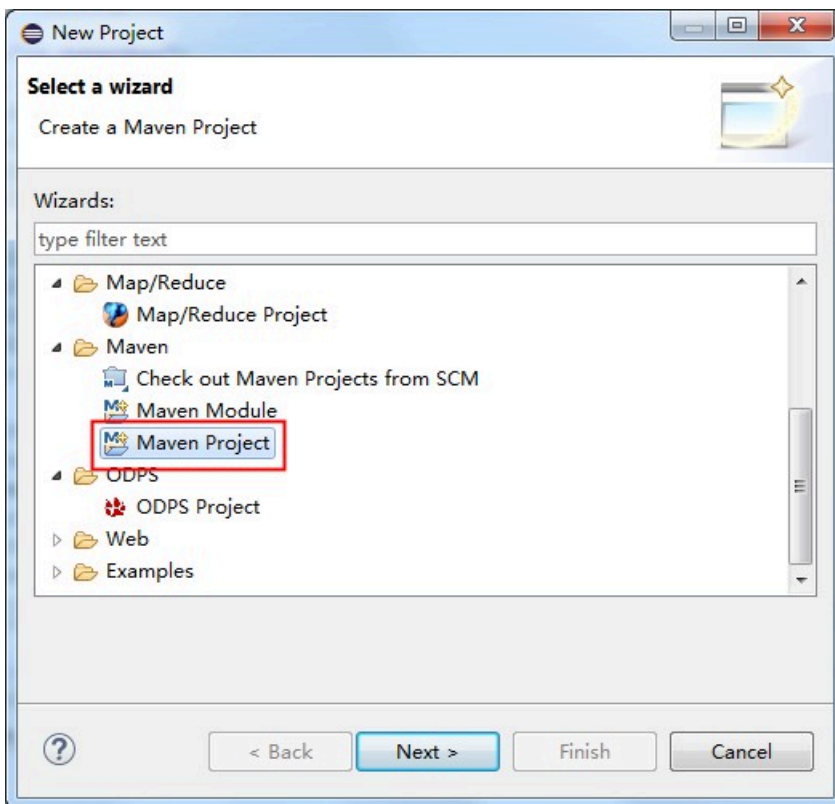
1. Go to the [Java SDK download page](#), and add the Maven repository storing Alibaba Cloud SDKs to the Maven software.
2. Decompress the downloaded Maven package, and add the following Maven repository information to the settings.xml file in the conf directory.

```
< repositories >
  < repository >
    < id > sonatype - nexus - staging </ id >
    < name > Sonatype Nexus Staging </ name >
    < url > https :// oss . sonatype . org / service / local / staging
/ deploy / maven2 /</ url >
    < releases >
      < enabled > true </ enabled >
    </ releases >
    < snapshots >
      < enabled > true </ enabled >
    < snapshots >
  </ repository >
```

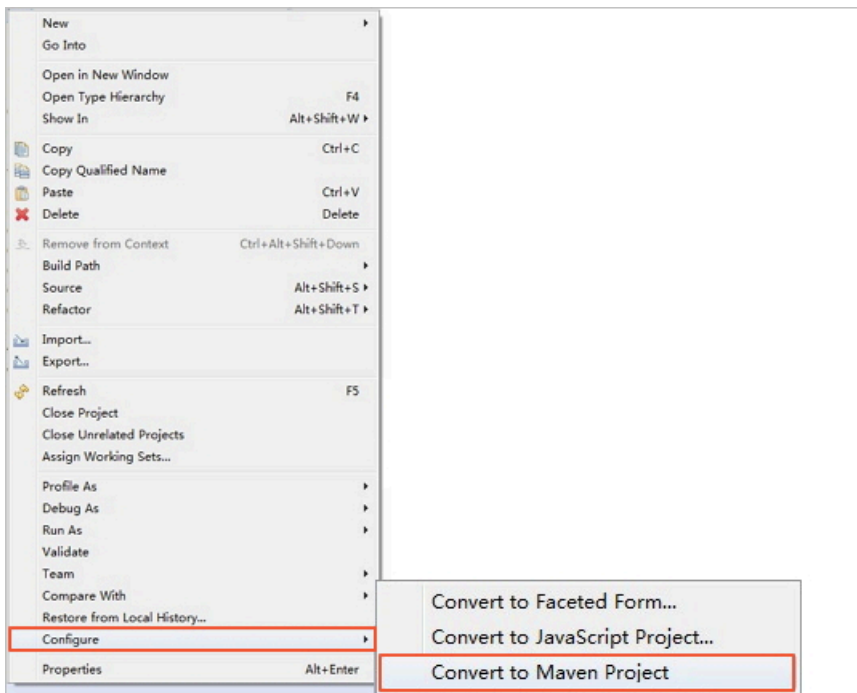
</ repositories >

3. Create a Maven project in either of the following ways:

- Method 1: Add a Maven project to Eclipse.

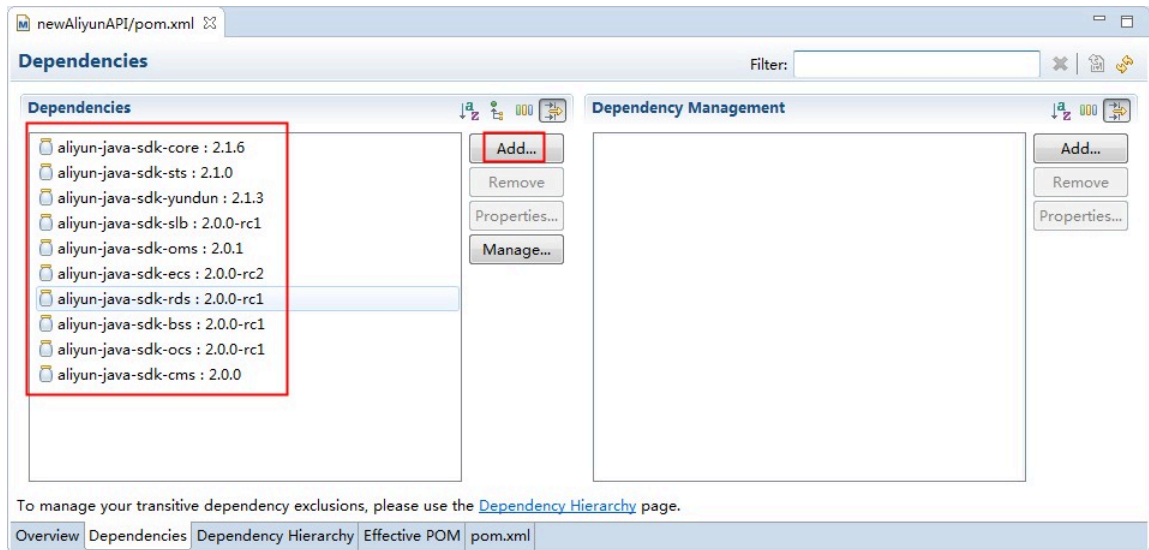


- Method 2: Convert an existing project to a Maven project.



4. Add dependencies to the Maven project in either of the following ways:

- **Method 1: Add the selected dependencies through the graphical interface.**

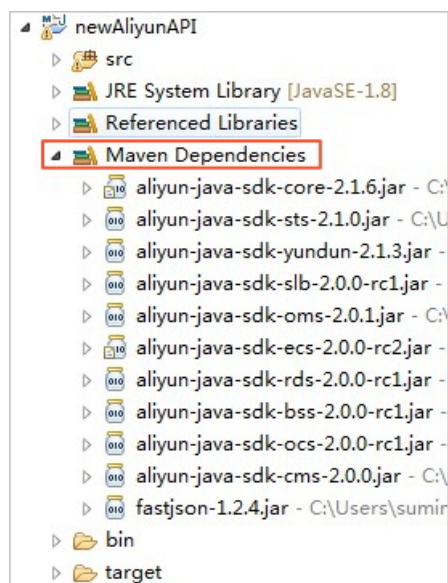


- **Method 2: Open the `pom.xml` file under the maven project, and add information similar to the following: You can add different dependencies as needed.**

```
< dependencies >
  < dependency >
    < groupId > com . aliyun </ groupId >
    < artifactId > aliyun - java - sdk - core </ artifactId >
    < version > 2 . 1 . 6 </ version >
  </ dependency >
  < dependency >
    < groupId > com . aliyun </ groupId >
    < artifactId > aliyun - java - sdk - sts </ artifactId >
    < version > 2 . 1 . 0 </ version >
  </ dependency >
  < dependency >
    < groupId > com . aliyun </ groupId >
    < artifactId > aliyun - java - sdk - yundun </ artifactId >
    < version > 2 . 1 . 3 </ version >
  </ dependency >
  < dependency >
    < groupId > com . aliyun </ groupId >
    < artifactId > aliyun - java - sdk - slb </ artifactId >
    < version > 2 . 0 . 0 - rc1 </ version >
  </ dependency >
</ dependencies >
```

```
</dependencies>
```

5. Save the changes. The corresponding Alibaba Cloud SDK JAR packages are automatically downloaded and added to the Maven dependencies.



How to distinguish old SDKs from new SDKs

If you are using an earlier version of SDK, we recommend that you switch to the latest version to explore its new features. The following table shows the differences between old SDKs and new SDKs.

Item	Earlier SDK	New SDKs
How to submit a request	execute()	getAcsResponse()
How to store the classes of AccessKey and AccessKeySecret	AliyunClient	IClientProfile
How to store the credential objects	new DefaultAliyunClient(APIUrl, Access Key, Access Key Secret)	DefaultProfile. getProfile(RegionId, Access Key, Access Key Secret)
Package prefix	com.aliyun.api	com.aliyuncs

2 Use API to run ECS

2.1 Create multiple instances at a time

`RunInstances` can create multiple ECS instances at a time. It helps you fast develop and deploy application.

Compared with `CreateInstance`, `RunInstances` has the following benefits:

- `RunInstances` contains `Amount` to create and automatically run up to 100 instances or preemptible instances for one request.
- When an instance is created, the instance status automatically becomes `Starting` and then `Running`. You do not need to call the `StartInstance` operation.
- Instances have Internet IPs allocated if you set the value of `InternetMaxBandwidthOut` greater than 0.
- You can also create 100 *preemptible instances* at a time to fully meet your requirements.
- Release plan can be scheduled by setting `AutoReleaseTime`, and the number of created parameters can be set by configuring `Amount`. The error codes and available parameters of `RunInstances` are completely compatible with `CreateInstance`.
- *Status polling of the created instances* is allowed since `InstanceIdSets` lists all the `InstanceIds` after the request.

Prerequisites

Make sure you have created an [AccessKey](#).



Notice:

Do not use the `AccessKey` of the primary account. If it is disclosed, your resources may be unsafe. Use the `AccessKey` of an RAM user account to reduce the risk of `AccessKey` disclosure.

Install ECS Python SDK

Make sure that you have runtime for Python. In this document, we take a version later than Python 2.7

as an example and the [SDK version](#) is 4.4.3.

```
pip install aliyun-python-sdk-ecs
```

If you get any message indicating that you have no operation permission, switch to `sudo`.

```
sudo pip install aliyun-python-sdk-ecs
```

The version of the SDK used in this article is 4.4.3. If you are using an older version of the SDK, update it.

Create instances

Create `RunInstancesRequest` object and then enter the related parameters:

In this example, we create two instances and specify to automatically check the instance status every 10 seconds. The creation procedure ends when the instance status turns into `Running`.

```
# your access key Id
ak_id = " YOU_ACCESS_KEY_ID "
# your access key secret
ak_secret = " YOU_ACCESS_SECRET "
region_id = " cn-beijing "
# your expected instance type
instance_type = " ecs.n4.small "
# The selected vswitchId
vswitch_id = " vsw-xxxxx "
# The selected image info
image_id = " centos_7_0_3_64_20G_alibase_20170818.vhd "
# The selected security group of VPC network
security_group_id = " sg-xxxxx "
# instance number to launch, support 1 - 100, default
# value is 100
amount = 2;
# The auto release time is in accordance with ISO8601
# and must be UTC. The format is `yyyy-MM-ddTHH:mm:ssZ`.
# The release time must be at least 30 minutes later than the current time and less than 3
# years from the current time.
auto_release_time = " 2017-12-05T22:40:00Z "
clt = client.AcsClient(ak_id, ak_secret, 'cn-beijing')
# create instance automatic running
def batch_create_instance():
    request = build_request()
    request.set_Amount(amount)
    _execute_request(request)
def _execute_request(request):
    response = _send_request(request)
    if response.get('Code') is None:
        instance_ids = response.get('InstanceIdSets').get(
            'InstanceIdSet')
        running_amount = 0
        while running_amount < amount:
            time.sleep(10)
```

```

        running_amount = check_instance_running (
instance_ids )
    print (" ecs instance %s is running ", instance_ids )
def check_instance_running ( instance_ids ):
    request = DescribeInstancesRequest ()
    request . set_InstanceIds ( json . dumps ( instance_ids ))
    response = _send_request ( request )
    if response . get ( ' Code ' ) is None :
        instances_list = response . get ( ' Instances ' ) . get ( '
Instance ' )
        running_count = 0
        for instance_detail in instances_list :
            if instance_detail . get ( ' Status ' ) == " Running " :
                running_count += 1
        return running_count
def build_request ():
    request = RunInstancesRequest ()
    request . set_ImageId ( image_id )
    request . set_VSwitchId ( vswitch_id )
    request . set_SecurityGroupId ( security_group_id )
    request . set_InstanceName ( " Instance12 - 04 " )
    request . set_InstanceType ( instance_type )
    return request
# send open api request
def _send_request ( request ):
    request . set_accept_format ( ' json ' )
    try :
        response_str = clt . do_action ( request )
        logging . info ( response_str )
        response_detail = json . loads ( response_str )
        return response_detail
    except Exception as e :
        logging . error ( e )

```

Create instances with Internet IP

[Create instances](#) and add a line of attribute to specify Internet bandwidth. In this example, we assign 1 Mbit/s bandwidth for each instance.

```

# create instance with public ip .
def batch_create_instance_with_public_ip ():
    request = build_request ()
    request . set_Amount ( amount )
    request . set_InternetMaxBandwidthOut ( 1 )
    _execute_request ( request )

```

Create instances with auto release time

[Create instances](#) and add a line of attribute to specify auto release time for instances. The auto release time is in accordance with [ISO8601](#) and must be UTC. The format is `YYYY - MM - DDTHH : mm : ssZ` . The release time cannot be 30 minutes earlier than the current time or more than 3 years from the current time.

```

# create instance with auto release time .
def batch_create_instance_with_auto_release_time ():
    request = build_request ()
    request . set_Amount ( amount )

```

```
request.set_AutoReleaseTime(auto_release_time)
_execute_request(request)
```

Complete example code

The complete example code is as follows.

```
# coding = utf - 8
# if the python sdk is not install using 'sudo pip
install aliyun - python - sdk - ecs '
# if the python sdk is install using 'sudo pip
install -- upgrade aliyun - python - sdk - ecs '
# make sure the sdk version is 4 . 4 . 3 , you can
use command ' pip show aliyun - python - sdk - ecs ' to
check
import json
import logging
import time
from aliyunsdkcore import client
from aliyunsdkecs.request.v20140526.DescribeInstancesRequest import DescribeInstancesRequest
from aliyunsdkecs.request.v20140526.RunInstancesRequest import RunInstancesRequest
logging.basicConfig(level=logging.INFO,
                    format='%(asctime)s %(filename)s [line:
%(lineno)d] %(levelname)s %(message)s ',
                    datefmt='%a, %d %b %Y %H:%M:%S')
# your access key Id
ak_id = "YOU_ACCESS_KEY_ID"
# your access key secret
ak_secret = "YOU_ACCESS_SECRET"
region_id = "cn-beijing"
# your expected instance type
instance_type = "ecs.n4.small"
# The selected vswitchId.
vswitch_id = "vsw-xxxxx"
# The selected image info
image_id = "centos_7_0_3_64_20G_alibase_20170818.vhd"
# The selected security group of VPC network
security_group_id = "sg-xxxxx"
# instance number to launch, support 1 - 100, default
value is 100
amount = 2;
# The auto release time is in accordance with ISO8601
and must be UTC. The format is `YYYY-MM-DDTHH
:mm:ssZ`. The release time must be at least 30
minutes later than the current time and less than 3
years from the current time.
auto_release_time = "2017-12-05T22:40:00Z"
clt = client.AcsClient(ak_id, ak_secret, 'cn-beijing')
# create instance automatic running
def batch_create_instance():
    request = build_request()
    request.set_Amount(amount)
    _execute_request(request)
# create instance with public ip.
def batch_create_instance_with_public_ip():
    request = build_request()
    request.set_Amount(amount)
    request.set_InternetMaxBandwidthOut(1)
    _execute_request(request)
# create instance with auto release time.
```



```

def batch_create_instance_with_auto_release_time():
    request = build_request()
    request.set_Amount(amount)
    request.set_AutoReleaseTime(auto_release_time)
    _execute_request(request)
def _execute_request(request):
    response = _send_request(request)
    if response.get('Code') is None:
        instance_ids = response.get('InstanceIdSets').get(
            'InstanceIdSet')
        running_amount = 0
        while running_amount < amount:
            time.sleep(10)
            running_amount = check_instance_running(
                instance_ids)
        print("ecs instance %s is running", instance_ids)
def check_instance_running(instance_ids):
    request = DescribeInstancesRequest()
    request.set_InstanceIds(json.dumps(instance_ids))
    response = _send_request(request)
    if response.get('Code') is None:
        instances_list = response.get('Instances').get(
            'Instance')
        running_count = 0
        for instance_detail in instances_list:
            if instance_detail.get('Status') == "Running":
                running_count += 1
        return running_count
def build_request():
    request = RunInstancesRequest()
    request.set_ImageId(image_id)
    request.set_VSwitchId(vswitch_id)
    request.set_SecurityGroupId(security_group_id)
    request.set_InstanceName("Instance12-04")
    request.set_InstanceType(instance_type)
    return request
# send open api request
def _send_request(request):
    request.set_AcceptFormat('json')
    try:
        response_str = clt.do_action(request)
        logging.info(response_str)
        response_detail = json.loads(response_str)
        return response_detail
    except Exception as e:
        logging.error(e)
if __name__ == '__main__':
    print "hello ecs batch create instance "
    # batch_create_instance()
    # batch_create_instance_with_public_ip()
    # batch_create_instance_with_auto_release_time()

```

2.2 Create an ECS instance

This topic describes how to create and manage an Alibaba Cloud ECS instance (either a Pay-As-You-Go or Subscription instance) by using Alibaba Cloud APIs.

The APIs described in this topic allow you to:

- [Create an ECS instance](#)
- [Query the status of an ECS instance](#)
- [Start an ECS instance](#)
- [Stop an ECS instance](#)
- [Allocate a public IP address](#)

Create a Pay-As-You-Go ECS instance

Required parameters

- `SecurityGroupId` : Security group ID. A security group is similar to a firewall and uses security group rules to control network access requests of instances. We recommend that you configure access rules only according to the actual needs. For more information, see [CreateSecurityGroup](#).
- `InstanceType` : Instance type. The option “one-core 2GiB n1.small” indicates that the input parameter is `ecs.n1.small`. For more information, see [Instance type families](#).
- `ImageId` : Image ID. You can use public images or custom images. For more information, see [DescribeImages](#).

Step 1: Create an ECS instance

The following code example shows the parameters `optimized` and `cloud_ssd`, which indicate the creation of an I/O optimized VPC instance with SSD as the system disk. For more information about other supported request parameters, see [CreateInstance](#).

```
# create one postpaid ecs instance .
def create_after_pay_instance ( image_id , instance_type ,
security_group_id ):
    request = CreateInstanceRequest ()
    request . set_ImageId ( image_id )
    request . set_SecurityGroupId ( security_group_id )
    request . set_InstanceType ( instance_type )
    request . set_IoOptimized ( ' optimized ' )
    request . set_VSwitchId ( ' vsw - vswitchid ' )
    request . set_VSwitchId ( ' vsw - vswitchid ' )
    request . set_SystemDiskCategory ( ' cloud_ssd ' )
    response = _send_request ( request )
    instance_id = response . get ( ' InstanceId ' )
    logging . info ( " instance %s created task submit
successfully .", instance_id )
```

```
return instance_id ;
```

After an ECS instance is created, the corresponding instance ID and request ID are returned as follows:

```
{" InstanceId ":" i -***"," RequestId ":" 006C1303 - BAC5 - 48E5 -
BCDF - 7FD5C2E639 5D "}
```

If the creation of an ECS instance fails, a corresponding error code and error message are returned as follows:

```
{ " RequestId ": " 540CFF28 - 407A - 40B5 - B6A5 - 74Bxxxxxxxx xx
", " HostId ": " ecs . aliyuncs . com ", " Code ": " InvalidSec
urityGroup Id . NotFound ", " Message ": " The specified
SecurityGr oupId does not exist ." }
```

Step 2: Query ECS instance status

Before you query the status of an ECS instance, note the following:

- The `StartInstance` operation can be performed on `Stopped` instances only.
- The `StopInstance` operation can be performed on `Running` instances only.



Note:

For more information about different ECS status, see [ECS instance lifecycle](#).

To query the status of a specific ECS instance, you can specify its `InstanceId`.

Specifically, when using `DescribeInstances`, you can pass the parameters in the form of a JSON array to query the instance status. The following code queries instance details:

```
# output the instance owned in current region .
def get_instance_detail_by_id ( instance_id , status ='
Stopped '):
    logging . info ( " Check instance %s status is %s ",
instance_id , status )
    request = DescribeInstancesRequest ()
    request . set_InstanceIds ( json . dumps ( [ instance_id ]))
    response = _send_request ( request )
    instance_detail = None
    if response is not None :
        instance_list = response . get ( ' Instances ' ) . get ( '
Instance ' )
        for item in instance_list :
            if item . get ( ' Status ' ) == status :
                instance_detail = item
                break ;
    return instance_detail ;
```

Step 3: Start an ECS instance

After an ECS instance is created, the default instance status is Stopped. To change the instance to the Running status, call `StartInstance` :

```
def start_instance ( instance_id ):
    request = StartInstanceRequest ()
    request . set_InstanceId ( instance_id )
    _send_request ( request )
```

(Optional) Step 4: Stop an ECS instance

To stop an ECS instance, call `StopInstance` :

```
def stop_instance ( instance_id ):
    request = StopInstanceRequest ()
    request . set_InstanceId ( instance_id )
    _send_request ( request )
```

Enable “ECS automatic startup” when creating an ECS instance

The ECS start and stop operations are asynchronous. You can use scripts to create an ECS instance, check its status, and perform the required operations. Specifically, the following code:

1. Checks whether the specified instance is in the Stopped status after obtaining an instance ID.
2. Calls `StartInstance` if it is in the Stopped status.
3. Waits for the instance to change to the Running status.

```
def check_instance_running ( instance_id ):
    detail = get_instance_detail_by_id ( instance_id =
instance_id , status = INSTANCE_RUNNING )
    index = 0
    while detail is None and index < 60 :
        detail = get_instance_detail_by_id ( instance_id =
instance_id );
        time . sleep ( 10 )
    if detail and detail . get ( ' Status ' ) == ' Stopped ':
        logging . info ( " instance %s is stopped now ." )
        start_instance ( instance_id = instance_id )
        logging . info ( " start instance %s job submit ." )
    detail = get_instance_detail_by_id ( instance_id =
instance_id , status = INSTANCE_RUNNING )
    while detail is None and index < 60 :
        detail = get_instance_detail_by_id ( instance_id =
instance_id , status = INSTANCE_RUNNING );
        time . sleep ( 10 )
    logging . info ( " instance %s is running now ." ,
instance_id )
    return instance_id ;
```

Allocate a public IP address

If you specify the Internet bandwidth when you create an ECS instance, you need to call an API to allocate a public IP address to the instance for Internet access. For more information, see [AllocatePublicIpAddress](#).

Create a Subscription ECS instance

Alibaba Cloud APIs also support the creation of Subscription ECS instances. If you want to create a Subscription ECS instance, make sure that you have a sufficient account balance or credit amount associated to your account so that the fees can be deducted directly during instance creation.

When you create a Subscription ECS instance, you only need to specify the payment option and duration. In the following code example, the billing duration of the instance is set to one month.

```
request . set_Period ( 1 )
request . set_InstanceChargeType ( ' PrePaid ' )
```

The complete code for the creation of a Subscription ECS instance is as follows:

```
# create one prepay ecs instance .
def create_pre_pay_instance ( image_id , instance_type ,
security_group_id ):
    request = CreateInstanceRequest ()
    request . set_ImageId ( image_id )
    request . set_SecurityGroupId ( security_group_id )
    request . set_InstanceType ( instance_type )
    request . set_IoOptimized ( ' optimized ' )
    request . set_SystemDiskCategory ( ' cloud_ssd ' )
    request . set_Period ( 1 )
    request . set_InstanceChargeType ( ' PrePaid ' )
    response = _send_request ( request )
    instance_id = response . get ( ' InstanceId ' )
    logging . info ( " instance %s created task submit
successfully .", instance_id )
    return instance_id ;
```

Example

The following example uses Python to describe the creation of an ECS instance.

```
# coding = utf - 8
# if the python sdk is not install using ' sudo pip
install aliyun - python - sdk - ecs '
# if the python sdk is install using ' sudo pip
install -- upgrade aliyun - python - sdk - ecs '
# make sure the sdk version is 2 . 1 . 2 , you can
use command ' pip show aliyun - python - sdk - ecs ' to
check
import json
import logging
import time
from aliyunsdkcore import client
```

```

from aliyunsdke cs . request . v20140526 . CreateInst
anceReques t import CreateInst anceReques t
from aliyunsdke cs . request . v20140526 . DescribeIn
stancesReq uest import DescribeIn stancesReq uest
from aliyunsdke cs . request . v20140526 . StartInsta nceRequest
import StartInsta nceRequest
# configurat ion the log output formatter , if you want
to save the output to file ,
# append ", filename = ' ecs_invoke . log '" after datefmt .
logging . basicConfi g ( level = logging . INFO ,
                        format = '%( asctime ) s %( filename ) s [ line :
%( lineno ) d ] %( levelname ) s %( message ) s ' ,
                        datefmt = '% a , % d % b % Y % H :% M :% S ' )
clt = client . AcscClient ( ' Your Access Key Id ' , ' Your
Access Key Secret ' , ' cn - beijing ' )
IMAGE_ID = ' ubuntu1404 _64_40G_cl oudinit_20 160727 . raw '
INSTANCE_T YPE = ' ecs . s2 . large ' # 2c4g generation 1
SECURITY_G ROUP_ID = ' sg -****'
INSTANCE_R UNNING = ' Running '
def create_inst ance_acti on () :
    instance_i d = create_aft er_pay_inst ance ( image_id =
IMAGE_ID , instance_t ype = INSTANCE_T YPE ,
                                                security_g roup_id =
SECURITY_G ROUP_ID )
    check_inst ance_runni ng ( instance_i d = instance_i d )
def create_pre pay_instan ce_acti on () :
    instance_i d = create_pre pay_instan ce ( image_id =
IMAGE_ID , instance_t ype = INSTANCE_T YPE ,
                                                security_g roup_id =
SECURITY_G ROUP_ID )
    check_inst ance_runni ng ( instance_i d = instance_i d )
# create one after pay ecs instance .
def create_aft er_pay_inst ance ( image_id , instance_t ype ,
security_g roup_id ) :
    request = CreateInst anceReques t () ;
    request . set_ImageI d ( image_id )
    request . set_Securi tyGroupId ( security_g roup_id )
    request . set_Instan ceType ( instance_t ype )
    request . set_IoOpti mized ( ' optimized ' )
    request . set_System DiskCatego ry ( ' cloud_ssd ' )
    response = _send_requ est ( request )
    instance_i d = response . get ( ' InstanceId ' )
    logging . info ( " instance %s created task submit
successfully ." , instance_i d )
    return instance_i d ;
# create one prepay ecs instance .
def create_pre pay_instan ce ( image_id , instance_t ype ,
security_g roup_id ) :
    request = CreateInst anceReques t () ;
    request . set_ImageI d ( image_id )
    request . set_Securi tyGroupId ( security_g roup_id )
    request . set_Instan ceType ( instance_t ype )
    request . set_IoOpti mized ( ' optimized ' )
    request . set_System DiskCatego ry ( ' cloud_ssd ' )
    request . set_Period ( 1 )
    request . set_Instan ceChargeTy pe ( ' PrePaid ' )
    response = _send_requ est ( request )
    instance_i d = response . get ( ' InstanceId ' )
    logging . info ( " instance %s created task submit
successfully ." , instance_i d )
    return instance_i d ;
def check_inst ance_runni ng ( instance_i d ) :
    detail = get_instan ce_detail_ by_id ( instance_i d =
instance_i d , status = INSTANCE_R UNNING )

```

```

    index = 0
    while detail is None and index < 60 :
        detail = get_instance_detail_by_id ( instance_id =
instance_id );
        time . sleep ( 10 )
        if detail and detail . get ( ' Status ' ) == ' Stopped ' :
            logging . info ( " instance % s is stopped now ." )
            start_instance ( instance_id = instance_id )
            logging . info ( " start instance % s job submit ." )
            detail = get_instance_detail_by_id ( instance_id =
instance_id , status = INSTANCE_RUNNING )
            while detail is None and index < 60 :
                detail = get_instance_detail_by_id ( instance_id =
instance_id , status = INSTANCE_RUNNING );
                time . sleep ( 10 )
            logging . info ( " instance % s is running now ." ,
instance_id )
            return instance_id ;
def start_instance ( instance_id ):
    request = StartInstanceRequest ()
    request . set_InstanceId ( instance_id )
    _send_request ( request )
# output the instance owned in current region .
def get_instance_detail_by_id ( instance_id , status = '
Stopped ' ):
    logging . info ( " Check instance % s status is % s " ,
instance_id , status )
    request = DescribeInstancesRequest ()
    request . set_InstanceIds ( json . dumps ( [ instance_id ] ) )
    response = _send_request ( request )
    instance_detail = None
    if response is not None :
        instance_list = response . get ( ' Instances ' ) . get ( '
Instance ' )
        for item in instance_list :
            if item . get ( ' Status ' ) == status :
                instance_detail = item
                break ;
    return instance_detail ;
# send open api request
def _send_request ( request ):
    request . set_Accept_Format ( ' json ' )
    try :
        response_str = clt . do_action ( request )
        logging . info ( response_str )
        response_detail = json . loads ( response_str )
        return response_detail
    except Exception as e :
        logging . error ( e )
if __name__ == '__main__' :
    logging . info ( " Create ECS by OpenApi ! " )
    create_instance_action ()
    # create_pay_instance_action ()

```

2.3 Manage instances

In addition to using the ECS console for resource creation and daily management, you can also use API to manage and customize resources. API allows you to manage and configure ECS instances with greater flexibility. Alibaba Cloud encapsulates API in an

SDK to integrate ECS instance management into existing systems. This topic describes how to manage ECS instances through API based on Python development. You can develop ECS instances easily even if you do not have Python development experience.

Get the AccessKey for a RAM user

An AccessKey (AccessKey ID and AccessKey Secret) is required when you want to use API to manage ECS instances. To keep your cloud service secure, you have to create a RAM user and generate an AccessKey for it, and authorize the RAM user to manage ECS resources only. Then, you can use the RAM user and its AccessKey to manage ECS resources by using API.

Follow these steps to get the AccessKey for a RAM user:

1. [Create a RAM user and get the AccessKey.](#)
2. [Grant permissions to the RAM user directly.](#) To manage ECS resources, you have to grant `AliyunECSFullAccess` to the RAM user.

Install the ECS Python SDK

Make sure that the Python runtime environment has been installed. This article uses Python 2.7+.

```
pip install aliyun-python-sdk-ecs
```

If you do not have the permission, switch to `sudo` to continue.

```
sudo pip install aliyun-python-sdk-ecs
```

The SDK version is 2.1.2.

Hello Alibaba Cloud

Create the file `hello_ecs_api.py`. To use SDK, you have to use the AccessKey of the RAM user to instantiate an `AcsClient` object.



Note:

The AccessKey allows the RAM user to access Alibaba Cloud APIs and give you full access to the user. Keep them safe.

```
from aliyunsdkcore import client
from aliyunsdkecs.request.v20140526 import DescribeInstancesRequest
from aliyunsdkecs.request.v20140526 import DescribeRegionsRequest
```



```
clt = client . AcscClient ( ' Your Access Key Id ', ' Your
Access Key Secret ', ' cn - beijing ' )
```

You can develop your first application after the `AcscClient` object is instantiated. Query the list of regions that your account supports. For more information, see [query the list of available regions](#).

```
def hello_aliy_un_regions ():
    request = DescribeRegionsRequest ()
    response = _send_request ( request )
    region_list = response . get ( ' Regions ' ) . get ( ' Region ' )
    assert response is not None
    assert region_list is not None
    result = map ( _print_region_id , region_list )
    logging . info ( " region list : % s " , result )
def _print_region_id ( item ):
    region_id = item . get ( " RegionId " )
    return region_id
def _send_request ( request ):
    request . set_accept_format ( ' json ' )
    try :
        response_str = clt . do_action ( request )
        logging . info ( response_str )
        response_detail = json . loads ( response_str )
        return response_detail
    except Exception as e :
        logging . error ( e )
hello_aliy_un_regions ()
```

In the command line, run `python hello_ecs_api . py` to obtain a list of supported regions. The output is similar to the following.

```
[ u ' cn - shenzhen ', u ' ap - southeast - 1 ', u ' cn - qingdao
', u ' cn - beijing ', u ' cn - shanghai ', u ' us - east - 1 ',
u ' cn - hongkong ', u ' me - east - 1 ', u ' ap - southeast - 2
', u ' cn - hangzhou ', u ' eu - central - 1 ', u ' ap - northeast
- 1 ', u ' us - west - 1 ' ]
```

Query the list of ECS instances in the current region

The process for querying the instance list is similar to the region list. You only need to replace the input parameter `DescribeRegionsRequest` with `DescribeInstancesRequest`. For a full list of query parameters, see [query an instance list](#).

```
def list_instances ():
    request = DescribeInstancesRequest ()
    response = _send_request ( request )
    if response is not None :
        instance_list = response . get ( ' Instances ' ) . get ( '
Instance ' )
        result = map ( _print_instance_id , instance_list )
        logging . info ( " current region include instance % s
", result )
    def _print_instance_id ( item ):
        instance_id = item . get ( ' InstanceId ' );
```

```
return instance_id
```

The output is as follows.

```
current region include instance [ u ' i -****', u ' i -
****']
```

For a full list of APIs, see [ECS API overview](#). If you want to [query a list of disks](#), replace `DescribeInstancesRequest` with `DescribeDisksRequest`.

Complete code

The following is the complete code of the operations described in this document.

```
# coding = utf - 8
# if the python sdk is not install using 'sudo pip
install aliyun - python - sdk - ecs '
# if the python sdk is install using 'sudo pip
install -- upgrade aliyun - python - sdk - ecs '
# make sure the sdk version is 2.1.2, you can
use command 'pip show aliyun - python - sdk - ecs ' to
check
import json
import logging
from aliynsdke ore import client
from aliynsdke cs . request . v20140526 . DescribeIn
stancesReq uest import DescribeIn stancesReq uest
from aliynsdke cs . request . v20140526 . DescribeRe
gionsReque st import DescribeRe gionsReque st
# configurat ion the log output formatter, if you want
to save the output to file,
# append ", filename = 'ecs_invoke . log '" after datefmt .
logging . basicConfi g ( level = logging . INFO ,
format = '%(asctime) s %(filename) s [ line :
%(lineno) d ] %(levelname) s %(message) s ',
datefmt = '% a , % d % b % Y % H :% M :% S ')
clt = client . AcsClient (' Your Access Key Id ', ' Your
Access Key Secret ', ' cn - beijing ')
# sample api to list aliyun open api .
def hello_aliy_un_regions ():
    request = DescribeRe gionsReque st ()
    response = _send_requ est ( request )
    if response is not None :
        region_lis t = response . get (' Regions '). get (' Region
')
        assert response is not None
        assert region_lis t is not None
        result = map (_print_reg ion_id , region_lis t )
        logging . info (" region list : % s ", result )
# output the instance owned in current region .
def list_insta nces ():
    request = DescribeIn stancesReq uest ()
    response = _send_requ est ( request )
    if response is not None :
        instance_l ist = response . get (' Instances '). get ('
Instance ')
        result = map (_print_ins tance_id , instance_l ist )
        logging . info (" current region include instance % s
", result )
def _print_ins tance_id ( item ):
```

```
instance_id = item.get('InstanceId');
return instance_id
def _print_region_id(item):
    region_id = item.get("RegionId")
    return region_id
# send open api request
def _send_request(request):
    request.set_accept_format('json')
    try:
        response_str = clt.do_action(request)
        logging.info(response_str)
        response_detail = json.loads(response_str)
        return response_detail
    except Exception as e:
        logging.error(e)
if __name__ == '__main__':
    logging.info("Hello Aliyun OpenApi!")
    hello_aliyun_regions()
    list_instances()
```

If you want to learn other API operations in ECS, see [ECS API operation](#).

2.4 Release an instance

One important feature of ECS is on-demand resource creation. You can create custom resources elastically on demand during peak service hours, and then release those resources after service computing is completed. This document describes how to easily release ECS instances and achieve elasticity.

This topic covers the following APIs:

- [DeleteInstance](#)
- [ModifyInstanceAutoReleaseTime](#)
- [StopInstance](#)
- [Instance list query API](#)

After an ECS instance is released, the physical resources used by the instance are recycled, including disks and snapshots. The data of the instance is completely lost and can never be recovered. If you want to retain the data, we recommend that you create snapshots of disks before releasing the ECS instance. The snapshots can be directly used to create a new ECS instance.

To release an ECS instance, you must stop it first. If any application is affected after the ECS instance is stopped, restart the instance.

Stop an ECS instance

Use the `StopInstance` interface to stop an ECS instance, regardless of the billing method of the instance. The stop command is as follows. When the `ForceStop` parameter is set to `true`, the ECS instance is stopped directly but data is not necessarily written to a disk, similar to power failure. Therefore, if you want to release an instance, set `ForceStop` to `true`.

```
def __stop_instance ( instance_id , force_stop = False ):
    """
    stop one ecs instance .
    : param instance_id : instance id of the ecs
    instance , like 'i-***'.
    : param force_stop : if force stop is true , it will
    force stop the server and not ensure the data
    write to disk correctly .
    : return :
    """
    request = StopInstanceRequest ( )
    request . set_InstanceId ( instance_id )
    request . set_ForceStop ( force_stop )
    logging . info ( " Stop %s command submit successfully
    .", instance_id )
    _send_request ( request )
```

Release an ECS instance

If you release an ECS instance when it is not in the Stopped status, an error occurs:

```
{" RequestId ":" 3C6DEAB4 - 7207 - 411F - 9A31 - 6ADE54C268 BE
", " HostId ":" ecs - cn - hangzhou . aliyuncs . com ", " Code ":"
IncorrectInstanceStatus ", " Message ":" The current status
of the resource does not support this operation ."}"
```

When the ECS instance is in the Stopped status, you can release it. The API has only two request parameters:

- **InstanceId:** Instance ID
- **Force:** If this parameter is set to “true” , the ECS instance is released forcibly even when it is not in the Stopped status. Use caution when setting this parameter. Release by mistake may affect your services.

The request to release an ECS instance is as follows.

```
def __release_instance ( instance_id , force = False ):
    """
    delete instance according instance id , only support
    after pay instance .
    : param instance_id : instance id of the ecs
    instance , like 'i-***'.
    : param force :
    if force is false , you need to make the ecs
    instance stopped , you can
```

```

    execute the delete action .
    If force is true , you can delete the instance
    even the instance is running .
    : return :
    '''
    request = DeleteInstanceRequest ()
    request . set_InstanceId ( instance_id )
    request . set_Force ( force )
    _send_request ( request )

```

The following response is returned when an ECS instance is released successfully:

```
{" RequestId ":" 689E5813 - D150 - 4664 - AF6F - 2A27BB4986 A3 "}
```

Set the automatic release time for an ECS instance

You can set the automatic release time for an ECS instance to simplify instance management. When the set time is reached, Alibaba Cloud releases your ECS instance automatically. Use the `ModifyInstanceAutoReleaseTime` to set the automatic release time for an ECS instance.



Note:

The automatic release time follows the ISO8601 standard in UTC time. The format is `yyyy-MM-ddTHH:mm:ssZ`. If the seconds place is not 00, it is automatically set to start from the current minute. The automatic release time must be at least half an hour later than the current time, and must not be more than 3 years since the current time.

```

def set_instance_auto_release_time ( instance_id ,
time_to_release = None ):
    '''
    setting instance auto delete time
    : param instance_id : instance id of the ecs
    instance , like 'i-***'.
    : param time_to_release : if the property is setting
    , such as '2017 - 01 - 30T00 : 00 : 00Z '
    it means setting the instance to be release at
    that time .
    if the property is None , it means cancel the
    auto delete time .
    : return :
    '''
    request = ModifyInstanceAutoReleaseTimeRequest ()
    request . set_InstanceId ( instance_id )
    if time_to_release is not None :
        request . set_AutoReleaseTime ( time_to_release )
    _send_request ( request )

```

Run the command `set_instance_auto_release_time ('i-1111', '2017 - 01 - 30T00 : 00 : 00Z')` to set the time.

Then you can use the `DescribeInstances` to query the automatic release time.

```
def describe_instance_detail ( instance_id ):
    """
    describe instance detail
    :param instance_id: instance id of the ecs
    instance, like 'i-***'.
    :return:
    """
    request = DescribeInstancesRequest ()
    request.set_InstanceIds ( json.dumps ( [ instance_id ] ))
    response = _send_request ( request )
    if response is not None:
        instance_list = response.get ( 'Instances' ).get ( '
Instance ' )
        if len ( instance_list ) > 0:
            return instance_list [ 0 ]
def check_auto_release_time_ready ( instance_id ):
    detail = describe_instance_detail ( instance_id =
instance_id )
    if detail is not None:
        release_time = detail.get ( 'AutoReleaseTime ' )
        return release_time
```

Cancel the automatic release

If you want to cancel the automatic release due to service changes, run the following command to set the automatic release time to null.

```
set_instance_auto_release_time ( ' i - 1111 ' )
```

Complete example code



Note:

Proceed with caution when releasing ECS instances.

```
# coding = utf - 8
# if the python sdk is not install using ' sudo pip
install aliyun - python - sdk - ecs '
# if the python sdk is install using ' sudo pip
install -- upgrade aliyun - python - sdk - ecs '
# make sure the sdk version is 2.1.2, you can
use command ' pip show aliyun - python - sdk - ecs ' to
check
import json
import logging
from aliyunsdkcore import client
from aliyunsdkecs.request.v20140526.DeleteInstanceRequest import DeleteInstanceRequest
from aliyunsdkecs.request.v20140526.DescribeInstancesRequest import DescribeInstancesRequest
from aliyunsdkecs.request.v20140526.ModifyInstanceAutoReleaseTimeRequest import \
ModifyInstanceAutoReleaseTimeRequest
from aliyunsdkecs.request.v20140526.StopInstanceRequest import StopInstanceRequest
```

```

# configuration the log output formatter, if you want
# to save the output to file,
# append ", filename = 'ecs_invoke.log'" after datefmt.
logging.basicConfig(level=logging.INFO,
                    format='%(asctime)s %(filename)s [line:
%(lineno)d] %(levelname)s %(message)s',
                    datefmt='%a, %d %b %Y %H:%M:%S')
clt = client.AcsClient('Your Access Key Id', 'Your
Access Key Secret', 'cn-beijing')
def _stop_instance(instance_id, force_stop=False):
    """
    stop one ecs instance.
    :param instance_id: instance id of the ecs
instance, like 'i-***'.
    :param force_stop: if force stop is true, it will
force stop the server and not ensure the data
write to disk correctly.
    :return:
    """
    request = StopInstanceRequest()
    request.set_InstanceId(instance_id)
    request.set_ForceStop(force_stop)
    logging.info("Stop %s command submit successfully
.", instance_id)
    _send_request(request)
def _describe_instance_detail(instance_id):
    """
    describe instance detail
    :param instance_id: instance id of the ecs
instance, like 'i-***'.
    :return:
    """
    request = DescribeInstancesRequest()
    request.set_InstanceIds(json.dumps([instance_id]))
    response = _send_request(request)
    if response is not None:
        instance_list = response.get('Instances').get('
Instance')
        if len(instance_list) > 0:
            return instance_list[0]
def check_auto_release_time_ready(instance_id):
    detail = _describe_instance_detail(instance_id =
instance_id)
    if detail is not None:
        release_time = detail.get('AutoReleaseTime')
        return release_time
def _release_instance(instance_id, force=False):
    """
    delete instance according instance id, only support
after pay instance.
    :param instance_id: instance id of the ecs
instance, like 'i-***'.
    :param force:
    if force is false, you need to make the ecs
instance stopped, you can
execute the delete action.
    If force is true, you can delete the instance
even the instance is running.
    :return:
    """
    request = DeleteInstanceRequest();
    request.set_InstanceId(instance_id)
    request.set_Force(force)
    _send_request(request)

```

```

def set_instance_auto_release_time ( instance_id ,
time_to_release = None ):
    """
    setting instance auto delete time
    : param instance_id : instance id of the ecs
instance , like 'i-***'.
    : param time_to_release : if the property is setting
, such as '2017-01-30T00:00:00Z'
it means setting the instance to be release at
that time .
if the property is None , it means cancel the
auto delete time .
    : return :
    """
    request = ModifyInstanceAutoReleaseTimeRequest ()
    request.set_InstanceId ( instance_id )
    if time_to_release is not None :
        request.set_AutoReleaseTime ( time_to_release )
    _send_request ( request )
    release_time = check_auto_release_time_ready (
instance_id )
    logging.info ( " Check instance %s auto release time
setting is %s . " , instance_id , release_time )
def _send_request ( request ):
    """
    send open api request
    : param request :
    : return :
    """
    request.set_AcceptFormat ( 'json' )
    try :
        response_str = clt.do_action ( request )
        logging.info ( response_str )
        response_detail = json.loads ( response_str )
        return response_detail
    except Exception as e :
        logging.error ( e )
if __name__ == '__main__':
    logging.info ( " Release ecs instance by Aliyun
OpenApi ! " )
    set_instance_auto_release_time ( 'i-1111' , '2017-01-
28T06:00:00Z' )
    # set_instance_auto_release_time ( 'i-1111' )
    # stop_instance ( 'i-1111' )
    # release_instance ( 'i-1111' )
    # release_instance ( 'i-1111' , True )

```

If you want to learn other API operations in ECS, see [ECS API operation](#).

2.5 Renew an instance

Lifecycle is important to ECS instances of the Subscription billing method. In addition to the ECS console or the ECS purchase page, Alibaba Cloud provides you with APIs to view the resource expiration time and renew your instance.

This topic involves the following key functions:

- Query ECS instances by expiration time.

- Renew instances.
- Query the automatic renewal time of an ECS instance.
- Set the automatic renewal time of an ECS instance.

Lifecycle is important to ECS instances in the Subscription mode. If you fail to renew your ECS instance on time, the instance may be locked or even released, thus affecting your service continuity. You can use APIs to view the resource expiration time and renew your instance.

This article covers the following APIs:

- [DescribeInstances](#)
- [ModifyInstanceAutoRenewAttribute](#)

Query the instances that will expire within the specified time range

Use the DescribeInstances interface to query the instances that will expire within the specified time range by setting the filter parameters `ExpiredStartTime` and `ExpiredEndTime`. The time parameters follow the ISO8601 standard in UTC time, using the format `yyyy-MM-ddTHH:mmZ`. The system returns a list of instances that will expire within the specified time range. If you want to filter by security group, add the security group ID.

```

INSTANCE_EXPIRED_START_TIME_IN_UTC_STRING = ' 2017 - 01 -
22T00 : 00Z '
INSTANCE_EXPIRE_END_TIME_IN_UTC_STRING = ' 2017 - 01 - 28T00 :
00Z '
def describe_expired_renew_instance ( page_size = 100 ,
page_number = 1 , instance_id = None ,
                                check_need_renew = True ,
security_group_id = None ):
    request = DescribeInstancesRequest ()
    if check_need_renew is True :
        request . set_Filter_3Key ( " ExpiredStartTime " )
        request . set_Filter_3Value ( INSTANCE_EXPIRED_STA
RT_TIME_IN_UTC_STRING )
        request . set_Filter_4Key ( " ExpiredEndTime " )
        request . set_Filter_4Value ( INSTANCE_EXPIRE_END_
TIME_IN_UTC_STRING )
    if instance_id is not None :
        request . set_InstanceIds ( json . dumps ( [ instance_id
]))
    if security_group_id :
        request . set_SecurityGroupId ( security_group_id )
    request . set_PageNumber ( page_number )
    request . set_PageSize ( page_size )

```

```
return _send_request ( request )
```

Renew ECS instances

Only ECS instances in the Subscription mode can be renewed. Pay-As-You-Go instances cannot be renewed. Renewals must be paid by account balance or credit. Fee deduction and order creation are in sync with API execution. Make sure that there is sufficient balance in your account.

```
def _renew_instance_action ( instance_id , period = ' 1 ' ):
    request = RenewInstanceRequest ()
    request . set_Period ( period )
    request . set_InstanceId ( instance_id )
    response = _send_request ( request )
    logging . info ( ' renew %s ready , output is %s ' ,
instance_id , response )
```

Fees are automatically deducted when the instance is renewed. After the renewal is completed, you can query the resource expiration time of the instance based on InstanceId . Because the API is executed asynchronously, the expiration time is updated in 10 seconds.

Enable automatic ECS instance renewal

Alibaba Cloud provides the [automatic renewal function](#) for ECS instances in the Subscription mode to help you reduce the cost of expired resource maintenance. Fee deduction for automatic renewal starts at 08:00:00 nine days before the expiration date. If fee deduction fails on the first day, the deduction process repeats on the following days in sequence until fees are deducted successfully or resources are locked after the nine-day period. Make sure that you have sufficient balance or credit amount in your account.

- Query automatic renewal setting

You can use OpenAPI to query and set automatic renewal. The API supports only ECS instances in the Subscription mode. If you use the API on a Pay-As-You-Go instance, an error is returned. You can query the automatic renewal status of up

to 100 ECS instances of the Subscription billing method at a time. Use commas to separate multiple instance IDs.

The input parameter of `DescribeInstanceAutoRenewAttribute` is the instance ID.

InstanceId: You can query up to 100 ECS instances in the Subscription mode at a time. Use commas to separate multiple instance IDs.

```
# check the instances is renew or not
def describe_auto_renew ( instance_ids , expected_auto_renew = True ):
    describe_request = DescribeInstanceAutoRenewAttributeRequest ()
    describe_request.set_InstanceIds ( instance_ids )
    response_detail = _send_request ( request = describe_request )
    failed_instance_ids = ''
    if response_detail is not None :
        attributes = response_detail.get ( ' InstanceRenewAttributes ' )
        if attributes :
            for item in attributes :
                auto_renew_status = item.get ( ' AutoRenewEnabled ' )
                if auto_renew_status != expected_auto_renew :
                    failed_instance_ids += item.get ( ' InstanceId ' ) + ','
    describe_auto_renew ( ' i - 1111 , i - 2222 ' )
```

The following content is returned:

```
{" InstanceRenewAttributes " : {" InstanceRenewAttribute " : [{" Duration " : 0 , " InstanceId " : " i - 1111 " , " AutoRenewEnabled " : false }, {" Duration " : 0 , " InstanceId " : " i - 2222 " , " AutoRenewEnabled " : false } ] } , " RequestId " : " 71FBB7A5 - C793 - 4A0D - B17E - D6B426EA74 6A " }
```

If automatic renewal is set, the returned attribute `AutoRenewEnabled` is true.

If automatic renewal is not set, the attribute is false.

- Enable and cancel automatic renewal for ECS instances

To enable automatic renewal for ECS instances, three input parameters are required:

- **InstanceId:** You can set automatic renewal for up to 100 ECS instances of the Subscription billing method at a time. Use commas to separate multiple instance IDs.
- **Duration:** Set to 1, 2, 3, 6, or 12, in unit of Month.
- **AutoRenew:** Set to true to enable automatic renewal. Set to false to disable automatic renewal.

```
def setting_instance_auto_renew ( instance_ids ,
auto_renew = True ):
    logging . info ( ' execute enable auto renew ' +
instance_ids )
    request = ModifyInstanceAutoRenewAttributeRequest ( );
    request . set_Duration ( 1 );
    request . set_AutoRenew ( auto_renew );
    request . set_InstanceId ( instance_ids )
    _send_request ( request )
```

When the operation is successful, the following response is returned:

```
{" RequestId ":" 7DAC9984 - AAB4 - 43EF - 8FC7 - 7D74C57BE4 6D "}
```

You can perform a query after successful renewal. The system returns the renewal duration and the status of automatic renewal (true/false).

```
{" InstanceRenewAttributes ":{" InstanceRenewAttribute ":[{"
Duration ": 1 ," InstanceId ":" i - 1111 "," AutoRenewEnabled ":
true },{" Duration ": 1 ," InstanceId ":" i - 2222 "," AutoRenewE
nabled ": true }]}," RequestId ":" 7F4D14B0 - D0D2 - 48C7 - B310 -
B1DF713D43 31 "}
```

Complete example code

```
# coding = utf - 8
# if the python sdk is not install using ' sudo pip
install aliyun - python - sdk - ecs '
# if the python sdk is install using ' sudo pip
install -- upgrade aliyun - python - sdk - ecs '
# make sure the sdk version is 2 . 1 . 2 , you can
use command ' pip show aliyun - python - sdk - ecs ' to
check
import json
import logging
from aliynsdkc ore import client
from aliynsdke cs . request . v20140526 . DescribeIn
stanceAutoRenewAttributeRequest import \
DescribeInstanceAutoRenewAttributeRequest
from aliynsdke cs . request . v20140526 . DescribeIn
stancesRequest import DescribeInstancesRequest
```

```

from aliyunsdke cs . request . v20140526 . ModifyInst
anceAutoRe newAttribu teRequest import \
    ModifyInst anceAutoRe newAttribu teRequest
from aliyunsdke cs . request . v20140526 . RenewInsta nceRequest
import RenewInsta nceRequest
logging . basicConfi g ( level = logging . INFO ,
                        format = '%(asctime) s %(filename) s [line :
%(lineno) d] %(levelname) s %(message) s ',
                        datefmt = '%a , %d %b %Y %H :%M :%S ')
clt = client . AcscClient (' Your Access Key Id ', ' Your
Access Key Secret ', ' cn - beijing ')
# data format in UTC , only support passed the value
for minute , seconds is not support .
INSTANCE_E XPIRED_STA RT_TIME_IN _UTC_STRIN G = ' 2017 - 01 -
22T00 : 00Z '
INSTANCE_E XPIRE_END_ TIME_IN_UT C_STRING = ' 2017 - 01 - 28T00 :
00Z '
def renew_job ( page_size = 100 , page_numbe r = 1 , check_need
_renew = True , security_g roup_id = None ):
    response = describe_n eed_renew_ instance ( page_size =
page_size , page_numbe r = page_numbe r ,
                                                check_need _renew =
check_need _renew ,
                                                security_g roup_id =
security_g roup_id )
    response_l ist = response . get (' Instances '). get ('
Instance ')
    logging . info ("%s instances need to renew ", str (
response . get (' TotalCount ')))
    if response_l ist > 0 :
        instance_i ds = ''
        for item in response_l ist :
            instance_i d = item . get (' InstanceId ')
            instance_i ds += instance_i d + ','
            renew_inst ance ( instance_i d = instance_i d )
        logging . info ("%s execute renew action ready ",
instance_i ds )
def describe_n eed_renew_ instance ( page_size = 100 ,
page_numbe r = 1 , instance_i d = None ,
                                                check_need _renew = True ,
security_g roup_id = None ):
    request = DescribeIn stancesReq uest ()
    if check_need _renew is True :
        request . set_Filter 3Key (" ExpiredSta rtTime ")
        request . set_Filter 3Value ( INSTANCE_E XPIRED_STA
RT_TIME_IN _UTC_STRIN G )
        request . set_Filter 4Key (" ExpiredEnd Time ")
        request . set_Filter 4Value ( INSTANCE_E XPIRE_END_
TIME_IN_UT C_STRING )
    if instance_i d is not None :
        request . set_Instan ceIds ( json . dumps ([ instance_i d
]))
    if security_g roup_id :
        request . set_Securi tyGroupId ( security_g roup_id )
        request . set_PageNu mber ( page_numbe r )
        request . set_PageSi ze ( page_size )
        return _send_requ est ( request )
# check the instances is renew or not
def describe_i nstance_au to_renew_s etting ( instance_i ds ,
expected_a uto_renew = True ):
    describe_r equest = DescribeIn stanceAuto RenewAttri
buteReques t ()
    describe_r equest . set_Instan ceId ( instance_i ds )

```

```

    response_detail = _send_request ( request = describe_instances )
    failed_instances_ids = ''
    if response_detail is not None :
        attributes = response_detail.get ( ' InstanceReservationAttributes ' ). get ( ' InstanceReservationAttributes ' )
        if attributes :
            for item in attributes :
                auto_renew_status = item . get ( ' AutoRenewEnabled ' )
                if auto_renew_status != expected_auto_renew :
                    failed_instances_ids += item . get ( ' InstanceId ' ) + ','
            if len ( failed_instances_ids ) > 0 :
                logging . error ( " instance %s auto renew not match expect %s .", failed_instances_ids ,
                    expected_auto_renew )
def setting_instance_auto_renew ( instance_ids , auto_renew = True ) :
    logging . info ( ' execute enable auto renew ' + instance_ids )
    request = ModifyInstanceAutoReservationAttributeRequest ()
    request . set_Duration ( 1 ) ;
    request . set_AutoRenew ( auto_renew ) ;
    request . set_InstanceId ( instance_ids )
    _send_request ( request )
    describe_instance_auto_renew_setting ( instance_ids , auto_renew )
# if using the instance id can be found means the instance is not renew successfully .
def check_instance_need_renew ( instance_id ) :
    response = describe_next_renew_instance ( instance_id = instance_id )
    if response is not None :
        return response . get ( ' TotalCount ' ) == 1
    return False
# Renew an instance for a month
def renew_instance ( instance_id , period = ' 1 ' ) :
    need_renew = check_instance_need_renew ( instance_id )
    if need_renew :
        _renew_instances_action ( instance_id = instance_id , period = period )
        # describe_next_renew_instance ( instance_id = instance_id , check_need_renew = False )
def _renew_instances_action ( instance_id , period = ' 1 ' ) :
    request = RenewInstanceRequest ()
    request . set_Period ( period )
    request . set_InstanceId ( instance_id )
    response = _send_request ( request )
    logging . info ( ' renew %s ready , output is %s ' , instance_id , response )
def _send_request ( request ) :
    request . set_AcceptFormat ( ' json ' )
    try :
        response_str = clt . do_action ( request )
        logging . info ( response_str )
        response_detail = json . loads ( response_str )
        return response_detail
    except Exception as e :
        logging . error ( e )
if __name__ == ' __main__ ' :
    logging . info ( " Renew ECS Instance by OpenApi !" )

```

```

# Query whether there is any instance that needs
to be renewed within the specified time range .
describe_instances_renewal_instance ()
# Renew an instance by direct fee deduction
renew_instance (' i - 1111 ')
# Query the status of automatic renewal
# describe_instances_auto_renewal_setting (' i - 1111 , i -
2222 ')
# Set automatic instance renewal
# setting_instance_auto_renewal (' i - 1111 , i - 2222 ')

```

If you want to learn other API operations in ECS, see [ECS API operation](#).

2.6 Use APIs to manage preemptible instances

This topic describes how to use Alibaba Cloud ECS SDKs to quickly create and manage preemptible instances.

Preparation

Before you begin, make sure that:

- You know which instance types and regions meet your business requirements.
- You have a basic understanding of Alibaba Cloud ECS SDKs and calling methods.

For more information, [see SDK documentation](#).

- The ECS SDK version for preemptible instances is 4.2.0 and later. Here is an example of how to change the pom dependency.

```

< dependency >
  < groupId > com . aliyun </ groupId >
  < artifactId > aliyun - java - sdk - core </ artifactId >
  < version > 3 . 2 . 8 </ version >
</ dependency >
< dependency >
  < groupId > com . aliyun </ groupId >
  < artifactId > aliyun - java - sdk - ecs </ artifactId >
  < version > 4 . 2 . 0 </ version >
</ dependency >

```

Query regions and available instance types

Use the [DescribeZones](#) interface to query the regions where you can create preemptible instances and the available instance types. The sample code is as follows:

- `OpenApiCaller.java`

```

public class OpenApiCaller {
  IClientProfile profile;
  IAcsClient client;
  public OpenApiCaller () {
    profile = DefaultProfile.getProfile (" cn - hangzhou
", AKSUtil.accessKeyId, AKSUtil.accessKeySecret );

```

```

        client = new DefaultAcsClient ( profile );
    }
    public < T extends AcsResponse > T doAction (
    AcsRequest < T > var1 ) {
        try {
            return client . getAcsResponse ( var1 );
        } catch ( Throwable e ) {
            e . printStackTrace ();
            return null ;
        }
    }
}

```

• DescribeZonesSample.java

```

public class DescribeZonesSample {
    public static void main ( String [] args ) {
        OpenApiCaller caller = new OpenApiCaller ();
        DescribeZonesRequest request = new DescribeZonesRequest ();
        request . setRegionId ( " cn - zhangjiakou " ); // You can
        use DescribeRegionsRequest to get the RegionId of
        each region
        request . setSpotStrategy ( " SpotWithPriceLimit " ); //
        This field must be entered to query the availability
        of instance types
        request . setInstanceChargeType ( " PostPaid " ); // Post -
        paid mode , preemptible instances must be post - paid
        DescribeZonesResponse response = caller . doAction (
        request );
        System . out . println ( JSON . toJSONString ( response ) );
    }
}

```

In the following output result, you can see the instance types, disk types, and network types available in each region.

```

{
  " requestId ": " 388D6321 - E587 - 470C - 8CFA - 8985E2963D AE ",
  " zones ": [
    {
      " localName ": " China North 3 Zone A ",
      " zoneId ": " cn - zhangjiakou - a ",
      " availableDiskCategories ": [
        " cloud_ssd ",
        " cloud_efficiency "
      ],
      " availableInstanceTypes ": [
        " ecs . e4 . large ",
        " ecs . n4 . 4xlarge ",
        " ecs . sn2 . medium ",
        " ecs . i1 . 2xlarge ",
        " ecs . se1 . 2xlarge ",
        " ecs . n4 . xlarge ",
        " ecs . se1ne . 2xlarge ",
        " ecs . se1 . large ",
        " ecs . sn2 . xlarge ",
        " ecs . se1ne . xlarge ",
        " ecs . xn4 . small ",
        " ecs . sn2ne . 4xlarge ",
        " ecs . se1ne . 4xlarge "
      ]
    }
  ]
}

```



```

    " ecs . sn1 . medium ",
    " ecs . n4 . 8xlarge ",
    " ecs . mn4 . large ",
    " ecs . e4 . 2xlarge ",
    " ecs . mn4 . 2xlarge ",
    " ecs . mn4 . 8xlarge ",
    " ecs . n4 . 2xlarge ",
    " ecs . e4 . xlarge ",
    " ecs . sn2ne . large ",
    " ecs . sn2ne . xlarge ",
    " ecs . sn1ne . large ",
    " ecs . n4 . large ",
    " ecs . sn1 . 3xlarge ",
    " ecs . e4 . 4xlarge ",
    " ecs . sn1ne . 2xlarge ",
    " ecs . e4 . small ",
    " ecs . i1 . 4xlarge ",
    " ecs . se1 . 4xlarge ",
    " ecs . sn2ne . 2xlarge ",
    " ecs . sn2 . 3xlarge ",
    " ecs . i1 . xlarge ",
    " ecs . n4 . small ",
    " ecs . sn1ne . 4xlarge ",
    " ecs . mn4 . 4xlarge ",
    " ecs . sn1ne . xlarge ",
    " ecs . se1ne . large ",
    " ecs . sn2 . large ",
    " ecs . i1 - c5d1 . 4xlarge ",
    " ecs . sn1 . xlarge ",
    " ecs . sn1 . large ",
    " ecs . mn4 . small ",
    " ecs . mn4 . xlarge ",
    " ecs . se1 . xlarge "
  ],
  "availableResourceCreation": [
    " VSwitch ",
    " IoOptimized ",
    " Instance ",
    " Disk "
  ],
  "availableResources": [
    {
      "dataDiskCategories": [
        " cloud_ssd ",
        " cloud_efficiency "
      ],
      "instanceGenerations": [
        " ecs - 3 ",
        " ecs - 2 "
      ],
      "instanceTypeFamilies": [
        " ecs . mn4 ",
        " ecs . sn1 ",
        " ecs . sn2 ",
        " ecs . sn1ne ",
        " ecs . xn4 ",
        " ecs . i1 ",
        " ecs . se1 ",
        " ecs . e4 ",
        " ecs . n4 ",
        " ecs . se1ne ",
        " ecs . sn2ne "
      ],
      "instanceTypes": [

```

```

        " ecs . n4 . 4xlarge ",
        " ecs . sn2 . medium ",
        " ecs . i1 . 2xlarge ",
        " ecs . se1 . 2xlarge ",
        " ecs . n4 . xlarge ",
        " ecs . se1ne . 2xlarge ",
        " ecs . se1 . large ",
        " ecs . sn2 . xlarge ",
        " ecs . se1ne . xlarge ",
        " ecs . xn4 . small ",
        " ecs . sn2ne . 4xlarge ",
        " ecs . se1ne . 4xlarge ",
        " ecs . sn1 . medium ",
        " ecs . n4 . 8xlarge ",
        " ecs . mn4 . large ",
        " ecs . mn4 . 2xlarge ",
        " ecs . mn4 . 8xlarge ",
        " ecs . n4 . 2xlarge ",
        " ecs . sn2ne . large ",
        " ecs . sn2ne . xlarge ",
        " ecs . sn1ne . large ",
        " ecs . n4 . large ",
        " ecs . sn1 . 3xlarge ",
        " ecs . sn1ne . 2xlarge ",
        " ecs . e4 . small ",
        " ecs . i1 . 4xlarge ",
        " ecs . se1 . 4xlarge ",
        " ecs . sn2ne . 2xlarge ",
        " ecs . sn2 . 3xlarge ",
        " ecs . i1 . xlarge ",
        " ecs . n4 . small ",
        " ecs . sn1ne . 4xlarge ",
        " ecs . mn4 . 4xlarge ",
        " ecs . sn1ne . xlarge ",
        " ecs . se1ne . large ",
        " ecs . sn2 . large ",
        " ecs . i1 - c5d1 . 4xlarge ",
        " ecs . sn1 . xlarge ",
        " ecs . sn1 . large ",
        " ecs . mn4 . small ",
        " ecs . mn4 . xlarge ",
        " ecs . se1 . xlarge "
    ],
    " ioOptimize d ": true ,
    " networkTyp es ": [
        " vpc "
    ],
    " systemDisk Categories ": [
        " cloud_ssd ",
        " cloud_effi ciency "
    ]
}
],
" availableV olumeCate g ories ": [
    " san_ssd ",
    " san_effici ency "
]
}
]

```

```
}

```

Query preemptible instance price history

Use the [DescribeSpotPriceHistory](#) interface to query the price changes of a preemptible instance type over the last 30 days, so you can find the most cost efficient regions and instance types. The sample code (`DescribeSpotPriceHistorySample.java`) is as follows.

```
public class DescribeSpotPriceHistorySample {
    public static void main ( String [] args ) {
        OpenApiClient caller = new OpenApiClient ();
        List< DescribeSpotPriceHistoryResponse.SpotPriceType > result = new ArrayList< DescribeSpotPriceHistoryResponse.SpotPriceType >();
        int offset = 0 ;
        while ( true ) {
            DescribeSpotPriceHistoryRequest request = new DescribeSpotPriceHistoryRequest ();
            request . setRegionId ( " cn - hangzhou " );// You can use DescribeRegionsRequest to get the RegionId of each region where preemptible instances are available
            request . setZoneId ( " cn - hangzhou - b " );// You must enter the zone
            request . setInstanceType ( " ecs . sn2 . medium " );// See the instance types returned by DescribeZones , this field is mandatory
            request . setNetworkType ( " vpc " );// See the network types returned by DescribeZones , this field is mandatory
            // request . setIoOptimized ( " optimized " );// Determines if the instance is I / O optimized , see IoOptimized returned by DescribeZones , this field is optional
            // request . setStartTime ( " 2017 - 09 - 20T08 : 45 : 08Z " );// The price start time , optional , default value : within 3 days
            // request . setEndTime ( " 2017 - 09 - 28T08 : 45 : 08Z " );// Price end time , optional
            request . setOffset ( offset );
            DescribeSpotPriceHistoryResponse response = caller . doAction ( request );
            if ( response != null && response . getSpotPrices () != null ) {
                result . addAll ( response . getSpotPrices () );
            }
            if ( response . getNextOffset () == null || response . getNextOffset () == 0 ) {
                break ;
            } else {
                offset = response . getNextOffset ();
            }
        }
        if ( ! result . isEmpty () ) {
            for ( DescribeSpotPriceHistoryResponse.SpotPriceType spotPriceType : result ) {
                System . out . println ( spotPriceType . getTimestamp () + "----> spotPrice:" + spotPriceType . getSpotPrice () + "----> originPrice:" + spotPriceType . getOriginPrice () );
            }
            System . out . println ( result . size () );
        }
    }
}
```

```

    } else {
    }
}

```

Returned results sample.

```

2017 - 09 - 26T06 : 28 : 55Z ---> spotPrice : 0 . 24 ---->
originPrice : 1 . 2
2017 - 09 - 26T14 : 00 : 00Z ---> spotPrice : 0 . 36 ---->
originPrice : 1 . 2
2017 - 09 - 26T15 : 00 : 00Z ---> spotPrice : 0 . 24 ---->
originPrice : 1 . 2
2017 - 09 - 27T14 : 00 : 00Z ---> spotPrice : 0 . 36 ---->
originPrice : 1 . 2
2017 - 09 - 27T15 : 00 : 00Z ---> spotPrice : 0 . 24 ---->
originPrice : 1 . 2
2017 - 09 - 28T14 : 00 : 00Z ---> spotPrice : 0 . 36 ---->
originPrice : 1 . 2
2017 - 09 - 28T15 : 00 : 00Z ---> spotPrice : 0 . 24 ---->
originPrice : 1 . 2
2017 - 09 - 29T06 : 28 : 55Z ---> spotPrice : 0 . 24 ---->
originPrice : 1 . 2

```

Repeat this process to find the price trends and recent prices of the instance type in each zones.



Note:

You can use average price or maximum price to determine if you can afford this preemptible instance. You can also use more rational data models to analyze historical price data and adjust your instance types and zones at will for maximum cost effectiveness.

Create a preemptible instance

You must complete the following work before creating a preemptible instance:

- To use a custom image to create a preemptible instance, you must have already [CreateImage](#).
- [Create a security group](#) in the console or use the [CreateSecurityGroup](#) to create a security group. Then, retrieve the security group ID (SecurityGroupId).
- In the console, create [a VPC and VSwitch](#), or use the [CreateVpc](#) and [CreateVSwitch](#) interfaces to do so. Then, retrieve the VSwitch ID (VSwitchId).

Use the [CreateInstance](#) to create a preemptible instance. The sample code (CreateInstaneSample.java) is as follows.

```

public class CreateInstaneSample {
    public static void main ( String [] args ) {

```

```

        OpenApiCaller caller = new OpenApiCaller ();
        CreateInstanceRequest request = new CreateInstanceRequest ();
        request . setRegionId ( " cn - hangzhou " ); // The region ID
        request . setZoneId ( " cn - hangzhou - b " ); // The zone ID
        request . setSecurityGroupId ( " sg - bp11nhf94i vkdxwb2gd4 " ); // The ID of the security group
        request . setImageId ( " centos_7_0_3_64_20G_alibase_20170818.vhd " ); // We recommend that you select a custom image you have prepared in this region
        request . setVSwitchId ( " vsw - bp164cyont hfudn9kj5br " ); // For VPC , the VSwitch ID is required
        request . setInstanceType ( " ecs . sn2 . medium " ); // Enter the instance type you want to purchase
        request . setIoOptimized ( " optimized " ); // See the parameters returned by DescribeZones
        request . setSystemDiskCategory ( " cloud_ssd " ); // See the parameters returned by DescribeZones , select one : cloud_ssd , cloud_efficiency , or cloud
        request . setSystemDiskSize ( 40 );
        request . setInstanceChargeType ( " PostPaid " ); // Post - paid is required for preemptible instances
        request . setSpotStrategy ( " SpotWithPriceLimit " ); // SpotWithPriceLimit : bid mode , SpotAsPriceGo : no bids , the maximum Pay - As - You - Go price
        request . setSpotPriceLimit ( 0 . 25F ); // This applies only to SpotWithPriceLimit . Set the maximum price you are willing to pay , units : USD / hour . An instance is created when this price is higher than the current market price
        CreateInstanceResponse response = caller . doAction ( request );
        System . out . println ( response . getInstanceId ());
    }
}

```

Recover a preemptible instance

Mandatory recovery can be imposed on preemptible instances due to changes in price or supply and demand. At such a time, the operation of the preemptible instance is suspended. Before being released, the preemptible instance enters the locked status and a prompt notifies you that the instance will be automatically recovered. You can design a withdrawal logic to automatically process instances in the recovery status.

Now, you can use the following methods to determine the suspension and lock statuses of preemptible instances:

- Obtain this information from the [instance metadata](#). Run the following command.

```
curl 'http://100.100.100.200/latest/meta-data/instance/spot/termination-time'
```

If the returned result is blank, this indicates the instance can continue to be used. If no result is returned, it indicates the instance can continue to be used. If the returned result contains UTC time stamp information in the format of YYYY-MM-DDTHH:mm:ssZ (for example `2015-01-05T18:02:00Z`), it indicates the instance will be released at the specified time.

- You can use the `OperationLocks` information returned by the [DescribeInstances](#) to see if an instance is in the `Awaiting Recovery` status. The sample code (`DescribeInstancesSample.java`) is as follows.

```
public class DescribeInstancesSample {
    public static void main (String [] args) throws
    InterruptedException {
        OpenApiCaller caller = new OpenApiCaller ();
        JSONArray allInstances = new JSONArray ();
        allInstances.addAll (Arrays.asList ("i-bp18hgfaiekoqwo0y2n", "i-bp1ecbyds24ij63w146c"));
        while (!allInstances.isEmpty ()) {
            DescribeInstancesRequest request = new
            DescribeInstancesRequest ();
            request.setRegionId ("cn-hangzhou");
            request.setInstanceIds (allInstances.toJSONString ()); // Specify the instance ID, maximum efficiency
            DescribeInstancesResponse response = caller.doAction (request);
            List<DescribeInstancesResponse.Instance>
            instanceList = response.getInstanceList ();
            if (instanceList != null && !instanceList.isEmpty ()) {
                for (DescribeInstancesResponse.Instance
            instance : instanceList) {
                    System.out.println ("result: instance:" +
            instance.getInstanceId () + ", az:" + instance.getZoneId ());
                    if (instance.getOperationLocks () != null
            ) {
                        for (DescribeInstancesResponse.Instance.LockReason
            lockReason : instance.getOperationLocks ()) {
                            System.out.println ("instance:" +
            instance.getInstanceId () + "--> lockReason:" + lockReason.getLockReason () + ", vmStatus:" + instance.getStatus ());
                            if ("Recycling".equals (lockReason.getLockReason ())) {
                                // do your action
                                System.out.println ("spot
            instance will be recycled immediately, instance id:"
            + instance.getInstanceId ());
                                allInstances.remove (instance.getInstanceId ());
                            }
                        }
                    }
                }
            }
        }
    }
}
```

```
    }
    }
    } else {
        System.out.println("try describeInstances again later...");
        Thread.sleep(2 * 60 * 1000);
    }
}
}
```

The output result when recovery is triggered is as follows.

```
instance : i - bp1ecbyds2 4ij63w146c --> lockReason : Recycling ,
vmStatus : Stopped
spot instance will be recycled immediately , instance
id : i - bp1ecbyds2 4ij63w146c
```

Other operations

You can start, stop, and release preemptible instances. These operations are the same as Pay-As-You-Go instances. For more information, see:

- Start an instance: [StartInstance](#)
- Stop an instance: [StopInstance](#)
- Release an instance: [DeleteInstance](#)

2.7 Query available resources for configuration changes

This topic describes how to query information about available resources before you change the configurations of an ECS instance.

Background information

Before you call an API to modify the configuration of an ECS instance (such as [ModifyInstanceSpec](#) or [ModifyPrepayInstanceSpec](#)), you can call [DescribeResourcesModification](#) to:

- Query the information of available resources you can use to upgrade an ECS instance.
- Query the information of available resources you can use to upgrade a system disk.

Limits

If you need to change the configurations of one or more ECS instances of different network types and different instance generations at a time, you can configure

`MigrateAcrossZone = true` in [DescribeResourcesModification](#). However, such

changes will modify the private IP addresses of ECS instances of the Classic network type. If you are using [phased-out instance types](#), you must exercise caution when you upgrade a non-I/O-optimized instance to an I/O-optimized instance. Depending on the network type or instance type family, changes to the configurations of one or more ECS instances at a time may cause the following results.

Item subject to change		Change configurations of a non-I/O-optimized Generation I instance across instance types		Change configurations of instances of other generations	
		Classic network	VPC	Classic network	VPC
Private IP address		Will change	Will not change	Will change	Will not change
Drive name (Linux only)	Basic cloud disk	Changes to xvda or xvdb		Will not change	
	Ultra disk	Changes to vda or vdb			
	cloud_efficiency				
Software license code		Will change			

Prerequisites

You have obtained the AccessKey and the region ID of the related resource. For more information, see [Regions and zones](#) and [Create an AccessKey](#).

Install the Python SDK



Note:

Make sure that your Python SDK version is v4.6.3 or later. To get the latest ECS SDK, go to [GitHub Repo Alibaba Cloud](#).

The following examples show how to install the Python SDK in the Linux (the version of Python is v2.7):

- For root users, run:

```
pip install aliyun-python-sdk-ecs
```

- For ordinary users, run:

```
sudo pip install aliyun-python-sdk-ecs
```



Note:

Make sure that your Python SDK version is v4.6.3 or later. To get the latest ECS SDK, go to [GitHub Repo Alibaba Cloud](#).

Scenario example

```
# Your AccessKeyId
ak_id = "YOU_ACCESS_KEY_ID"

# Your AccessKeySecret
ak_secret = "YOU_ACCESS_SECRET"

# ID of the region to which the resource belongs
region_id = "cn-hangzhou"

# Resource ID, such as an instance ID
resource_id = "i-xxxx"

# Target resource type. Valid values: InstanceType |
# SystemDisk
destination_instance_type = 'InstanceType'
destination_system_disk = 'SystemDisk'

# Actions for changing configurations. Valid values:
# Upgrade | Downgrade | Renew | RenewModify
operation_type = 'Upgrade'

# Target instance type. You can call DescribeInstanceTypes to get the latest instance types. When
# the DestinationResource parameter takes the value
# of SystemDisk, the InstanceType parameter must be
# specified.
instance_type = "ecs.n1.large"

# Exercise caution when configuring the MigrateAcrossZone parameter. MigrateAcrossZone (Boolean) allows
# you to change configurations across one or more
# instances. Default value: False
# If the parameter MigrateAcrossZone takes the value
# of True, after you have upgraded your ECS instance
# according to the returned information, note the
# following:
# Classic network instances:
# 1. For a Generation I ECS instance, if it is
# upgraded from a non-I/O-optimized instance to an
# I/O-optimized instance, its private IP address, drive
# name, and software license code will change. For
# Linux instances, the basic cloud disk (cloud) will be
# identified as xvda or xvdb, and the Ultra Disk (
```

```

cloud_efficiency ) and SSD Cloud disk ( cloud_ssd ) will
be identified as vda or vdb .
# 2 . For other generation s of instances , the private
IP address will change .
# VPC instances : For a Generation I ECS instance , if
it is upgraded from a non - I / O - optimized instance
to an I / O - optimized instance , its drive name
and software license code will change . For Linux
instances , the basic cloud disk ( cloud ) will be
identified as xvda or xvdb , and the Ultra Disk (
cloud_efficiency ) and SSD Cloud disk ( cloud_ssd ) will
be identified as vda or vdb .
migrate_across_zone = False

clt = client . AcscClient ( ak_id , ak_secret , region_id )

def _execute_request ( request ):
    response = _send_request ( request )
    if response is None :
        print ' response is None '
        return
    if response . get ( ' Code ' ) is None :
        availableZones = response . get ( ' AvailableZones ' ).
get ( ' AvailableZone ' )
        if availableZones is None :
            print ' availableZones is None '
            return
        for availableZone in availableZones :
            zoneId = availableZone . get ( ' ZoneId ' )
            values = []
            availableResources = availableZone . get ( '
AvailableResources ' ). get ( ' AvailableResource ' )
            if availableResources is None :
                print ' availableResources is None '
                return
            for availableResource in availableResources :
                supportedResources = availableResource . get ( '
SupportedResources ' ). get ( ' SupportedResource ' )
                if supportedResources is None :
                    print ' supportedResource is None '
                    return
                for supportedResource in supportedResources
:
                    status = supportedResource . get ( ' Status ' )
                    if status == " Available ":
                        value = supportedResource . get ( ' Value
' )
                        values . append ( value )
            print " ecs in zone %s resource value list
is %s "%( zoneId , values )

def build_request ():
    request = DescribeResourcesModificationRequest ()
    request . set_ResourceId ( resource_id )
    request . set_MigrateAcrossZone ( migrate_across_zone )
    request . set_OperationType ( operation_type )
    return request

# Send API request
def _send_request ( request ):
    request . set_AcceptFormat ( ' json ' )
    try :
        response_str = clt . do_action ( request )
        logging . info ( response_str )

```

```

        response_detail = json.loads(response_string)
        return response_detail
    except Exception as e:
        logging.error(e)
        return response_detail
    except Exception as e:
        logging.error(e)

```

Scenario 1: Query available resources for upgrading an instance type

```

# Query available resources for upgrading an instance type
def describe_resource_instance_type():
    request = build_request()
    request.set_DestinationResource(destination_instance_type)
    _execute_request(request)

```

Scenario 2: Query available resources for upgrading a system disk

```

# Query available resources for upgrading a system disk
def describe_resource_system_disk():
    request = build_request()
    request.set_DestinationResource(destination_system_disk)
    request.set_InstanceType(instance_type)
    _execute_request(request)

```

Complete code

```

# coding = utf - 8

# if the python sdk is not install , run 'sudo pip
install aliyun - python - sdk - ecs '
# if the python sdk is install , run 'sudo pip
install -- upgrade aliyun - python - sdk - ecs '
# make sure the sdk version is 4 . 6 . 3 . To check
your version , run 'pip show aliyun - python - sdk - ecs ' '
pip show aliyun - python - sdk - ecs ' to check

import json
import logging

from aliyunsdkcore import client
from aliyunsdkecs.request.v20140526.DescribeResourcesModificationRequest import DescribeResourcesModificationRequest

logging.basicConfig(level=logging.INFO,
                    format='%(asctime)s %(filename)s [line:
%(lineno)d] %(levelname)s %(message)s ',
                    datefmt='%a, %d %b %Y %H:%M:%S')

# Your AccessKeyId
ak_id = "YOU_ACCESS_KEY_ID"

# Your AccessKeySecret
ak_secret = "YOU_ACCESS_SECRET"

# ID of the region to which the resource belongs

```

```

region_id = " cn - hangzhou "

# Resource ID , such as an instance ID
resource_id = " i - xxxx "

# Target resource type . Valid values : InstanceType |
SystemDisk
destination_instance_type = ' InstanceType '
destination_system_disk = ' SystemDisk '

# Actions for changing configurations . Valid values :
Upgrade | Downgrade | RenewDowngrade | RenewModify
operation_type = ' Upgrade '

# Target instance type . You can call DescribeInstanceTypes
to get the latest instance types . When the DestinationResource
parameter takes the value of SystemDisk , the InstanceType
parameter must be specified .
instance_type = " ecs . n1 . large "

# Exercise caution when configuring the MigrateAcrossZone
parameter . MigrateAcrossZone ( Boolean ) allows you to change
configurations across one or more instances . Default value : False
# If the parameter MigrateAcrossZone takes the value of True ,
after you have upgraded your ECS instance according to the
returned information , note the following :
# Classic network instances :
# 1 . For a Generation I ECS instance , if it is upgraded from a
non - I / O - optimized instance to an I / O - optimized instance ,
its private IP address , drive name , and software license code
will change . For Linux instances , the basic cloud disk ( cloud )
will be identified as xvda or xvdb , and the Ultra Disk ( cloud_efficiency )
and SSD Cloud disk ( cloud_ssd ) will be identified as vda or vdb .
# 2 . For other generations of instances , the private IP address
will change .
# VPC instances : For a Generation I ECS instance , if it is
upgraded from a non - I / O - optimized instance to an I / O -
optimized instance , its drive name and software license code
will change . For Linux instances , the basic cloud disk ( cloud )
will be identified as xvda or xvdb , and the Ultra Disk ( cloud_efficiency )
and SSD Cloud disk ( cloud_ssd ) will be identified as vda or vdb .
migrate_across_zone = False

clt = client . AcsClient ( ak_id , ak_secret , region_id )

# Query available resources for upgrading an instance type
def describe_resource_instance_type ():
    request = build_request ()
    request . set_DestinationResource ( destination_instance_type )
    _execute_request ( request )

# Query available resources for upgrading a system disk
def describe_resource_system_disk ():
    request = build_request ()

```

```

    request . set_DestinationResource ( destination_system_d
isk )
    request . set_InstanceType ( instance_type )
    _execute_request ( request )

def _execute_request ( request ):
    response = _send_request ( request )
    if response is None :
        print ' response is None '
        return
    if response . get ( ' Code ' ) is None :
        availableZones = response . get ( ' AvailableZones ' ).
get ( ' AvailableZone ' )
        if availableZones is None :
            print ' availableZones is None '
            return
        for availableZone in availableZones :
            zoneId = availableZone . get ( ' ZoneId ' )
            values = []
            availableResources = availableZone . get ( '
AvailableResources ' ). get ( ' AvailableResource ' )
            if availableResources is None :
                print ' availableResources is None '
                return
            for availableResource in availableResources :
                supportedResources = availableResource . get ( '
SupportedResources ' ). get ( ' SupportedResource ' )
                if supportedResources is None :
                    print ' supportedResource is None '
                    return
                for supportedResource in supportedResources
:
                    status = supportedResource . get ( ' Status ' )
                    if status == " Available ":
                        value = supportedResource . get ( ' Value
' )
                        values . append ( value )
            print " ecs in zone %s resource value list
is %s "%( zoneId , values )

def build_request ():
    request = DescribeResourcesModificationRequest ()
    request . set_ResourceId ( resource_id )
    request . set_MigrateAcrossZone ( migrate_across_zone )
    request . set_OperationType ( operation_type )
    return request

# Send the API request
def _send_request ( request ):
    request . set_AcceptFormat ( ' json ' )
    try :
        response_str = clt . do_action ( request )
        logging . info ( response_str )
        response_detail = json . loads ( response_str )
        return response_detail
    except Exception as e :
        logging . error ( e )

if __name__ == ' __main__ ':
    print " hello ecs describe resources "
    # describe_resource_in_stance_type ()
    # describe_resource_system_disk ()
    # describe_resource_in_stance_type ()

```

```
# describe_resource_system_disk ()
```