

# 阿里云

# 阿里云SDK开发指南

## .NET SDK

文档版本：20181025

# 法律声明

---

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。
2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。
3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。
5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

## 通用约定

| 格式                                                                                | 说明                                | 样例                                                                                                                       |
|-----------------------------------------------------------------------------------|-----------------------------------|--------------------------------------------------------------------------------------------------------------------------|
|  | 该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。  |  禁止：<br>重置操作将丢失用户配置数据。                  |
|  | 该类警示信息可能导致系统重大变更甚至故障，或者导致人身伤害等结果。 |  警告：<br>重启操作将导致业务中断，恢复业务所需时间约10分钟。      |
|  | 用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。      |  说明：<br>您也可以通过按 <b>Ctrl + A</b> 选中全部文件。 |
| >                                                                                 | 多级菜单递进。                           | 设置 > 网络 > 设置网络类型                                                                                                         |
| 粗体                                                                                | 表示按键、菜单、页面名称等UI元素。                | 单击 确定。                                                                                                                   |
| <code>courier</code><br>字体                                                        | 命令。                               | 执行 <code>cd /d C:/windows</code> 命令，进入Windows系统文件夹。                                                                      |
| 斜体                                                                                | 表示参数、变量。                          | <code>bae log list --instanceid Instance_ID</code>                                                                       |
| [ ]或者[a b]                                                                        | 表示可选项，至多选择一个。                     | <code>ipconfig [-all -t]</code>                                                                                          |
| { }或者{a b}                                                                        | 表示必选项，至多选择一个。                     | <code>swich {stand   slave}</code>                                                                                       |

# 目录

---

|                                 |           |
|---------------------------------|-----------|
| 法律声明.....                       | I         |
| 通用约定.....                       | I         |
| <b>1 快速开始.....</b>              | <b>1</b>  |
| <b>2 使用手册.....</b>              | <b>3</b>  |
| 2.1 使用.NET SDK.....             | 3         |
| 2.2 错误处理.....                   | 5         |
| 2.3 配置STS Token.....            | 6         |
| 2.4 配置RamRole实现ECS实例的无AK访问..... | 8         |
| 2.5 使用CommonRequest进行调用.....    | 10        |
| <b>3 调用示例.....</b>              | <b>13</b> |
| 3.1 云服务器ECS.....                | 13        |
| 3.2 云数据库RDS.....                | 14        |
| 3.3 负载均衡SLB.....                | 15        |
| 3.4 专有网络VPC.....                | 16        |
| 3.5 弹性收缩.....                   | 17        |
| 3.6 内容分发网络CDN.....              | 18        |
| <b>4 常见错误.....</b>              | <b>20</b> |

# 1 快速开始

欢迎使用阿里云开发者工具套件（SDK）。阿里云.NET SDK让您不用复杂编程即可访问云服务器、云数据库RDS、云监控等多个阿里云服务。本操作介绍如何获取阿里云.NET SDK并开始调用。

如果您在使用SDK的过程中遇到任何问题，欢迎前往阿里云SDK[问答社区](#)提问，提问前请阅读提问引导[提交反馈](#)。

## 环境准备

- 使用阿里云.NET SDK，您需要一个RAM账号以及一对AccessKey ID和AccessKey Secret。请在阿里云控制台中的[AccessKey管理页面](#)上创建和查看您的AccessKey，或联系您的系统管理员。
- 使用阿里云SDK调用某个产品的API前，确保您已经在阿里云控制台开通了该产品。
- 阿里云.NET SDK适用于.NET Framework 4.0及以上版本。

## 安装阿里云 .NET SDK

要调用一个产品的SDK，您需要安装SDK核心库和该产品的SDK。例如，对于云服务器 SDK的调用，您需要获取和安装SDK核心库和云服务器的SDK。您可以通过以下两种方式安装.NET SDK：

- 添加DLL引用
  1. 从[.NET SDK 发布列表](#)下载包含DLL文件的压缩包，解压获得DLL文件。
  2. 在Visual Studio的解决方案资源管理器中右键单击您的项目，然后单击引用。
  3. 在弹出的菜单中单击添加引用。
  4. 在弹出对话框中，单击浏览，选择之前下载的DLL文件，单击确定。
- 项目引入方式
  1. 执行以下命令，在GitHub中下载SDK的源代码。

```
git clone https://github.com/aliyun/aliyun-openapi-net-sdk.git
```

您可以在下载生成的文件夹中找到名为 *aliyun-net-openapi-XXX* 的子文件夹，再找到 *\\\*.csproj* 文件，即 .NET 项目文件。例如，在 *aliyun-net-openapi-ecs* 子文件夹下，有一个 *aliyun-net-sdk-ecs.csproj* 文件。

2. 在Visual Studio的界面中，右键单击您的解决方案。
3. 单击添加 > 现有项目。

4. 在弹出的对话框中，选择刚才下载的源代码中相应的.NET 项目文件（例如 `aliyun-net-sdk-ecs.csproj`），单击打开。
5. 右键单击您的项目，单击引用 > 添加引用。

## 使用.NET SDK

以下这个代码示例展示了调用阿里云.NET SDK的3个主要步骤：

1. 创建并初始化DefaultAcsClient实例。
2. 创建API请求并设置参数。
3. 发起请求并处理应答或异常。

```
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Core.Exceptions;
using Aliyun.Acs.Ecs.Model.V20140526;

class TestProgram
{
    static void Main(string[] args)
    {
        // 构建一个 Client，用于发起请求
        IClientProfile profile = DefaultProfile.GetProfile(
            "<your-region-id>",
            "<your-access-key-id>",
            "<your-access-key-secret>");
        DefaultAcsClient client = new DefaultAcsClient(profile);
        try
        {
            // 构造请求
            DescribeInstancesRequest request = new DescribeInstancesRequest();
            request.PageSize = 10;
            // 发起请求，并得到 Response
            DescribeInstancesResponse response = client.GetAcsResponse(request);
            System.Console.WriteLine(response.TotalCount);
        }
        catch (ServerException ex)
        {
            System.Console.WriteLine(ex.ToString());
        }
        catch (ClientException ex)
        {
            System.Console.WriteLine(ex.ToString());
        }
    }
}
```

## 2 使用手册

### 2.1 使用.NET SDK

本文档介绍如何安装及调用阿里云.NET SDK。

#### 安装

您可以通过以下两种方式安装.NET SDK：

- 添加DLL引用
  1. 从[.NET SDK 发布列表](#)下载包含DLL文件的压缩包，解压获得DLL文件。
  2. 在Visual Studio的解决方案资源管理器中右键单击您的项目，然后单击引用。
  3. 在弹出的菜单中单击添加引用。
  4. 在弹出对话框中，单击浏览，选择之前下载的DLL文件，单击确定。
- 项目引入方式
  1. 执行以下命令，在GitHub中下载SDK的源代码。

```
git clone https://github.com/aliyun/aliyun-openapi-net-sdk.git
```

您可以在下载生成的文件夹中找到名为 *aliyun-net-openapi-XXX* 的子文件夹，再找到 *\\\*.csproj* 文件，即 .NET 项目文件。例如，在 *aliyun-net-openapi-ecs* 子文件夹下，有一个 *aliyun-net-sdk-ecs.csproj* 文件。

2. 在Visual Studio的界面中，右键单击您的解决方案。
3. 单击添加 > 现有项目。
4. 在弹出的对话框中，选择刚才下载的源代码中相应的.NET 项目文件（例如 *aliyun-net-sdk-ecs.csproj*），单击打开。
5. 右键单击您的项目，单击引用 > 添加引用。

#### 设置身份验证凭据

当使用阿里云SDK访问阿里云服务时，您需要提供阿里云账号进行身份验证。

目前，.NET SDK支持以下几种身份验证方式：

| 验证方式      | 说明                                |
|-----------|-----------------------------------|
| AccessKey | 使用AccessKey ID和AccessKey Secret访问 |

| 验证方式       | 说明                        |
|------------|---------------------------|
| StsToken   | 使用STS Token访问             |
| RamRoleArn | 使用RAM子账号的AssumeRole方式访问   |
| EcsRamRole | 在ECS实例上通过EcsRamRole实现免密验证 |

本文以AccessKey为例说明如何设置身份凭证。为了保证您的账号安全，建议您使用RAM账号来访问阿里云服务。阿里云账号的AccessKey对拥有的资源有完全的权限。RAM账号由阿里云账号授权创建，仅有对特定资源限定的操作权限。参考[创建AccessKey](#)创建RAM账号的AccessKey。

使用AccessKey作为访问凭据，需要在初始化Client时设置凭证。



说明：

确保包含AccessKey的代码不会泄漏（例如提交到外部公开的GitHub项目），否则将会危害您的阿里云账号的信息安全。

```
IClientProfile clientProfile = DefaultProfile.GetProfile(
    "<your-region-id>",           // 地域ID
    "<your-access-key-id>",       // RAM账号的AccessKey ID
    "<your-access-key-secret>"); // RAM账号AccessKey Secret
```

## 发起调用

本文以ECS为例，介绍如何使用阿里云.NET SDK发起请求。

### 1. 新建一个默认客户端实例。

```
DefaultAcsClient client = new DefaultAcsClient(clientProfile);
```

### 2. 创建请求。

请求类的命名规范为\${apiName}Request，其中\${apiName}为API名称，例如DescribeInstances。

在引入多个产品SDK时，有可能存在Request类同名的情况，请注意按照package区分。

```
DescribeInstancesRequest request = new DescribeInstancesRequest();
```

```
request.PageSize = 10;
```

### 3. 发起调用并处理应答。

```
DescribeInstancesResponse response = client.GetAcsResponse(request);
```

正常情况下，应答中的所有字段，都会被反序列化到response中，您可以直接调用response.getXXX()来获得应答中的字段。

```
System.Console.WriteLine(response.TotalCount);
```

## 2.2 错误处理

在使用阿里云.NET SDK时，当服务端或者SDK端出错时，SDK会返回相应的异常信息。这些异常信息中会包含具体的错误信息包括错误码（Error Code）和错误消息（Error Message）。

一般情况下，您不需要对阿里云.NET SDK返回的错误进行处理。您只需要根据服务端的错误提示处理服务端错误即可。

- ServerException是服务端错误信息。
- ClientException是SDK端错误信息。

例如，当出现如下错误时，您可以根据具体的错误信息，修改AccessKey的ID即可。

```
InvalidAccessKeyId.NotFound : Specified access key is not found.
```

如果您需要对SDK的错误进行处理，可以参考如下示例编写代码：

```
try
{
    CreateInstanceResponse response = client.GetAcsResponse(request);
    // 处理Response
    // ...
}
catch (ServerException e)
{
    // 这里可以添加您自己的错误处理逻辑
    // 例如，打印具体的错误信息
    Console.WriteLine("ErrorCode=" + e.ErrorCode);
    Console.WriteLine("ErrorMessage=" + e.ErrorMessage);
    // 如果遇上棘手问题，可以提交工单，并将RequestId提供给我们
    Console.WriteLine("ErrorCode=" + e.RequestId);
}
catch (ClientException e)
{
    // 这里可以添加您自己的错误处理逻辑
    // 例如，打印具体的错误信息
    Console.WriteLine("ErrorCode=" + e.ErrorCode);
    Console.WriteLine("ErrorMessage=" + e.ErrorMessage);
}
```

```
}
```

## 2.3 配置STS Token

直接使用阿里云账号的主账号的AccessKey ID和AccessKey Secret进行应用开发会有有一定的安全风险，为了提升安全性，除了通过RAM角色控制权限范围外，您还可以使用为RAM角色签发的STS Token来访问阿里云服务。

### STS Token介绍

阿里云STS (Security Token Service) 是为阿里云账号（或RAM用户）提供短期访问权限管理的云服务。通过STS，您可以为联盟用户（您的本地账号系统所管理的用户）颁发一个自定义时效和访问权限的访问凭证。联盟用户可以使用STS短期访问凭证直接调用阿里云服务API，或登录阿里云管理控制台操作被授权访问的资源。

使用STS Token调用SDK有以下优势：

- 减少了主账号AccessKey ID和AccessKey Secret泄露的风险，特别是移动设备等场景。
- 能使用灵活的权限控制，STS Token有一定的时间限制，并且根据RAM角色的灵活设置对ECS、SLB等资源的精细授权。



说明：

在使用STS Token前，确保该产品支持STS Token验证。

### 设置STS Token

您可以通过以下两种方式设置STS Token：

- 方式一：直接使用STS Token

直接使用STS Token时，您需要自行维护STS Token的周期性更新。

```
using System;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Auth;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Core.Exceptions;
using Aliyun.Acs.Ecs.Model.V20140526;

class SimpleSTSTokenSample
{
    static void Main(string[] args)
    {
        BasicSessionCredentials credentials = new BasicSessionCredentials(
            "<your-access-key-id>",
            "<your-access-key-secret>",
```

```
        "<your-session-token>");

        // 创建客户端实例
        IClientProfile profile = DefaultProfile.GetProfile("<your-
region-id>");
        DefaultAcsClient client = new DefaultAcsClient(profile,
credentials);
        try
        {
            // 创建API请求并设置参数
            DescribeInstancesRequest request = new DescribeIn
stancesRequest();
            request.PageSize = 10;

            // 请求并打印处理结果
            DescribeInstancesResponse response = client.GetAcsResp
onse(request);
            Console.WriteLine("TotalCount: {0}", response.TotalCount
);
        }
        catch (ServerException e)
        {
            Console.WriteLine(e.ErrorCode);
            Console.WriteLine(e.ErrorMessage);
        }
        catch (ClientException e)
        {
            Console.WriteLine(e.ErrorCode);
            Console.WriteLine(e.ErrorMessage);
        }
    }
}
```

其中：

- **region-id**是您正在使用的地域ID。你可以通过调用**DescribeRegions**接口查看地域ID。
- **sts-access-key-id**、**sts-access-key-secret**和**sts-session-token**是调用**AssumeRole**接口返回的授权信息。
- 方式二：使用SDK自动管理STS Token

您可以通过指定RAM的角色信息，让SDK帮您自动申请并维护STS Token。

```
using System;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Auth;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Core.Exceptions;
using Aliyun.Acs.Ecs.Model.V20140526;

class UseRoleArnSample
{
    static void Main(string[] args)
    {
        IClientProfile profile = DefaultProfile.GetProfile("<your-
region-id>");
        BasicCredentials basicCredentials = new BasicCredentials(
            "<your-access-key-id>",
```

```
        "<your-access-key-secret>");
        STSAssumeRoleSessionCredentialsProvider provider = new
        STSAssumeRoleSessionCredentialsProvider(
            basicCredentials,
            "<your-role-arn>",
            profile);

        // 创建客户端实例
        DefaultAcsClient client = new DefaultAcsClient(profile,
        provider);

        try
        {
            // 创建API请求并设置参数
            DescribeInstancesRequest request = new DescribeIn
            stancesRequest();
            request.PageSize = 10;

            // 请求并打印处理结果
            DescribeInstancesResponse response = client.GetAcsResp
            onse(request);
            Console.WriteLine("TotalCount: {0}", response.TotalCount
            );
        }
        catch (ServerException e)
        {
            Console.WriteLine(e.ErrorCode);
            Console.WriteLine(e.ErrorMessage);
        }
        catch (ClientException e)
        {
            Console.WriteLine(e.ErrorCode);
            Console.WriteLine(e.ErrorMessage);
        }
    }
}
```

其中：

- **role-arn**是角色全局资源描述符，您可以通过访问[RAM控制台](#)，单击角色名，进入详情页后查询角色名对应的role-arn。
- **role-session-name**是临时角色名称，您可以通过调用**AssumeRole**接口来获取一个扮演该角色的临时身份，创建成功后，便可以使用创建时的**RoleSessionName**参数作为本方式的**role-session-name**参数。

## 2.4 配置RamRole实现ECS实例的无AK访问

为了提高应用部署的安全性的同时提升便利性，阿里云SDK支持通过[实例元数据](#)服务来获取ECS RAM角色的授权信息来访问阿里云资源和服务。使用这种方式，您部署在ECS上的应用程序，无需在SDK上配置授权信息即可访问阿里云API（即不需要配置AccessKey）。通过这种方式授权的SDK，可以拥有这个ECS RAM角色的权限。



说明：

确保ECS实例已经配置了RAM角色。

#### 示例

```
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Auth;
using Aliyun.Acs.Core.Exceptions;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Ecs.Model.V20140526;

class NoAKAccessSample
{
    static void Main(string[] args)
    {
        // 构建一个客户端实例，用于发起请求
        DefaultProfile profile = DefaultProfile.GetProfile("<your-region-id>");
        InstanceProfileCredentialsProvider provider = new InstanceProfileCredentialsProvider(
            "<your-role-name>");
        DefaultAcsClient client = new DefaultAcsClient(profile, provider);

        try
        {
            // 构造请求
            DescribeInstancesRequest request = new DescribeInstancesRequest();
            request.PageSize = 10;
            // 发起请求，并得到 Response
            DescribeInstancesResponse response = client.GetAcsResponse(request);
            System.Console.WriteLine(response.TotalCount);
        }
        catch (ServerException ex)
        {
            System.Console.WriteLine(ex.ToString());
        }
        catch (ClientException ex)
        {
            System.Console.WriteLine(ex.ToString());
        }
    }
}
```

其中：

- **role-name**是与ECS实例关联的RAM角色名称。
- **region-id**是使用的地域的ID，你可以通过调用**DescribeRegions**接口查看地域ID。



说明：

示例中的`region-id`是目标服务(RAM角色有访问权限)的API所在地域，不一定等于这个ECS实例的地域。

## 2.5 使用CommonRequest进行调用

当您要调用的某个产品的API没有提供SDK时，可以采用泛用型的API调用方式 ( CommonRequest ) 。使用CommonRequest调用方式可实现任意Open API接口的调用。

### CommonRequest调用特点

CommonRequest的特点如下：

1. 轻量：只需Core包即可发起调用，无需下载安装各产品的SDK。
2. 简便：无需更新SDK即可调用最新发布的API。
3. 快速迭代。

### 使用CommonRequest

阿里云产品的API有RPC和RESTful两种风格，不同风格的API的CommonRequest的调用方法也不同。

通常API参数中包含**Action**参数的是RPC风格，包含**PathPattern**参数的是RESTful风格。一般情况下，每个产品内，所有API的调用风格是统一的。每个API仅支持特定的一种风格调用，传入错误的标识，可能会调用到其他API，或收到**ApiNotFound**的错误信息。

发起一次CommonRequest请求，您需要获取以下几个参数的值。您可以在[文档中心](#)各产品的API文档中获取以下参数的值。此外，部分产品也可以通过[OpenAPI Explorer](#)来获取API的参数信息。

- 域名(domain)：该产品的服务地址。
- API版本(version)：该API的版本号，格式为YYYY-MM-DD。

您可以在各产品的API文档的公共参数部分获取API版本。

- 接口信息：要调用的接口名称。

— 当调用的API为RPC风格时 ( 大部分阿里云产品API为RPC风格 ) 如ECS和RDS，需要获取**Action**参数，使用`request.ApiName = "<Action>"`的方式来指定API名称。

例如**RunInstances**接口，在发起CommonRequest请求时，需要使用`request.ApiName = "RunInstances"`来指定API名称。

— 当调用的API为RESTful风格时如容器服务，需要获取**PathPattern**参数，使用`request.PathPattern = "<PathPattern>"`的方式来指定RESTful路径。

例如容器服务的查看所有集群实例的API的**PathPattern**为/clusters，在发起CommonRequest请求时，要使用request.PathPattern = "/clusters"指定RESTful路径。

#### 示例：调用RPC风格的API

以下代码展示了如何使用CommonRequest的方式调用ECS的DescribeInstanceStatus接口：

```
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Exceptions;
using Aliyun.Acs.Core.Profile;

class Sample
{
    static void Main(string[] args)
    {
        // 构建一个客户端实例，用于发起请求
        IClientProfile profile = DefaultProfile.GetProfile(
            "<your-region-id>",
            "<your-access-key-id>",
            "<your-access-key-secret>");
        DefaultAcsClient client = new DefaultAcsClient(profile);

        try
        {
            // 构造请求
            CommonRequest request = new CommonRequest();
            request.Domain = "ecs.aliyuncs.com";
            request.Version = "2014-05-26";

            // 因为是 RPC 风格接口，需指定 ApiName(Action)
            request.Action = "DescribeInstanceStatus";
            request.AddQueryParameters("PageNumber", "1");
            request.AddQueryParameters("PageSize", "30");

            // 发起请求，并得到 Response
            CommonResponse response = client.GetCommonResponse(request);

            System.Console.WriteLine(response.Data);
        }
        catch (ServerException ex)
        {
            System.Console.WriteLine(ex.ToString());
        }
        catch (ClientException ex)
        {
            System.Console.WriteLine(ex.ToString());
        }
    }
}
```

#### 示例：调用RESTful风格的API

以下代码展示了如何使用CommonRequest的方式调用容器服务的查看所有集群实例接口：

```
using Aliyun.Acs.Core;
```

```
using Aliyun.Acs.Core.Exceptions;
using Aliyun.Acs.Core.Profile;

class Sample
{
    static void Main(string[] args)
    {
        // 构建一个客户端实例，用于发起请求
        IClientProfile profile = DefaultProfile.GetProfile(
            "<your-region-id>",
            "<your-access-key-id>",
            "<your-access-key-secret>");
        DefaultAcsClient client = new DefaultAcsClient(profile);

        try
        {
            // 构造请求
            CommonRequest request = new CommonRequest();
            request.Domain = "cs.aliyuncs.com";
            request.Version = "2015-12-15";

            // 因为是Restful风格接口，需指定 UriPattern
            request.UriPattern = "/clusters";

            // 发起请求，并得到 Response
            CommonResponse response = client.GetCommonResponse(request);

            System.Console.WriteLine(response.Data);
        }
        catch (ServerException ex)
        {
            System.Console.WriteLine(ex.ToString());
        }
        catch (ClientException ex)
        {
            System.Console.WriteLine(ex.ToString());
        }
    }
}
```

## 3 调用示例

### 3.1 云服务器ECS

本示例介绍如何使用阿里云.NET SDK调用ECS的**CreateInstance**接口创建一个ECS实例。

云服务器 ( Elastic Compute Service , 简称ECS ) 是阿里云提供了一种基础云计算服务。使用云服务器ECS就像使用水、电、煤气等资源一样便捷、高效。您无需提前采购硬件设备，而是根据业务需要，随时创建所需数量的云服务器实例。如果不再需要云服务器，也可以方便的释放资源，节省费用。

在创建ECS实例前，您需要获取以下信息：

- 镜像 ID

调用**DescribeImages**接口查看要使用的镜像 ID。

- 实例规格

查看[实例规格族](#)选择要创建的ECS实例的规格。

示例代码



说明：

运行该示例代码将创建ECS实例，并产生实际费用。

```
using System;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Core.Exceptions;
using Aliyun.Acs.Ecs.Model.V20140526;

class Sample
{
    static void Main(string[] args)
    {
        // 创建客户端实例
        IClientProfile clientProfile = DefaultProfile.GetProfile("<
your-region-id>", "<your-access-key-id>", "<your-access-key-secret>");
        DefaultAcsClient client = new DefaultAcsClient(clientProfile);
        try
        {
            // 创建API请求并设置参数
            CreateInstanceRequest request = new CreateInstanceRequest
            ();
            request.ImageId = "_32_23c472_20120822172155_aliguest.vhd
            ";
            request.InstanceType = "ecs.t1.small";
```

```
        // 请求并打印处理结果
        CreateInstanceResponse response = client.GetAcsResponse(
request);
        Console.WriteLine("InstanceId: {0}", response.InstanceId);
    }
    catch (ServerException e)
    {
        Console.WriteLine(e.ErrorCode);
        Console.WriteLine(e.ErrorMessage);
    }
    catch (ClientException e)
    {
        Console.WriteLine(e.ErrorCode);
        Console.WriteLine(e.ErrorMessage);
    }
}
}
```

## 3.2 云数据库RDS

本示例介绍如何使用阿里云.NET SDK调用RDS的**CreateDBInstance**接口创建一个RDS实例。

阿里云关系型数据库 ( Relational Database Service, 简称RDS ) 是一种稳定可靠、可弹性伸缩的在线数据库服务。基于阿里云分布式文件系统和高性能存储, RDS 支持MySQL、SQL Server、PostgreSQL和PPAS ( Postgre Plus Advanced Server, 一种高度兼容 Oracle 的数据库 ) 引擎, 并且提供了容灾、备份、恢复、监控和迁移等方面的全套解决方案, 彻底解决数据库运维的烦恼。

### 示例代码



说明：

运行该示例代码将创建RDS实例, 并产生实际费用。

```
import C#.util.UUID;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.rds.model.v20140815.CreateDBInstanceRequest;
import com.aliyuncs.rds.model.v20140815.CreateDBInstanceResponse;
public class Demo {
    public static void main(String[] args) {

        // 创建DefaultAcsClient实例并初始化
        DefaultProfile profile = DefaultProfile.getProfile(
            "<your-region-id>", // 您的可用区ID
            "<your-access-key-id>", // 您的AccessKey ID
            "<your-access-key-secret>"); // 您的AccessKey Secret
        IAcsClient client = new DefaultAcsClient(profile);
        // 创建API请求并设置参数

        CreateDBInstanceRequest request = new CreateDBInstanceRequest
        ();
```

```
request.setEngine("MySQL");
request.setEngineVersion("5.7");
request.setDBInstanceClass("mysql.n1.micro.1");
request.setDBInstanceStorage(20);
request.setDBInstanceNetType("Intranet");
request.setSecurityIPList("0.0.0.0/0");
request.setPayType("Postpaid");
request.setDBInstanceDescription("MyRds");
request.setClientToken(UUID.randomUUID().toString());

// 发起请求并处理应答或异常
CreateDBInstanceResponse response;
try {
    response = client.getAcsResponse(request);
    String dbInstanceId = response.getDBInstanceId();
    System.out.println("Create dbInstance success, instanceId"
= " + dbInstanceId);
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
```

### 3.3 负载均衡SLB

本示例介绍如何使用阿里云.NET SDK调用SLB的**CreateLoadBalancer**接口创建一个SLB实例。

负载均衡（Server Load Balancer）是将访问流量根据转发策略分发到后端多台云服务器（ECS实例）的流量分发控制服务。负载均衡扩展了应用的服务能力，增强了应用的可用性。

示例代码



说明：

运行该示例代码将创建SLB实例，并产生实际费用。

```
using System;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Core.Exceptions;
using Aliyun.Acs.Slb.Model.V20140515;

class Sample
{
    static void Main(string[] args)
    {
        // 创建客户端实例
        IClientProfile clientProfile = DefaultProfile.GetProfile("<
your-region-id>", "<your-access-key-id>", "<your-access-key-secret>");
        DefaultAcsClient client = new DefaultAcsClient(clientProfile);

        try
        {
            // 创建API请求并设置参数
```

```
        CreateLoadBalancerRequest request = new CreateLoadBalancerRequest();
        request.LoadBalancerName = "my-sample-slb";
        request.AddressType = "internet";
        request.InternetChargeType = "paybytraffic";

        // 请求并打印处理结果
        CreateLoadBalancerResponse response = client.GetAcsResponse(request);
        Console.WriteLine("LoadBalancerId: {0}", response.LoadBalancerId);
    }
    catch (ServerException e)
    {
        Console.WriteLine(e.ErrorCode);
        Console.WriteLine(e.ErrorMessage);
    }
    catch (ClientException e)
    {
        Console.WriteLine(e.ErrorCode);
        Console.WriteLine(e.ErrorMessage);
    }
}
```

### 3.4 专有网络VPC

本示例介绍如何使用阿里云.NET SDK调用VPC的**CreateVpc**接口创建一个VPC。

专有网络VPC ( Virtual Private Cloud ) 是用户基于阿里云创建的自定义私有网络, 不同的专有网络之间二层逻辑隔离, 用户可以在自己创建的专有网络内创建和管理云产品实例, 比如ECS、负载均衡、RDS等。

示例代码

```
using System;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Core.Exceptions;
using Aliyun.Acs.Vpc.Model.V20160428;

class Sample
{
    static void Main(string[] args)
    {
        // 创建客户端实例
        IClientProfile clientProfile = DefaultProfile.GetProfile("<your-region-id>", "<your-access-key-id>", "<your-access-key-secret>");
        DefaultAcsClient client = new DefaultAcsClient(clientProfile);
        try
        {
            // 创建并发起CreateVpcRequest请求
            CreateVpcRequest createRequest = new CreateVpcRequest();
            CreateVpcResponse createResponse = client.GetAcsResponse(createRequest);

            // 创建并发起DescribeVpcAttributeRequest请求
```

```
DescribeVpcAttributeRequest describeRequest = new
DescribeVpcAttributeRequest();
describeRequest.VpcId = createResponse.VpcId;
DescribeVpcAttributeResponse describeResponse = client.
GetAcsResponse(describeRequest);
Console.WriteLine("vpcId: {0}", describeResponse.VpcId);
}
catch (ServerException e)
{
    Console.WriteLine(e.ErrorCode);
    Console.WriteLine(e.ErrorMessage);
}
catch (ClientException e)
{
    Console.WriteLine(e.ErrorCode);
    Console.WriteLine(e.ErrorMessage);
}
}
```

## 3.5 弹性收缩

本示例介绍如何使用阿里云.NET SDK调用弹性收缩的**CreateScalingGroup**接口创建一个弹性伸缩组和如何通过**IoOptimized**参数指定 I/O 优化实例。

弹性伸缩是根据用户的业务需求和策略，自动调整其弹性计算资源的管理服务。根据业务需求，您可以自动地调整其弹性计算资源。

示例代码：创建弹性伸缩组



说明：

确保您已经开通了弹性伸缩服务。

```
using System;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Core.Exceptions;
using Aliyun.Acs.Ess.Model.V20140828;

class Sample
{
    static void Main(string[] args)
    {
        // 创建客户端实例
        IClientProfile clientProfile = DefaultProfile.GetProfile("<
your-region-id>", "<your-access-key-id>", "<your-access-key-secret>");
        DefaultAcsClient client = new DefaultAcsClient(clientProfile);

        try
        {
            // 创建伸缩组 (Scaling Group)
            CreateScalingGroupRequest csgRequest = new CreateScal
ingGroupRequest();
            csgRequest.MaxSize = 10;
            csgRequest.MinSize = 2;
        }
    }
}
```

```

        CreateScalingGroupResponse csgResponse = client.GetAcsResponse(csgRequest);
        String scalingGroupId = csgResponse.ScalingGroupId;
        Console.WriteLine("ScalingGroupId: {0}", scalingGroupId);

        // 创建伸缩配置(Scaling Configuration)
        CreateScalingConfigurationRequest cscRequest = new CreateScalingConfigurationRequest();
        cscRequest.ScalingGroupId = scalingGroupId;
        cscRequest.ImageId = "centos_7_04_64_20G_alibase_201701015.vhd";
        cscRequest.InstanceType = "ecs.t1.xsmall";
        cscRequest.SecurityGroupId = "G00000000123456789"; // 可以通过 ECS 的 DescribeSecurityGroups 接口获取 SecurityGroupId
        CreateScalingConfigurationResponse cscResponse = client.GetAcsResponse(cscRequest);
        Console.WriteLine("ScalingConfigurationId: {0}", cscResponse.ScalingConfigurationId);
    }
    catch (ServerException e)
    {
        Console.WriteLine(e.ErrorCode);
        Console.WriteLine(e.ErrorMessage);
    }
    catch (ClientException e)
    {
        Console.WriteLine(e.ErrorCode);
        Console.WriteLine(e.ErrorMessage);
    }
}

```

示例代码：指定 I/O 优化实例

```

CreateScalingConfigurationRequest cscRequest = new CreateScalingConfigurationRequest();
cscRequest.ScalingGroupId = scalingGroupId;
cscRequest.ImageId = "centos_7_04_64_20G_alibase_201701015.vhd";
cscRequest.InstanceType = "ecs.t1.xsmall";
cscRequest.IoOptimized = "optimized";
cscRequest.SecurityGroupId = "G00000000123456789";

```

### 3.6 内容分发网络CDN

本示例介绍如何使用阿里云.NET SDK调用CDN的**AddCdnDomainRequest**接口添加加速域名。

内容分发网络 ( Content Delivery Network , 简称CDN ) 是建立并覆盖在承载网之上、由分布在不同区域的边缘节点服务器群组成的分布式网络。

在调用该接口时，注意：

- 创建加速域名之前，必须先开通CDN服务。
- 加速域名必须已备案完成。

- 源站内容如果不在阿里云平台上，需要进行审核，审核工作会在下一工作日前完成。

```
using System;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Core.Exceptions;
using Aliyun.Acs.Cdn.Model.V20141111;

class Sample
{
    static void Main(string[] args)
    {
        // 创建客户端实例
        IClientProfile clientProfile = DefaultProfile.GetProfile("<
your-region-id>", "<your-access-key-id>", "<your-access-key-secret>");
        DefaultAcsClient client = new DefaultAcsClient(clientProfile);
        try
        {
            // 创建API请求并设置参数
            AddCdnDomainRequest request = new AddCdnDomainRequest();
            request.CdnType = "web";
            request.DomainName = "test.com";
            request.Sources = "test.com";
            request.SourceType = "domain";

            // 发起请求
            AddCdnDomainResponse response = client.GetAcsResponse(
request);
            Console.WriteLine("Success");
        }
        catch (ServerException e)
        {
            Console.WriteLine(e.ErrorCode);
            Console.WriteLine(e.ErrorMessage);
        }
        catch (ClientException e)
        {
            Console.WriteLine(e.ErrorCode);
            Console.WriteLine(e.ErrorMessage);
        }
    }
}
```

## 4 常见错误

您可以参考本文档中提供的常见问题排查SDK使用中的问题。

| 错误码                         | 原因                                           | 解决方法                                      |
|-----------------------------|----------------------------------------------|-------------------------------------------|
| SDK.InvalidRegionId         | SDK无法自动获取调用的产品在指定地域的endpoint。                | 请检查您的endpoint是否正确。                        |
| InvalidAccessKeyId.NotFound | 设置的AccessKey ID无效。                           | 请检查您的AccessKey ID是否正确。                    |
| SDK.ConnectionReset         | 发起请求时无法连接到服务端点。                              | 请检测您的网络是否可用。                              |
| SDK.InvalidProfile          | 您在初始化DefaultAcsClient时提供的DefaultProfile参数无效。 | 请确保您提供的DefaultProfile参数对象已正确初始化，并配置了相应参数。 |
| SDK.SessionTokenExpired     | 您在初始化DefaultAcsClient时提供的DefaultProfile参数无效。 | 请确保您提供的SessionToken参数在有效期内，一般有效期为一小时。     |
| SignatureDoesNotMatch       | 您提供的SessionToken无效或已过期。                      | 请确保您提供的DefaultProfile参数对象已正确初始化，并配置了相应参数。 |
| SDK.InvalidProfile          | 您在请求时提供的AccessKey Secret与AccessKey ID不匹配。    | 请确保您提供的AccessKey Secret有效。                |