阿里云
容器服务

开发指南

文档版本：20190321

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:** 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:** 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:** 您也可以通过按Ctrl + A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**。
`courier` 字体	命令。	执行 `cd /d C:/windows` 命令，进入Windows系统文件夹。
*## *	表示参数、变量。	`bae log list --instanceid` *`Instance_ID`*
[]或者[a|b]	表示可选项，至多选择一个。	`ipconfig` *`[-all|-t]`*
{}或者{a|b}	表示必选项，至多选择一个。	`swich` *`{stand | slave}`*

目录

1 通过 CLI 使用容器服务

1.1 概述

阿里云命令行工具是用 Go 语言编写的，基于阿里云 OpenAPI 打造的，用于管理阿里云资源的工具。通过下载和配置该工具，您可以在一个命令行方式下使用多个阿里云产品。

关于阿里云命令行工具的详细介绍以及安装说明，参见 阿里云 CLI 文档。

容器服务是 RESTful 风格的 API。目前容器服务支持 Swarm 和 Kubernetes 两种调度方式，下面是目前容器服务开放的 API 列表。

说明:

关于阿里云容器服务已支持的 API，参见 容器服务 API 参考。

API 接口	解释	适用范围
查看所有集群实例	查看您在容器服务中创建的所有集群（包括 Swarm 和 Kubernetes 集群）。	Swarm 集群和 Kubernetes 集群
查看集群实例	根据集群 ID，查看集群的详细信息。	Swarm 集群和 Kubernetes 集群
创建集群实例	创建一个新的集群实例，并新建指定数量的节点。	Swarm 集群和 Kubernetes 集群
扩容集群	增加集群中节点的数量。	Swarm 集群和 Kubernetes 集群
添加已有实例	添加已有实例到集群。	Swarm 集群和 Kubernetes 集群
移除集群内节点	根据集群 ID，以及节点 IP 从容器集群中移除节点。	Swarm 集群
查看镜像列表	查看目前所支持的地域下支持的镜像列表。	Swarm 集群
重置节点	重置集群中的某个节点。	Swarm 集群
删除集群实例	根据集群 ID，删除集群实例，并释放集群所有节点资源。	Swarm 集群和 Kubernetes 集群
获取集群证书	根据集群 ID，获取集群的证书信息。	Swarm 集群

1.2 查看所有集群实例

查看您在容器服务中创建的所有集群，包括 Swarm 和 Kubernetes 集群。具体的 API 描述，参见 容器服务 API 参考。

适用范围

Swarm 集群和 Kubernetes 集群。

API请求响应

请求格式

```
aliyun cs GET /clusters
```

响应结果

```
[
    {
        "agent_version": "string",
        "cluster_id": "string",
        "created": "datetime",
        "external_loadbalancer_id": "string",
        "master_url": "string",
        "name": "string",
        "network_mode": "string",
        "region_id": "string",
        "security_group_id": "string",
        "size": "numbers",
        "state": "string",
        "updated": "datetime",
        "vpc_id": "string",
        "vswitch_id": "string"
    }
]
```

1.3 查看集群实例

根据集群 ID，查看集群的详细信息。具体的 API 描述，参见 容器服务 API 参考。

适用范围

Swarm 集群和 Kubernetes 集群。

API请求响应

请求格式

```
aliyun cs GET /clusters/<cluster_id>
```

响应结果

```
{
```

```
    "agent_version": "string",
    "cluster_id": "string",
    "created": "datetime",
    "external_loadbalancer_id": "string",
    "master_url": "string",
    "name": "string",
    "network_mode": "string",
    "region_id": "string",
    "security_group_id": "string",
    "size": "numbers",
    "state": "string",
    "updated": "datetime",
    "vpc_id": "string",
    "vswitch_id": "string"
}
```

1.4 创建集群实例

创建一个新的集群实例，并新建指定数量的节点。具体的 API 描述，参见 容器服务 API 参考。

适用范围

Swarm 集群和 Kubernetes 集群。

API请求响应

请求格式

```
aliyun cs  POST /clusters --header "Content-Type=application/json" --
body "$(cat create.json)"
```

参数说明：

- `--header`需要指定 `Content-Type` 为 `application/json`。
- `--body` 是要发送给服务端的 body 内容，可以从本地文件读取，需要是有效的 JSON 格式。`create.json` 的内容如下所示。

swarm 集群

```
{
"password": "ECS机器SSH密码",
"region_id": "地域ID",
"instance_type": "实例规格",
"name": "集群名称",
"size": 节点数量,
"network_mode": "网络类型，目前仅支持vpc",
"vpc_id": "VPC示例ID",
"vswitch_id": "VPC下的交换机ID",
"subnet_cidr": "容器CIDR",
"data_disk_category": "数据盘类型",
"data_disk_size": 数据盘大小,
"need_slb": 是否默认创建SLB,
"io_optimized":"是否IO优化，VPC下目前默认为IO优化",
"ecs_image_id": "镜像ID",
"release_eip_flag": "是否需要在集群配置完成后释放EIP"
```

```
}
```

Kubernetes集群(单可用区)

```
{
"disable_rollback": "失败是否回滚",
"name": "集群名称",
"timeout_mins": 集群创建超时时间,
"cluster_type": "Kubernetes",
"region_id": "地域",
"vpcid": "VPC ID",
"zoneid": "可用区",
"vswitchid": "交换机ID",
"container_cidr": "容器POD CIDR",
"service_cidr": "服务CIDR",
"ssh_flags": "是否开放公网SSH登陆",
"cloud_monitor_flags":"是否安装云监控插件",
"login_password": "节点SSH登陆密码，和key_pair二选一",
"key_pair":"keypair名称，和login_password 二选一",
"master_instance_charge_type":"Master实例付费类型，PostPaid|PrePaid",
"master_period_unit":"包年包月单位，Month,Year，只有在PrePaid下生效",
"master_period":"包年包月时长，只有在PrePaid下生效",
"master_auto_renew":"Master节点是否自动续费",
"master_auto_renew_period":"Master节点续费周期",
"master_instance_type": "Master实例规格",
"master_system_disk_category": "Master系统盘类型",
"master_system_disk_size":"Master节点系统盘大小",
"master_data_disk":"Master节点是否挂载数据盘",
"master_data_disk_category":"Master节点数据盘类型",
"master_data_disk_size":"Master节点数据盘大小",
"worker_instance_charge_type":"Worker节点付费类型PrePaid|PostPaid",
"worker_period_unit":"包年包月单位，Month,Year，只有在PrePaid下生效",
"worker_period":"包年包月时长，只有在PrePaid下生效",
"worker_auto_renew":"Worker节点自动续费true|false",
"worker_auto_renew_period":"Worker节点续费周期",
"worker_instance_type": "Worker实例规格",
"worker_system_disk_category": "Worker系统盘类型",
"worker_system_disk_size": "Worker节点系统盘大小",
"worker_data_disk":"Worker节点是否挂载数据盘",
"worker_data_disk_category":"Worker节点数据盘类型",
"worker_data_disk_size":"Worker节点数据盘大小",
"num_of_nodes": "Worker节点数",
"snat_entry": 是否配置SNATEntry,
"public_slb":"是否创建公网API Server对应的SLB"
}
```

Kubernetes集群(多可用区)

```
{
"disable_rollback":"失败是否回滚",
"name": "集群名称",
"timeout_mins": 集群创建超时时间,
"cluster_type": "Kubernetes",
"region_id": "地域"
"multi_az": true,
"vpcid": "VPC ID ",
"container_cidr": "容器 CIDR",
"service_cidr": "服务 CIDR",
"vswitch_id_a": "第一个可用区交换机ID",
"vswitch_id_b": "第二个可用区交换机ID",
"vswitch_id_c": "第三个可用区交换机ID",
"master_instance_type_a": "第一个可用区Master节点实例规格",
```

```
"master_instance_type_b": "第二个可用区Master节点实例规格",
"master_instance_type_c": "第三个可用区Master节点实例规格",
"master_instance_charge_type":"Master实例付费类型, PostPaid|PrePaid",
"master_period_unit":"包年包月单位, Month,Year, 只有在PrePaid下生效",
"master_period":"包年包月时长, 只有在PrePaid下生效",
"master_auto_renew":"Master节点是否自动续费",
"master_auto_renew_period":"Master节点续费周期",
"master_system_disk_category": "Master节点系统盘类型",
"master_system_disk_size": "Master节点系统盘大小",
"master_data_disk":"Master节点是否挂载数据盘",
"master_data_disk_category":"Master节点数据盘类型",
"master_data_disk_size":"Master节点数据盘大小",
"worker_instance_type_a": "第一个可用区Worker节点实例规格",
"worker_instance_type_b": "第二个可用区Worker节点实例规格",
"worker_instance_type_c": "第三个可用区Worker节点实例规格",
"worker_instance_charge_type":"Worker节点付费类型PrePaid|PostPaid",
"worker_period_unit":"包年包月单位, Month,Year, 只有在PrePaid下生效",
"worker_period":"包年包月时长, 只有在PrePaid下生效",
"worker_auto_renew":"Worker节点自动续费true|false",
"worker_auto_renew_period":"Worker节点续费周期",
"worker_system_disk_category": "Worker节点系统盘类型",
"worker_system_disk_size": "Worker节点系统盘大小",
"worker_data_disk":"Worker节点是否挂载数据盘",
"worker_data_disk_category":"Worker节点数据盘类型",
"worker_data_disk_size":"Worker节点数据盘大小",
"num_of_nodes_a": "第一个可用区Worker节点数",
"num_of_nodes_b": "第二个可用区Worker节点数",
"num_of_nodes_c": "第三个可用区Worker节点数",
"ssh_flags": "是否开放公网SSH 登陆",
"login_password": "SSH 登陆密码",
"cloud_monitor_flags":"是否安装云监控插件",
"public_slb":"是否创建公网API Server对应的SLB"
}
```

托管Kubernetes集群

```
{
"disable_rollback": "失败是否回滚",
"name": "集群名称",
"timeout_mins": 集群创建超时时间,
"cluster_type": "ManagedKubernetes",
"region_id": "地域, 目前仅支持北京和杭州地域(cn-beijing,cn-hangzhou)",
"vpcid": "VPC ID",
"zoneid": "可用区",
"vswitchid": "交换机ID",
"container_cidr": "容器POD CIDR",
"service_cidr": "服务CIDR",
"cloud_monitor_flags":"是否安装云监控插件",
"login_password": "节点SSH登陆密码, 和key_pair二选一",
"key_pair":"keypair名称, 和login_password 二选一",
"worker_instance_charge_type":"Worker节点付费类型PrePaid|PostPaid",
"worker_period_unit":"包年包月单位, Month,Year, 只有在PrePaid下生效",
"worker_period":"包年包月时长, 只有在PrePaid下生效",
"worker_auto_renew":"Worker节点自动续费true|false",
"worker_auto_renew_period":"Worker节点续费周期",
"worker_instance_type": "Worker实例规格",
"worker_system_disk_category": "Worker系统盘类型",
"worker_system_disk_size": "Worker节点系统盘大小",
"worker_data_disk":"是否挂载数据盘 true|false",
"worker_data_disk_category":"数据盘类型",
"worker_data_disk_size":"数据盘大小",
"num_of_nodes": "Worker节点数",
"snat_entry": 是否配置SNATEntry,
```

```
}ntry": 是否配置SNATEntry,
    }
```

响应结果

```
{
    "cluster_id": "c61cf530524474386a7ab5a1c192a0d57",
    "request_id": "348D4C9C-9105-4A1B-A86E-B58F0F875575",
    "task_id": "T-5ad724ab94a2b109e8000004"
}
```

1.5 扩容集群

增加集群中节点的数量。具体的 API 描述，参见 容器服务 API 参考。

适用范围

Swarm 集群和 Kubernetes 集群。

API 请求响应

请求格式

```
aliyun cs  PUT /clusters/<cluster_id> --header "Content-Type=
application/json" --body "$(cat scale.json)"
```

参数说明：

- `--header` 需要指定 Content-Type 为 application/json。
- `--body` 是要发送给服务端的 body 内容，可以从本地文件读取，需要是有效的 JSON 格式。

  `scale.json` 的内容如下所示。

Swarm集群

```
{
        "password": "ECS 机器 SSH 密码",
        "instance_type": "实例规格",
        "size": 节点数量,
        "data_disk_category": "数据盘类型",
        "data_disk_size": 数据盘大小,
        "io_optimized":"是否 IO 优化，VPC 下目前默认为 IO 优化",
        "ecs_image_id": "镜像 ID",
        "release_eip_flag": "是否需要在集群配置完成后释放 EIP"
}
```

Kubernetes集群

```
{ "disable_rollback": "失败是否回滚",
"timeout_mins": 集群创建超时时间,

"worker_instance_type": "Worker实例规格",
"login_password": "节点SSH登录密码",
"num_of_nodes": "Worker节点数"
```

```
}
```

响应结果

```
{
    "cluster_id": "c61cf530524474386a7ab5a1c192a0d57",
    "request_id": "348D4C9C-9105-4A1B-A86E-B58F0F875575",
    "task_id": "T-5ad724ab94a2b109e8000004"
}
```

1.6 添加已有实例

添加已有实例到集群。具体的 API 描述，参见 容器服务 API 参考。

适用范围

Swarm 集群和 Kubernetes 集群。

API 请求响应

请求格式

```
aliyun cs  POST /clusters/<cluster_id>/attach --header "Content-Type=
application/json" --body "$(cat attach.json)"
```

参数说明：

- `--header` **需要指定** `Content-Type` **为** `application/json`。
- `--body` **是要发送给服务端的 body 内容，可以从本地文件读取，需要是有效的 JSON 格式。**

  `attach.json` **的内容如下所示。**

```
{
        "password": "ECS 机器 SSH 密码",
        "instances": "ECS 示例数组",
        "ecs_image_id": "镜像 ID",
        "release_eip_flag": "是否需要在集群配置完成后释放 EIP"
}
```

响应结果

```
{
    "list": [
        {
            "code": "200",
            "instanceId": "i-2zee3oiwcyoz7kwdo8bt",
            "message": "successful"
        },
        {
            "code": "200",
            "instanceId": "i-2ze0lgm3y6iylcbtcypf",
            "message": "successful"
        }
    ],
    "task_id": "T-5a544aff80282e39ea000039"
```

```
}
```

1.7 重置节点

重置集群中的某个节点。具体的 API 描述，参见 容器服务 API 参考。

适用范围

Swarm 集群。

API请求响应

请求格式

```
aliyun cs  POST /clusters/<cluster_id>/instances/<instance_id>/reset
 --header "Content-Type=application/json" --body "$(cat reset.json)"
```

参数说明：

- `--header` 需要指定 `Content-Type` 为 `application/json`。
- `--body` 是要发送给服务端的 body 内容，可以从本地文件读取，需要是有效的 JSON 格式。

  `reset.json` 的内容如下所示。

```
{
        "password": "ECS 机器 SSH 密码",
        "ecs_image_id": "镜像 ID",
        "release_eip_flag": "是否需要在集群配置完成后释放 EIP"
}
```

响应结果

```
{
    "cluster_id": "c61cf530524474386a7ab5a1c192a0d57",
    "request_id": "348D4C9C-9105-4A1B-A86E-B58F0F875575",
    "task_id": "T-5ad724ab94a2b109e8000004"
}
```

2 Swarm API参考

2.1 简介

您可以使用本文档介绍的 API 对容器服务进行相关操作。

说明:

请确保在使用这些接口前，您已充分了解了容器服务说明、使用协议和收费方式。

术语表

术语	中文	说明
Cluster	集群	您的容器集群，一个集群可以部署多个应用。
Node	节点	您的容器集群中的某一个节点，目前只支持 ECS 实例。
Project	应用	一个复杂的应用可以由多个服务组合而成，最简单的应用可能只包含一个容器。
Service	服务	一组基于相同镜像和配置定义的容器，作为一个可伸缩的微服务。
Container	容器	Docker 容器运行时实例。

2.2 API 概述

容器服务的 API 主要分为3个部分：

- 集群管理接口
- 应用管理接口
- 触发器

集群管理接口

容器服务提供了一些用于集群管理的接口，例如创建集群、删除集群等。

集群管理接口列表：

API	描述
GetClusterList	查看所有集群实例
CreateCluster	创建集群实例
DeleteCluster	删除集群实例
GetClusterById	查看集群实例
GetClusterCerts	获取集群证书
UpdateClusterSizeById	更新集群节点数量

应用管理接口

应用管理提供了 Docker Remote API 兼容接口。您可以像访问单个 Docker Engine 一样，操作您的 Docker 集群。

应用管理接口列表：

API	描述
List Projects	查看应用实例列表
Create project	创建应用实例
Retrieve project	查看应用实例
Start project	启动应用实例
Stop project	停止应用实例
Kill project	终止应用实例
Update project	更新应用实例
Delete project	删除应用实例
List Services	查看服务实例列表
Retrieve service	查看服务实例
Start service	启动服务实例
Stop service	停止服务实例
Kill service	终止服务实例
Scale service	伸缩服务实例
Create volume	创建数据卷
View volume	查看数据卷

API	描述
List volumes	查看数据卷列表
Delete volume	删除数据卷

触发器

触发器是容器服务提供的简单快捷地进行持续部署的 API。更多详细信息参见 触发器。

2.3 更新历史

发布时间	更新	说明
2015-12-15	第一版确定	提供了集群管理的基本接口
2016-02-04	增加应用相关接口	提供了应用管理的基本接口

2.4 状态表

状态	说明
创建中（Launching）	集群正在申请相应的云资源
运行中（Running）	集群正在运行
创建失败（Failed）	集群申请云资源失败
启动中（Starting）	集群正在启动
停止中（Stopping）	集群正在停止运行
停止（Stopped）	集群停止运行
重启中（Restarting）	集群正在重启
升级中（Updating）	集群升级服务端版本
集群规模变更中（Scaling）	变更集群的节点数量
删除中（Deleting）	正在删除集群
已删除（Deleted）	集群删除成功

2.5 集群API调用方式

2.5.1 概述

对容器服务 API 接口的调用是通过向容器服务 API 的服务端地址发送 HTTP 请求，并按照接口说明在请求中加入相应请求参数来完成的。根据请求的处理情况，系统会返回处理结果。

1. 请求结构
2. 公共参数
3. 返回结果
4. 签名机制

2.5.2 请求结构

服务地址

阿里云容器服务的 Open API 接入地址为 cs.aliyuncs.com。

通信协议

支持通过 HTTP 或 HTTPS 通道进行请求通信。为了获得更高的安全性，推荐您使用 HTTPS 通道发送请求。

请求方法

使用 HTTP 的 PUT、 POST、 GET、 DELETE 等 HTTP Method 发送不同的请求。

请求参数

每个请求都需要包含公共请求参数和指定操作所特有的请求参数。

请求编码

请求及返回结果都使用 UTF-8 字符集进行编码。

2.5.3 公共参数

公共请求头部

公共请求参数是指每个接口都需要使用到的请求参数。

参数名称	说明	选项
Authorization	用于验证请求合法性的认证信息，采用 AccessKeyId:Signature 的形式。	Required

参数名称	说明	选项
Content-Length	RFC 2616 中定义的 HTTP 请求内容长度。	Required
Content-Type	RFC 2616 中定义的 HTTP 请求内容类型。	Required
Content-MD5	HTTP 协议消息体的 128-bit MD5 散列值转换成 BASE64 编码的结果。为了防止所有请求被篡改，建议所有请求都附加该信息。	Required
Date	请求的构造时间，目前只支持 GMT 格式。如果与 MNS 的服务器时间前后差异超过 15 分钟将返回本次请求非法。	Required
Host	访问 Host 值，例如：diku.aliyuncs.com。	Required
Accept	客户端需要的返回值类型，支持 application/json 和 application/xml。	Required
x-acs-version	API 版本号。目前版本号为 2015-12-15。	Required
x-acs-region-id	地域（Region）指的是 ECS 实例所在的物理位置。更多详细信息参见 地域概念 和 *ECS*查询可用地域列表。	Required
x-acs-signature-nonce	唯一随机数，用于防止网络重放攻击。您在不同请求间要使用不同的随机数值。	Required
x-acs-signature-method	用户签名方式，目前只支持 HMAC-SHA1。	Required

示例

```
GET /clusters HTTP/1.1
Host: cs.aliyuncs.com
Accept: application/json
User-Agent: cs-sdk-python/0.0.1 (Darwin/15.2.0/x86_64;2.7.10)
x-acs-signature-nonce: f63659d4-10ac-483b-99da-ea8fde61eae3
Authorization: acs <yourAccessKeyId>:<yourSignature>
x-acs-signature-version: 1.0
Date: Wed, 16 Dec 2015 11:18:47 GMT
x-acs-signature-method: HMAC-SHA1
```

```
Content-Type: application/json;charset=utf-8
X-Acs-Region-Id: cn-beijing
Content-Length: 0
```

公共返回头部

您发送的每次接口调用请求，无论成功与否，系统都会返回一个唯一识别码 RequestId。

示例

XML 示例：

```
<?xml version="1.0" encoding="UTF-8"?>
<!—结果的根结点-->
<接口名称+Response>
  | <!—返回请求标签-->
  | <RequestId>4C467B38-3910-447D-87BC-AC049166F216</RequestId>
  | <!—返回结果数据-->
</接口名称+Response>
```

JSON 示例：

```
{
    "RequestId": "4C467B38-3910-447D-87BC-AC049166F216"
    /* 返回结果数据 */
}
```

2.5.4 返回结果

调用 API 服务后返回数据采用统一格式。返回的 HTTP 状态码为 `2xx`，代表调用成功；返回的 HTTP 状态码为 `4xx` 或 `5xx`，代表调用失败。调用成功返回的数据格式主要有 XML 和 JSON 两种，外部系统可以在请求时传入参数来制定返回的数据格式，默认为 XML 格式。

为了便于您查看，本文档中的返回示例做了格式化处理，实际返回结果是没有换行、缩进等处理的。

2.5.5 签名机制

签名机制说明

Access Key ID 和 Access Key Secret 由阿里云官方颁发给访问者（可以通过阿里云官方网站申请和管理），其中 Access Key ID 用于标识访问者的身份；Access Key Secret 是用于加密签名字符串和服务器端验证签名字符串的密钥，必须严格保密，只有阿里云和用户知道。

容器服务会对每个访问的请求进行验证，每个向容器服务提交的请求，都需要在请求中包含签名（Signature）信息。容器服务通过使用 Access Key ID 和 Access Key Secret 进行对称加密的方法来验证请求的发送者身份。如果计算出来的验证码和提供的一样即认为该请求是有效的；否则，容器服务将拒绝处理这次请求，并返回 HTTP 403 错误。

用户可以在 HTTP 请求中增加授权（Authorization）的 Head 来包含签名信息，表明这个消息已被授权。

容器服务要求将签名包含在 HTTP Header 中，格式为 `Authorization: acs [Access Key Id]:[Signature]`。

`Signature` 的计算方法如下：

```
Signature = base64(hmac-sha1(VERB + "\n"
            + ACCEPT + "\n" +
            + Content-MD5 + "\n"
            + Content-Type + "\n"
            + Date + "\n"
            + CanonicalizedHeaders + "\n"
            + CanonicalizedResource))
```

- `VERB` 表示 HTTP 的 Method。比如示例中的 `PUT`。
- `Accept` 客户端需要的返回值类型，支持 application/json 和 application/xml。
- `Content-MD5` 表示请求内容数据的 MD5 值。
- `Content-Type` 表示请求内容的类型。
- `Date` 表示此次操作的时间，不能为空，目前只支持 GMT 格式。如果请求时间与 CAS 服务器时间相差超过 1 5分钟，CAS 会判定此请求不合法，并返回 400 错误。错误信息及错误码详见本文档第 5 部分。比如示例中的 `Thu, 17 Mar 2012 18:49:58 GMT`。
- `CanonicalizedHeaders` 表示 HTTP 中以 `x-acs-` 开始的字段组合。
- `CanonicalizedResource` 表示 HTTP 所请求资源的 URI (统一资源标识符)。比如示例中的 `/clusters?name=my-clusters&resource=new`。

 说明：

`CanonicalizedHeaders`（即以 `x-acs-` 开头的 header）在签名验证前需要符合以下规范：

1. 将所有以 `x-acs-` 为前缀的 HTTP 请求头的名字转换成小写字母。比如将 `X-ACS-Meta-Name: TaoBao` 转换为 `x-acs-meta-name: TaoBao`。阿里云规范请求头的名字是大小写不敏感的，建议全部使用小写。
2. 如果一个公共请求头的值部分过长，则需要处理其中的 `\t`、`\n`、`\r`、`\f`分隔符，将其替换为英文半角的空格。
3. 将上一步得到的所有 HTTP 阿里云规范头按照字典序进行升序排列。
4. 删除请求头和内容之间分隔符两端出现的任何空格。比如将 `x-acs-meta-name: TaoBao, Alipay` 转换为 `x-acs-meta-name:TaoBao,Alipay`。
5. 将所有的头和内容用 `\n` 分隔符分隔拼成最后的 `CanonicalizedHeaders`。

说明:

`CanonicalizedResource` 的格式规范: `CanonicalizedResource` 表示客户想要访问资源的规范描述，需要将子资源和 qurey 一同按照字典序，从小到大排列并以 `&` 为分隔符生成子资源字符串（`?` 后的所有参数）。

```
http://cs.aliyuncs.com/clusters?name=my-clusters&resource=new
```

`CanonicalizedResource` 应该为:

```
/clusters?name=my-clusters&resource=new
```

签名示例

示例概述

您可以通过该示例，了解加签的步骤。

示例使用的 accessKeyId 和 accessKeySecret 分别为 `access_key_id` 和 `access_key_secret`。推荐您使用自己的 OpenAPI 调用程序，来计算下面这个示例的加签串，您自己的加签结果和示例结果。

请求的示例如下:

```
POST http://cs.aliyuncs.com/clusters?param1=value1&param2=value2 HTTP/
1.1
Accept-Encoding: identity
Content-Length: 210
Content-MD5: 6U4ALMkKSj0PYbeQSHqgmA==
x-acs-version: 2015-12-15
Accept: application/json
User-Agent: cs-sdk-python/0.0.1 (Darwin/15.2.0/x86_64;2.7.10)
x-acs-signature-nonce: fbf6909a-93a5-45d3-8b1c-3e03a7916799
x-acs-signature-version: 1.0
Date: Wed, 16 Dec 2015 12:20:18 GMT
x-acs-signature-method: HMAC-SHA1
Content-Type: application/json;charset=utf-8
X-Acs-Region-Id: cn-beijing
Authorization: acs access_key_id:/ZmVlMDNkNDA1ZTQyMWViYWY1MTRhZG
VjODgxMDM4YzRiMzEzNTg0ZA==
{"password": "Just$test","instance_type": "ecs.m2.medium","name": "my-
test-cluster-97082734","size": 1,"network_mode": "classic","data_disk_
category": "cloud","data_disk_size": 10,"ecs_image_id": "m-253llee3l"}
```

请求构造过程

计算 `Content-Length` 和 `Content-MD5`

`Content-Length`: body 内容的长度。

说明:

示例 body 首位没有空格或换行符

```
body: {"password": "Just$test","instance_type": "ecs.m2.medium","name
": "my-test-cluster-97082734","size": 1,"network_mode": "classic","
data_disk_category": "cloud","data_disk_size": 10,"ecs_image_id": "m-
253llee3l"}
Content-Length: 210
```

`Content-MD5`: MD5 的计算过程。

```
body: {"password": "Just$test","instance_type": "ecs.m2.medium","name
": "my-test-cluster-97082734","size": 1,"network_mode": "classic","
data_disk_category": "cloud","data_disk_size": 10,"ecs_image_id": "m-
253llee3l"}
# 计算 body 的 md5 值
md5(body): e94e002cc90a4a3d0f61b790487aa098
# 将 md5 值转化成字节数组。将 md5 中的每两个十六进制位合并，转化为一个字节。
# 例如: e9 -> 11111111111111111111111111101001 -> -23
bytes(md5(body)): {[-23], [78], [0], [44], [-55], [10], [74], [61], [
15], [97], [-73], [-112], [72], [122], [-96], [-104]}
# 将得到的字节数组做一个 base64 转换
base64(bytes(md5(body))): 6U4ALMkKSj0PYbeQSHqgmA==
Content-MD5: 6U4ALMkKSj0PYbeQSHqgmA==
```

处理 `CanonicalizedHeaders`

```
# 将所有以‘x-acs-’开头的头部列出来
x-acs-version: 2015-12-15
x-acs-signature-nonce: ca480402-7689-43ba-acc4-4d2013d9d8d4
x-acs-signature-version: 1.0
x-acs-signature-method: HMAC-SHA1
X-Acs-Region-Id: cn-beijing
# 将请求名字变成小写，去掉每一行首尾的空格，并按照字典序进行排序。删除请求头和内容之
间分隔符两端出现的任何空格。
# 注意：最后一行没有换行符。
x-acs-region-id:cn-beijing
x-acs-signature-method:HMAC-SHA1
x-acs-signature-nonce:fbf6909a-93a5-45d3-8b1c-3e03a7916799
x-acs-signature-version:1.0
x-acs-version:2015-12-15
```

计算 `CanonicalizedResource`

示例得到的 CanonicalizedResource，长度应该为 27。

说明:

第一行行尾有一个 `\n` **的换行符。**

```
/clusters?param1=value1&param2=value2
```

计算 `Signature`

组装 SignatureString。示例中的加签字符串的长度为 307。除最后一行外，每一行行尾均有一个 \n 的换行符。

```
POST
application/json
6U4ALMkKSj0PYbeQSHqgmA==
application/json;charset=utf-8
Wed, 16 Dec 2015 12:20:18 GMT
x-acs-region-id:cn-beijing
x-acs-signature-method:HMAC-SHA1
x-acs-signature-nonce:fbf6909a-93a5-45d3-8b1c-3e03a7916799
x-acs-signature-version:1.0
x-acs-version:2015-12-15
/clusters?param1=value1&param2=value2
```

计算 Signature

```
# 使用 accessKeySecret 来对加签字符串进行加密，其中示例使用的 accessKeySecret
 是 access_key_secret。
hmac-sha1(SignatureString): fee03d405e421ebaf514adec881038c4b313584d
# 类似于 Content-MD5 的计算方式，将得到的加密串转化成字节数组。
# 将得到的字符数组做一个 base64 转换。得到最后的签名串。
base64(bytes(hmac-sha1(SignatureString))): ZmVlMDNkNDA1ZTQyMWVi
YWY1MTRhZGVjODgxMDM4YzRiMzEzNTg0ZA==
Signature: ZmVlMDNkNDA1ZTQyMWViYWY1MTRhZGVjODgxMDM4YzRiMzEzNTg0ZA==
```

完成

经过以上的处理，添加一些其他头部信息，最终构成的 HTTP 请求如下所示。

```
POST http://cs.aliyuncs.com/clusters?param1=value1&param2=value2 HTTP/
1.1
Accept-Encoding: identity
Content-Length: 210
Content-MD5: 6U4ALMkKSj0PYbeQSHqgmA==
x-acs-version: 2015-12-15
Accept: application/json
User-Agent: cs-sdk-python/0.0.1 (Darwin/15.2.0/x86_64;2.7.10)
x-acs-signature-nonce: fbf6909a-93a5-45d3-8b1c-3e03a7916799
x-acs-signature-version: 1.0
Date: Wed, 16 Dec 2015 12:20:18 GMT
x-acs-signature-method: HMAC-SHA1
Content-Type: application/json;charset=utf-8
X-Acs-Region-Id: cn-beijing
Authorization: acs access_key_id:/ZmVlMDNkNDA1ZTQyMWViYWY1MTRhZG
VjODgxMDM4YzRiMzEzNTg0ZA==
```

```
{"password": "Just$test","instance_type": "ecs.m2.medium","name": "my-
test-cluster-97082734","size": 1,"network_mode": "classic","data_disk_
category": "cloud","data_disk_size": 10,"ecs_image_id": "m-253llee3l"}
```

2.6 触发器

2.6.1 触发器

简介

触发器是容器服务提供的简单快捷地进行重新部署和资源伸缩的 API。

由于标准的 API 需要保证安全性，因此需要严格的鉴权。但是对于需要与其他三方系统（例如 Jenkins 或者其他持续集成的 CI/CD 系统）集成的场景来说，所需要的权限是有限的，可能仅仅需要消息通知。因此为了保证安全性与便捷性，带有部分鉴权策略，并且可以灵活调用的 API 被广泛应用于持续集成持续交付的场景中。

容器服务目前提供重新部署触发器和资源伸缩触发器。

- 重新部署触发器

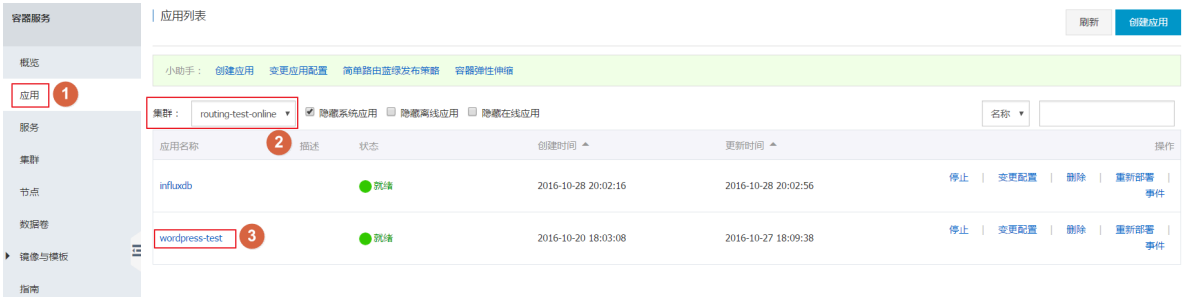
  您可以与自己的监控系统进行集成，当发现系统异常时进行重新部署；您也可以与容器 Hub 进行集成，当容器新镜像构建完成后，可以自动进行部署等。
- 资源伸缩触发器

  您可以通过调用资源伸缩触发器来实现容器伸缩。

创建触发器

操作流程

1. 登录 容器服务管理控制台。
2. 单击左侧导航栏中的 应用。
3. 选择目标应用所在的集群。
4. 选择目标应用并单击应用的名称。如下图所示。

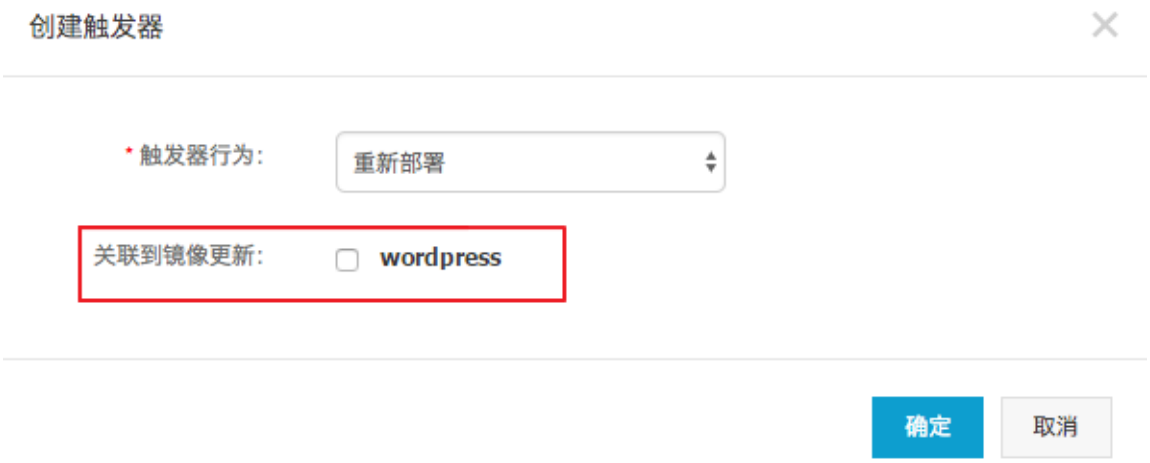


5. 在应用详情页面，单击 创建触发器。如下图所示。

6. 在 触发器行为 下拉框中选择 重新部署 或者 资源伸缩 并单击 确定。

- 重新部署

  当您对应用使用的镜像有写权限时，您可以勾选 关联到镜像更新。勾选后，当容器新镜像构建完成后，可以使用最新镜像自动对应用进行重新部署。



- 资源伸缩

  在 服务 下拉框中选择需要设置资源伸缩触发器的服务。

  说明:

您需要将所对应集群的 Agent 升级到最新版本，才能使用资源伸缩触发器。

此时生成的触发器地址即为 API 的地址。

后续操作

您可以通过三方集成系统进行触发，使用 GET 或者 POST 都可以进行触发，例如使用 `curl` 命令触发。

调用重新部署触发器：

```
curl 'https://cs.console.aliyun.com/hook/trigger?triggerUrl=YzI4YTk5Nz
FkZWZkYzQ2MTJiOWZkNTM1MzY2ZDU1M2NifGNvbGxlY3RkLWJlbmNobWFya3xyZWRlcGxv
eXwxOGlxbjc1Z25uMmVzfA==&secret=44586c6b466352395143584c397065
4ff5323d2509d546fdc1b33054b0928da8'
```

调用资源伸缩触发器：

说明：

调用资源伸缩触发器时，需要在触发器 URL 中手动添加以下参数：

参数名称	必填	语义	可选值
type	是	伸缩类型	缩容：scale_in；扩容：scale_out
step	是	伸缩数量	正整数，1~100

例如，调用下面的触发器会执行扩容五个容器的操作。

```
curl 'https://cs.console.aliyun.com/hook/trigger?triggerUrl=Y2IxZjI5Yz
hhYjIwMzRlMjBiYjc2OGUzYTlmZDgyNDAyfHdvcmRwcmVzcy10ZXN0fHNjYWxpbmd8MTkz
ZmEyMXFwZXVwMXw=&secret=53374142724e4e4a626f664a313131556e62c671
6cd0d97d096900b3ad42a9ad&type=scale_out&step=5'
```