容器服务API 参考

# 简介

## 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。 |

# API 概述

## 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 |
| Live-Migrate project | 迁移应用实例 |
| 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 |

集群API调用方式

# 调用方式

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

# 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。

# 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 |
| 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-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 | Required |
|  | 同的随机数值。 |  |
| 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 ACSbW2iBbyX0Pk9N:mp/6Wdr18V3qAc5gbgsqMe/iB0c 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 \*/

}

# 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。

集群API列表

# 查看所有集群实例

描述

说明：查看您在容器服务中创建的所有集群。如果传递了 name 参数，则只返回指定集群的详情。

## 请求信息

请求行 RequestLine

GET /clusters?name={name} HTTP/1.1

### 请求行参数 URI Param

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 描述 |
| name | string | 否 | 集群名称 |

特有请求头 RequestHead

无，请参考 公共请求头部。

### 请求体 RequestBody

无

## 返回信息

返回行 ResponseLine

HTTP/1.1 200 OK

特有返回头 ResponseHead

无，请参考 公共返回头部。

### 返回体 ResponseBody

[

{

"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"

}

]

### 返回体解释

Cluster 的格式

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| agent\_version | string | Agent 版本号。 |
| cluster\_id | String | 集群 ID，集群的唯一标识。 |
| Created | string | 集群的创建时间。 |
| external\_loadbalancer\_id | String | 集群负载均衡服务的 ID。 |
| master\_url | string | 集群 Master 地址，您可以通过该地址连接您的集群进行相关操  作。更多详细信息，参见 通过  Docker 工具连接集群。 |
| Name | string | 集群名称，由您在创建集群时指定，在每个用户下唯一。 |
| network\_mode | String | 集群网络模式（经典网络: classic、VPC 网络: vpc）。 |
| region\_id | String | 集群所在地域 ID。更多详细信息，参见 地域。 |
| security\_group\_id | String | 安全组 ID。 |
| Size | String | 节点数。 |
| State | String | 集群状态。更多详细信息，参见集群的生命周期。 |
| Updated | string | 最后更新时间。 |
| vpc\_id | string | VPC ID。 |
| vswitch\_id | string | vswitch ID。 |

## 示例

### 请求示例

GET /clusters HTTP/1.1

<公共请求头>

### 返回示例

HTTP/1.1 200 OK

<公共响应头>

[

{

"agent\_version": "0.5-e56dab3",

"cluster\_id": "c978ca3eaacd3409a9437db07598f1f69",

"created": "2015-12-11T03:52:40Z",

"external\_loadbalancer\_id": "1518f2b7e4c-cn-beijing-btc-a01",

"master\_url": "https://182.92.245.56:17589",

"name": "my-python-cluster-039de960",

"network\_mode": "classic",

"region\_id": "cn-beijing",

"security\_group\_id": "sg-25yqjuxhz",

"size": 5,

"state": "running",

"updated": "2015-12-15T15:01:58Z",

"vpc\_id": "",

"vswitch\_id": ""

},

{

"agent\_version": "0.5-e56dab3",

"cluster\_id": "c1eb19e0093204cbb86c3a80334d2129e",

"created": "2015-12-15T14:26:58Z",

"external\_loadbalancer\_id": "151a6099de1-cn-beijing-btc-a01",

"master\_url": "https://182.92.245.56:11905",

"name": "my-test-cluster-002b3f3d",

"network\_mode": "classic",

"region\_id": "cn-beijing",

"security\_group\_id": "sg-25rg2ws9f",

"size": 1,

"state": "running",

"updated": "2015-12-15T14:43:55Z",

"vpc\_id": "",

"vswitch\_id": ""

}

]

# 创建集群实例

描述

说明：创建一个新的集群实例，并新建指定数量的节点。

## 请求信息

请求行 RequestLine

POST /clusters HTTP/1.1

请求行参数 URI Param

无

特有请求头 RequestHead

无，请参考 公共请求头部。

### 请求体 RequestBody

ClusterCreation { "name": "string",

"size": "number",

"network\_mode": "string",

"subnet\_cidr": "string",

"instance\_type": "string",

"vpc\_id": "string",

"vswitch\_id": "string",

"password": "string",

"data\_disk\_size": "number",

"data\_disk\_category": "number"

"ecs\_image\_id": "string",

}

### 请求体解释

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 必须 | 描述 |
| name | String | 是 | 集群名称, 集群名称可以使用大小写英文字母、中文、数字、中划线  。 |
| size | int | 是 | 集群 ECS 节点数量。 |
| instance\_type | string | 是 | ECS 规格类型代码。更多详细信息，参见实例资源规格对照表。 |
| network\_mode | String | 是 | 集群网络模式（经典网  络: classic、VPC 网络 : vpc） |
| subnet\_cidr | String | 是 | 集群可以使用的网络地  址块，例如  ：192.168.24.0/22。  只有网络模式为 vpc  的时候，才需要设置该字段。 |
| vpc\_id | string | 是 | VPC 网络 ID。只有网络模式为 vpc 的时候  ，才需要设置该字段。  更多详细信息，参见  VPC 网络介绍。 |
| vswitch\_id | string | 是 | VPC 网络的交换机  ID。只有网络模式为  vpc 的时候，才需要设置该字段。 |
| password | string | 是 | root 账号密码。 |
| data\_disk\_category | string | 是 | ECS 使用的磁盘类型  。更多详细信息，参见 |
|  |  |  | 磁盘种类表。 |
| data\_disk\_size | number | 是 | 节点共享磁盘大小。 |
| ecs\_image\_id | string | 可选 | ECS 使用的系统镜像  ID。 |

### ecs\_image\_id 列表

|  |  |  |
| --- | --- | --- |
| 地域 | 系统 | ecs\_image\_id |
| 北京 | Ubuntu 14.04 64 位 | m-25i16eaw8 |
| 北京 | CentOS 7.0 64 位 | m-25ksqahve |
| 杭州 | Ubuntu 14.04 64 位 | m-2313ixw4g |
| 杭州 | CentOS 7.0 64 位 | m-23i32bks6 |

如果您需要自定义集群的 ECS 镜像的 ID，需要保证 ECS 镜像满足以下条件：

* 操作系统：Ubuntu、Centos。
* Linux Kernel version ≥ 3.18，用于支持 overlayfs 以及 overlay network。 - 镜像中删除 /etc/docker/key.json 文件。

## 返回信息

返回行 ResponseLine

HTTP/1.1 202 Accepted

特有返回头 ResponseHead

无，请参考 公共返回头部。

返回体 ResponseBody

{ “cluster\_id”:”string”, “request\_id”:”string”}

## 示例

### 请求示例

POST /clusters HTTP/1.1

<公共请求头>

{

"password": "Just$test",

"instance\_type": "ecs.s2.small",

"name": "my-cluster-001",

"size": 2,

"network\_mode": "classic",

"data\_disk\_category": "cloud",

"data\_disk\_size": 20,

"charge\_type": "PayByTraffic",

"ecs\_image\_id":"m-xx251ll"

}

### 返回示例

HTTP/1.1 202 Accepted

<公共响应头>

{

"cluster\_id":"c01994faf5f364cc79b6ffb604061fa4a",

"request\_id":"b44afc3c-46a4-4087-a215-c333a1218316"

}

# 删除集群实例

描述

说明：根据集群 ID，删除集群实例，并释放集群所有节点资源。

## 请求信息

请求行 RequestLine

DELETE /clusters/{cluster\_id} HTTP/1.1

### 请求行参数 URI Param

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 描述 |
| cluster\_id | string | 是 | 集群 ID |

特有请求头 RequestHead

无，请参考 公共请求头部。

### 请求体 RequestBody

无

## 返回信息

返回行 ResponseLine

HTTP/1.1 202 Accepted

特有返回头 ResponseHead

无，请参考 公共返回头部。

### 返回体 ResponseBody

无

## 示例

### 请求示例

DELETE /clusters/Cccfd68c474454665ace07efce924f75f HTTP/1.1

<公共请求头>

### 返回示例

HTTP/1.1 202 Accepted

<公共响应头>

# 查看集群实例

## 描述

说明：根据集群 ID，查看集群的详细信息

## 请求信息

### 请求行 RequestLine

|  |  |  |  |
| --- | --- | --- | --- |
| GET /clusters/{cluster\_id} HTTP/1.1 | | | |
| 请求行参数 URI Param | |  |  |
| 名称 | 类型 | 是否必须 | 描述 |
| cluster\_id | string | 是 | 集群 ID |
| 特有请求头 RequestHead    无，请参考 公共请求头部。    请求体 RequestBody    无    返回信息    返回行 ResponseLine | |  |  |
| HTTP/1.1 200 OK | |  |  |

特有返回头 ResponseHead

无，请参考 公共返回头部。

### 返回体 ResponseBody

{

"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"

}

### 返回体解释

Cluster 的格式

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| agent\_version | string | Agent 版本号。 |
| cluster\_id | String | 集群 ID，集群的唯一标识。 |
| created | string | 集群的创建时间。 |
| external\_loadbalancer\_id | String | 集群负载均衡服务的 ID。 |
| master\_url | string | 集群 Master 地址，您可以通过该地址连接您的集群进行相关操  作。更多详细信息，参见 通过  Docker 工具连接集群。 |
| name | string | 集群名称，由您在创建集群时指定，在每个用户下唯一。 |
| network\_mode | String | 集群网络模式（经典网络: classic、VPC 网络: vpc）。 |
| region\_id | String | 集群所在地域 ID。更多详细信息，参见 地域。 |
| security\_group\_id | String | 安全组 ID。 |
| size | String | 节点数。 |
| state | String | 集群状态。更多详细信息，参见集群的生命周期。 |
| updated | string | 最后更新时间。 |
| vpc\_id | string | VPC ID。 |
| vswitch\_id | string | vswitch ID。 |

## 示例

### 请求示例

GET /clusters/C5b5e80b0b64a4bf6939d2d8fbbc5ded7 HTTP/1.1 <公共请求头>

### 返回示例

HTTP/1.1 200 Ok

<公共响应头>

{

"agent\_version": "0.5-e56dab3",

"cluster\_id": "c978ca3eaacd3409a9437db07598f1f69",

"created": "2015-12-11T03:52:40Z",

"external\_loadbalancer\_id": "1518f2b7e4c-cn-beijing-btc-a01",

"master\_url": "https://182.92.245.56:17589",

"name": "my-python-cluster-039de960",

"network\_mode": "classic",

"region\_id": "cn-beijing",

"security\_group\_id": "sg-25yqjuxhz",

"size": 5,

"state": "running",

"updated": "2015-12-15T15:01:58Z",

"vpc\_id": "",

"vswitch\_id": ""

}

# 获取集群证书信息

## 描述

根据集群 ID，获取集群的证书信息。使用 Docker Client 操作集群时，需要使用该证书进行访问。更多详细信息，参见 配置证书。

## 请求信息

请求行 RequestLine

GET /clusters/{cluster\_id}/certs HTTP/1.1

### 请求行参数 URI Param

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 描述 |
| cluster\_id | string | 是 | 集群 ID |

特有请求头 RequestHead

无，请参考 公共请求头部。

### 请求体 RequestBody

无

## 返回信息

返回行 ResponseLine

HTTP/1.1 200 OK

特有返回头 ResponseHead

无，请参考 公共返回头部。

### 返回体 ResponseBody

{

"ca": "string",

"cert": "string",

"key": "string"

}

### 返回体解析

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| ca | string | 认证机构证书，ca.perm |
| cert | string | 用户公钥证书，cert.perm |
| key | string | 用户私钥证书，key.perm |

## 示例

### 请求示例

GET /clusters/Cccfd68c474454665ace07efce924f75f/certs HTTP/1.1

<公共请求头>

### 返回示例

HTTP/1.1 200 OK

<公共响应头>

{

"ca": "-----BEGINCERTIFICATE-----ca content-----ENDCERTIFICATE-----\n",

"cert": "-----BEGINCERTIFICATE-----cert content-----ENDCERTIFICATE-----\n",

"key": "-----BEGINRSAPRIVATEKEY-----key content-----ENDRSAPRIVATEKEY-----\n"

}

# 集群扩容

描述

说明：增加集群中节点的数量。

## 请求信息

### 请求行 RequestLine

|  |  |  |  |
| --- | --- | --- | --- |
| PUT /clusters/{cluster\_id} HTTP/1.1 | | | |
| 请求行参数 URI Param | |  |  |
| 名称 | 类型 | 是否必须 | 描述 |
| cluster\_id | string | 是 | 集群 ID |
| 特有请求头 RequestHead    无，请参考 公共请求头部。    请求体 RequestBody | |  |  |
| {  "password": "string",  "instance\_type": "string",  "size": "number", | |  |  |

"data\_disk\_category": "number"

"data\_disk\_size": "number",

"ecs\_image\_id": "string",

}

### 请求体解析

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 描述 |
| password | string | 是 | ECS 实例密码。 |
| instance\_type | String | 是 | ECS 规格类型代码。更多详细信息，参见实例资源规格对照表。 |
| size | int | 是 | 节点的总数量，要大于现有节点数量。 |
| data\_disk\_category | Number | 否 | ECS 使用的磁盘类型  。更多详细信息，参见磁盘种类表。 |
| data\_disk\_size | Number | 否 | 节点共享磁盘大小（单位：GB）。 |
| ecs\_image\_id | String | 是 | ECS 使用的系统镜像  ID。 |

### ecs\_image\_id 列表

|  |  |  |
| --- | --- | --- |
| 地域 | 系统 | ecs\_image\_id |
| 北京 | Ubuntu 14.04 64 位 | m-25i16eaw8 |
| 北京 | CentOS 7.0 64 位 | m-25ksqahve |
| 杭州 | Ubuntu 14.04 64 位 | m-2313ixw4g |
| 杭州 | CentOS 7.0 64 位 | m-23i32bks6 |

如果您需要自定义集群的 ECS 镜像的 ID，需要保证 ECS 镜像满足以下条件：

* 操作系统：Ubuntu、Centos。
* Linux Kernel version ≥ 3.18，用于支持 overlayfs 以及 overlay network。 - 镜像中删除 /etc/docker/key.json 文件。

## 返回信息

返回行 ResponseLine

HTTP/1.1 200 OK

特有返回头 ResponseHead

无，请参考 公共返回头部。

### 返回体 ResponseBody

无

## 示例

### 请求示例

PUT /clusters/Cccfd68c474454665ace07efce924f75f HTTP/1.1

<公共请求头>

{

"password": "password",

"instance\_type": "ecs.s3.large",

"size": 2,

"data\_disk\_category": "cloud\_ssd"

"data\_disk\_size": 500,

"ecs\_image\_id": "ubuntu1404\_64\_20G\_aliaegis\_20150130.vhd",

}

### 返回示例

HTTP/1.1 202 Accepted

<公共响应头>

应用API调用方式

# 调用方式

应用管理 REST API 需要指向集群的接入点地址, 并通过自签名证书的 HTTPS 请求和集群进行交互。

## 获取集群 Endpoint 和证书

### 控制台方式

登录 容器服务管理控制台。

单击左侧导航栏中的 集群。

选择需要查看的集群并单击 管理。



您可以查看集群的 endpoint 并单击 下载证书 下载集群证书。



通过 API 访问，您需要将截图里的 tcp 需要改为 https。

### 编程方式获取

您需要先通过集群管理的 API 获取:

获取集群的 master\_url 字段值。更多详细信息，参见 查看集群信息。

获取集群的证书。更多详细信息，参见 查看集群证书。

API 返回结果：

{

"ca": "string", ##认证机构证书，ca.pem

"cert": "string", ##用户公钥证书，cert.pem

"key": "string" ##用户私钥证书，key.pem

}

推荐将返回结果的三个 string 的内容保存为一个目录下的三个文件 ca.pem、cert.pem、key.pem。大部分的工具或编程框架都是以文件的方式加载 https 证书.

## 调用应用管理的 API

假设您的集群名称为 ClusterName，并且已经将上面三个证书存储到 ~/.docker/aliyun/ClusterName 目录下。上面获得的 master\_url 地址为 https://123.123.123.123:1234。

### 应用 API 列表

详见 应用API列表。

下面以查看应用列表接口为例 (context path 为 /projects/)。

### curl 方式

# 提示: 请注意你的 curl 版本，您可能需要升级你的 curl.

curl --insecure --cert ~/.docker/aliyun/ClusterName/cert.pem --key ~/.docker/aliyun/ClusterName/key.pem https://123.123.123.123:1234/projects/

### PHP 方式

|  |
| --- |
| <?php  $ch = curl\_init();  curl\_setopt($ch, CURLOPT\_URL, "https://123.123.123.123:1234/projects/"); curl\_setopt($ch, CURLOPT\_SSLKEY, "~/.docker/aliyun/ClusterName/key.pem"); curl\_setopt($ch, CURLOPT\_CAINFO, "~/.docker/aliyun/ClusterName/ca.pem"); curl\_setopt($ch, CURLOPT\_SSLCERT, "~/.docker/aliyun/ClusterName/cert.pem");  $result=curl\_exec($ch); echo $result;  curl\_close($ch);  ?> |

### Python 方式

import requests

res = requests.get('https://123.123.123.123:1234/projects/', verify='~/.docker/aliyun/ClusterName/ca.pem', cert=('~/.docker/aliyun/ClusterName/cert.pem', '~/.docker/aliyun/ClusterName/key.pem')) print res.content

### JAVA 方式

添加 Maven 依赖：

<dependency>

<groupId>org.apache.httpcomponents</groupId>

<artifactId>httpclient</artifactId>

<version>4.5.1</version> </dependency>

<dependency>

<groupId>org.bouncycastle</groupId>

<artifactId>bcpkix-jdk15on</artifactId>

<version>1.52</version> </dependency>

代码示例：

|  |
| --- |
| import java.nio.file.Path; import java.nio.charset.Charset;  import java.nio.file.Files; import java.nio.file.Paths; import java.security.KeyFactory; import java.security.KeyStore; import java.security.PrivateKey; import java.security.cert.Certificate; import java.security.cert.CertificateFactory; import java.security.spec.PKCS8EncodedKeySpec; import javax.net.ssl.SSLContext; import org.bouncycastle.openssl.PEMKeyPair; import org.bouncycastle.openssl.PEMParser;  import org.apache.http.client.methods.CloseableHttpResponse; import org.apache.http.client.methods.HttpGet;  import org.apache.http.conn.ssl.SSLConnectionSocketFactory; import org.apache.http.impl.client.CloseableHttpClient; import org.apache.http.impl.client.HttpClients; import org.apache.http.ssl.SSLContexts; import org.apache.http.util.EntityUtils;  public class Test { public static void main(String[] argc) throws Exception {  final char[] KEY\_STORE\_PASSWORD = "".toCharArray();  //获取证书地址  Path caCertPath = Paths.get("~/.docker/aliyun/ClusterName/ca.pem");  Path clientCertPath = Paths.get("~/.docker/aliyun/ClusterName/cert.pem"); Path clientKeyPath = Paths.get("~/.docker/aliyun/ClusterName/key.pem"); final CertificateFactory cf = CertificateFactory.getInstance("X.509"); |
| final Certificate caCert = cf.generateCertificate(Files.newInputStream(caCertPath)); final Certificate clientCert = cf.generateCertificate( Files.newInputStream(clientCertPath));  final PEMKeyPair clientKeyPair = (PEMKeyPair) new PEMParser(  Files.newBufferedReader(clientKeyPath,  Charset.defaultCharset()))  .readObject();  final PKCS8EncodedKeySpec spec = new PKCS8EncodedKeySpec(  clientKeyPair.getPrivateKeyInfo().getEncoded()); final KeyFactory kf = KeyFactory.getInstance("RSA"); final PrivateKey clientKey = kf.generatePrivate(spec);  //设置信任的证书  final KeyStore trustStore = KeyStore.getInstance(KeyStore.getDefaultType()); trustStore.load(null, null); trustStore.setEntry("ca", new KeyStore.TrustedCertificateEntry(caCert), null);  //设置私钥  final KeyStore keyStore = KeyStore.getInstance(KeyStore.getDefaultType()); keyStore.load(null, null);  keyStore.setCertificateEntry("client", clientCert); keyStore.setKeyEntry("key", clientKey, KEY\_STORE\_PASSWORD, new Certificate[]{clientCert});  SSLContext sslContext = SSLContexts.custom()  .loadTrustMaterial(trustStore, null)  .loadKeyMaterial(keyStore, KEY\_STORE\_PASSWORD)  .build();  SSLConnectionSocketFactory sslsf = new SSLConnectionSocketFactory( sslContext,  SSLConnectionSocketFactory.getDefaultHostnameVerifier());  //httpclient连接  CloseableHttpClient httpclient = HttpClients.custom()  .setSSLSocketFactory(sslsf)  .build(); try {  HttpGet httpget = new HttpGet("https://123.123.123.123:1234/projects/");  CloseableHttpResponse response = httpclient.execute(httpget); try {  System.out.println("----------------------------------------");  String bodyAsString = EntityUtils.toString(response.getEntity());  System.out.println(bodyAsString);  } finally {  response.close();  }  } finally {  httpclient.close();  }  }  } |

应用API列表

# 迁移容器实例

描述

说明：对容器实例进行热迁移

## 请求信息

请求行 RequestLine

Get/Projects/{name:name}/live\_migrate HTTP/1.1

### 请求行参数 URI Param

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 描述 |
| srcIP | string | 是 | 源地址 |
| desIP | string | 是 | 目的地址 |
| strategy | int | 是 | 迁移策略(主要是checkpoint/pre-copy/post-copy这三种) |
| page-server | boolean | 是 | 是否使用page-server传输内存页面 |
| containerd\_id | string | 是 | 容器id |

特有请求头 RequestHead

无

### 请求体 RequestBody

无

## 返回信息

### 返回行 ResponseLine

HTTP/1.1 200 OK

特有返回头 ResponseHead

无

### 返回体 ResponseBody

[

{

“contaienr\_id” : “ String “

”source\_ip” : “String”

”destenation\_ip” : “String”

”live\_migration\_start\_time” : “String”

”live\_migration\_end\_time” : “String”

”strategy ” : “String”

”Page-Server” :” Boolean”

”success” : “ Boolean”

”error” : “ String “

”state” : “String”

”trans\_data\_size” : “String”

”iteration\_num” : “ int “

}

]

### 返回体解释

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| container\_id | string | 被迁移容器对象id |
| source\_ip | string | 迁移宿主机ip |
| destentation\_ip | string | 迁移目的机ip地址 |
| live\_migration\_start\_time | string | 热迁移开始时间 |
| live\_migration\_end\_time | string | 热迁移结束时间 |
| strategy | string | 容器热迁移策略 |
| Page-Server | string | 是否使用PageServer |
| success | boolean | 热迁移是否成功 |
| error | string | 热迁移失败原因 |
| state | string | 被迁移容器的运行状态 |
| trans\_data\_size | string | 迁移过程中传输数据的大小 |
| iteration\_num | int | 内存迁移过程中的迭代迁移次数 |

## 示例

请求示例

Get/Projects/{name:name}/live\_migrate HTTP/1.1

### 返回示例

{

contaienr\_id:e432sw13009

source\_ip:130.128.58.130

destenation\_ip:130.128.50.120

live\_migration\_start\_time:2017/7/2 10:50  
strategy:Pre-Copy  
Page-Server:true  
success:false  
error:docker restore falied,More info in the checkpoint.log  
state:"running on the source!"  
trans\_data\_size:100M  
iteration\_num:6  
}

# 查看应用实例列表

描述

说明：查看您在容器集群中创建的所有应用实例列表。

## 请求信息

请求行 RequestLine

GET /projects/ HTTP/1.1

### 请求行参数 URI Param

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 描述 |
| q | string | 否 | 应用名称 |
| services | boolean | 否 | 是否包含应用的服务信息，缺省值 true |
| containers | boolean | 否 | 是否包含服务的容器信息，缺省值 true |

特有请求头 RequestHead

无

### 请求体 RequestBody

无

## 返回信息

### 返回行 ResponseLine

HTTP/1.1 200 OK

特有返回头 ResponseHead

无

### 返回体 ResponseBody

[

{

"name": "string",

"description": "string",

"template": "string",

"version": "string",

"created": "datetime",

"updated": "datetime",

"desired\_state": "string",

"current\_state": "string",

"environment": {

"key": "value",

...

}

"services": [

...

]

}

]

### 返回体解释

Project 的格式

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| name | string | 应用名称 |
| description | string | 应用描述 |
| template | string | 应用 Compose 模板 |
| version | string | 应用版本 |
| created | datetime | 应用创建时间 |
| updated | datetime | 应用更新时间 |
| desired\_state | string | 期望状态 （如果当前状态是中间状态时，期望状态指明变迁终态） |
| current\_state | string | 当前状态 |
| environment | map | 环境变量 key/value |
| services | array | 服务列表 |

## 示例

请求示例

GET /projects HTTP/1.1

### 返回示例

HTTP/1.1 200 OK

Content-Type:application/json;charset=UTF-8

[

{

"name": "test",

"description": "This is a test application",

"template": "...",

"version": "1.0",

"environment": {

"COMPOSE\_PROJECT\_NAME": "test"

},

"created": "2016-02-02T07:45:13.113833319Z",

"updated": "2016-02-02T07:45:16.03142154Z",

"desired\_state": "running",

"current\_state": "running",

"services": [

...

]

},

...

]

# 创建应用实例

## 描述

说明：创建一个新的应用实例。

## 请求信息

请求行 RequestLine

POST /projects/ HTTP/1.1

请求行参数 URI Param

无

特有请求头 RequestHead

Content-Type: application/json

### 请求体 RequestBody

JSON object

{

"name": "string",

"description": "string",

"template": "string",

"version": "string",

"environment": {

"key": "value",

...

}

}

### 请求体解释

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 必须 | 描述 |
| name | string | 是 | 应用名称。名称为  1~64 个字符，可包含数字，英文字符和连字符（-）,且不能以连字符（-）开头。 |
| description | string | 否 | 应用描述。 |
| template | string | 是 | 字符串格式的应用的  Compose yaml 模板  ，注意需要按照 JSON 格式进行转义。 |
| version | string | 否 | 应用版本，缺省值为  1.0。 |
| environment | map | 否 | key/value 用于替换 Compose 模板的变量参数。 |

## 返回信息

返回行 ResponseLine

HTTP/1.1 201 Created

特有返回头 ResponseHead

Location /projects/<name>

## 示例

### 请求示例

POST /projects HTTP/1.1

Content-Type: application/json

{

"name": "test",

"description": "This is a test application",

"template": "web:\r\n image: nginx",

"version": "1.0",

"environment": {

"USER": "abc",

"PWD": "password"

}

}

### 返回示例

HTTP/1.1 201 Created

Location /projects/test

# 删除应用实例

## 描述

说明：根据应用名称，删除应用实例。

## 请求信息

请求行 RequestLine

DELETE /projects/{name}?force={force}&v={volume} HTTP/1.1

### 请求行参数 URI Param

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 描述 |
| name | string | 是 | 应用实例名称。 |
| force | string | 否 | 是否强制删除，可选值  为 true、1 或 false、 0，缺省为 false。 |
| volume | string | 否 | 是否删除容器卷，可选值为 true、1 或  false、0，缺省为 false。 |

特有请求头 RequestHead

无

### 请求体 RequestBody

无

## 返回信息

返回行 ResponseLine

HTTP/1.1 200 OK

特有返回头 ResponseHead

无

### 返回体 ResponseBody

无

## 示例

### 请求示例

|  |  |  |  |
| --- | --- | --- | --- |
| DELETE /projects/test HTTP/1.1 | | | |
| 返回示例 | |  |  |
| HTTP/1.1 200 OK | |  |  |
| 查看应用实例  描述    说明：根据应用实例名称，查看实例的详细信息。      请求信息    请求行 RequestLine | |  |  |
| GET /projects/{name} HTTP/1.1 | |  |  |
| 请求行参数 URI Param | |  |  |
| 名称 | 类型 | 是否必须 | 描述 |
| name | string | 是 | 应用实例名称 |

特有请求头 RequestHead

无

### 请求体 RequestBody

无

## 返回信息

返回行 ResponseLine

HTTP/1.1 200 OK

特有返回头 ResponseHead

无

### 返回体 ResponseBody

{

"name": "string",

"description": "string",

"template": "string",

"version": "string",

"created": "datetime",

"updated": "datetime",

"desired\_state": "string",

"current\_state": "string",

"environment": {

"key": "value",

...

}

"services": [

...

]

}

### 返回体解释

应用实例的格式

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| name | string | 应用名称 |
| description | string | 应用描述 |
| template | string | 应用Compose模板 |
| version | string | 应用版本 |
| created | datetime | 应用创建时间 |
| updated | datetime | 应用更新时间 |
| desired\_state | string | 期望状态 （如果当前状态是中间状态时，期望状态指明变迁终态） |
| current\_state | string | 当前状态 |
| environment | map | 环境变量key/value |
| services | array | 服务列表 |

## 示例

请求示例

GET /projects/test HTTP/1.1

### 返回示例

HTTP/1.1 200 Ok

Content-Type:application/json;charset=UTF-8

{

"name": "test",

"description": "This is a test application",

"template": "...",

"version": "1.0",

"environment": {

"COMPOSE\_PROJECT\_NAME": "test"

},

"created": "2016-02-02T07:45:13.113833319Z",

"updated": "2016-02-02T07:45:16.03142154Z",

"desired\_state": "running",

"current\_state": "running",

"services": [

...

]

}

# 启动应用实例

描述

说明：启动一个应用实例，会依照服务之间依赖顺序进行启动操作。

## 请求信息

### 请求行 RequestLine

|  |  |  |  |
| --- | --- | --- | --- |
| POST /projects/{name}/start HTTP/1.1 | |  |  |
| 请求行参数 URI Param | |  |  |
| 名称 | 类型 | 是否必须 | 描述 |
| name | string | 是 | 应用实例名称 |
| 特有请求头 RequestHead    无    请求体 RequestBody    无    返回信息    返回行 ResponseLine | |  |  |
| HTTP/1.1 200 OK | |  |  |
| 特有返回头 ResponseHead    无    返回体 ResponseBody    无    示例    请求示例 | |  |  |
| POST /projects/test/start HTTP/1.1 | |  |  |
| 返回示例 | |  |  |
| HTTP/1.1 200 OK | |  |  |

# 停止应用实例

描述

说明：停止一个应用实例所包含的所有容器，会依照服务之间依赖顺序的逆序进行停止操作。

## 请求信息

请求行 RequestLine

POST /projects/{name}/stop?t={timeout} HTTP/1.1

### 请求行参数 URI Param

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 描述 |
| name | string | 是 | 应用实例名称 |
| timeout | int | 否 | 停止容器的超时时间  （以秒为单位），缺省为 10 |

特有请求头 RequestHead

无

### 请求体 RequestBody

无

## 返回信息

返回行 ResponseLine

HTTP/1.1 200 OK

特有返回头 ResponseHead

无

### 返回体 ResponseBody

无

## 示例

请求示例

POST /projects/test/stop HTTP/1.1

返回示例

HTTP/1.1 200 OK

# 终止应用实例

## 描述

说明：向一个应用实例所包含的所有容器发送信号（Signal），会依照服务之间依赖顺序的逆序进行发送。缺

省信号会终止所有容器。

## 请求信息

请求行 RequestLine

POST /projects/{name}/kill?signal={signal} HTTP/1.1

### 请求行参数 URI Param

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 描述 |
| name | string | 是 | 应用实例名称 |
| signal | string | 否 | 缺省为 KILL |

特有请求头 RequestHead

无

### 请求体 RequestBody

无

## 返回信息

返回行 ResponseLine

HTTP/1.1 200 OK

特有返回头 ResponseHead

无

### 返回体 ResponseBody

无

## 示例

请求示例

POST /projects/test/kill HTTP/1.1

返回示例

HTTP/1.1 200 OK

# 更新应用配置

## 描述

说明：更新一个应用实例的配置信息。

## 请求信息

### 请求行 RequestLine

|  |  |  |  |
| --- | --- | --- | --- |
| POST /projects/{name}/update HTTP/1.1 | | | |
| 请求行参数 URI Param | |  |  |
| 名称 | 类型 | 是否必须 | 描述 |
| name | string | 是 | 应用实例名称 |
| 特有请求头 RequestHead | |  |  |
| Content-Type: application/json | |  |  |

### 请求体 RequestBody

JSON object

{

"description": "string",

"template": "string",

"version": "string",

"environment": {

"key": "value",

...

}

}

### 请求体解析

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 必须 | 描述 |
| description | string | 否 | 更新的应用描述。 |
| template | string | 是 | 更新的 Compose yaml 模板，注意需要按照 JSON 格式进行转义。 |
| version | string | 否 | 更新的应用版本，更新的版本应和原有版本不同，否则会返回 HTTP |
|  |  |  | code 409。 |
| environment | map | 环境变量 | key/value 用于替换 Compose 模板的环境变量 |

## 返回信息

返回行 ResponseLine

HTTP/1.1 202 Accepted

特有返回头 ResponseHead

无

### 返回体 ResponseBody

无

## 示例

### 请求示例

POST /projects/test/update HTTP/1.1

Content-Type: application/json

{

"description": "This is a test application",

"template": "web:\r\n image: nginx",

"version": "2.0",

"environment": {

"USER": "abc",

"PWD": "newpwd"

}

}

返回示例

HTTP/1.1 202 Accepted

# 查看服务实例列表

描述

说明：查看您在容器集群中创建的所有服务实例列表。

## 请求信息

请求行 RequestLine

GET /services/ HTTP/1.1

### 请求行参数 URI Param

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 描述 |
| q | string | 否 | 服务名称 |
| containers | boolean | 否 | 是否包含服务的容器信息，缺省值 true |

特有请求头 RequestHead

无

### 请求体 RequestBody

无

## 返回信息

返回行 ResponseLine

HTTP/1.1 200 OK

特有返回头 ResponseHead 无

### 返回体 ResponseBody

|  |
| --- |
| [  {  "id": "string",  "name": "string",  "project": "string",  "description": "string",  "created": "datetime",  "updated": "datetime",  "desired\_state": "string",  "current\_state": "string",  "definition": { "key": "value",  ...  },  "extensions": {  "key": "value",  ...  },  "containers": {  "key": "value",  ...  }  },  ...  ] |

### 返回体解释

服务实例的格式

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| id | string | 服务 ID |
| name | string | 服务名称 |
| project | string | 应用名称 |
| created | datetime | 服务创建时间 |
| updated | datetime | 服务更新时间 |
| desired\_state | string | 期望状态 （如果当前状态是中间状态时，期望状态指明变迁终态） |
| current\_state | string | 当前状态 |
| definition | map | Compose 中服务定义  key/value |
| extensions | map | 容器服务 Compose 中服务扩  展 key/value |
| containers | map | 服务中所包含容器 key （容器 |
|  |  | id）/value（属性） |

## 示例

请求示例

GET /services/ HTTP/1.1

### 返回示例

|  |
| --- |
| HTTP/1.1 200 Ok  Content-Type:application/json;charset=UTF-8  [  {  "id": "wordpress\_db",  "name": "db",  "project": "wordpress",  "definition": {  "environment": [  "MYSQL\_ROOT\_PASSWORD=password"  ],  "image": "mysql:5.7",  "restart": "always"  },  "extensions": {  "scale": 1,  "logs": [  "/var/log/mysql"  ]  },  "created": "2016-04-21T13:36:32.440646459Z",  "updated": "2016-04-21T13:36:33.270308958Z",  "desired\_state": "running",  "current\_state": "running", "containers": {  "5616f05d27516b3502a391fd2ca9d312cabffa5ad431bf261ea81f4ceabd476e": { "name": "/wordpress\_db\_1",  "node": "10.246.2.3",  "ip": "10.0.0.2",  "running": true,  "status": "running",  "health": "success"  }  }  }  ] |

# 查看服务实例

## 描述

说明：根据服务实例 ID 查看详细信息。

## 请求信息

### 请求行 RequestLine

|  |  |  |  |
| --- | --- | --- | --- |
| GET /services/{service\_id} HTTP/1.1 | | | |
| 请求行参数 URI Param | |  |  |
| 名称 | 类型 | 是否必须 | 描述 |
| service\_id | string | 是 | 服务实例 ID, 格式为  {project\_name}\_{serv ice\_name} |
| 特有请求头 RequestHead    无    请求体 RequestBody    无    返回信息    返回行 ResponseLine | |  |  |
| HTTP/1.1 200 OK | |  |  |

特有返回头 ResponseHead

无

### 返回体 ResponseBody

|  |
| --- |
| {  "id": "string",  "name": "string",  "project": "string",  "description": "string",  "created": "datetime",  "updated": "datetime",  "desired\_state": "string",  "current\_state": "string",  "definition": { "key": "value",  ...  },  "extensions": {  "key": "value",  ...  },  "containers": {  "key": "value",  ...  }  } |

### 返回体解释

服务实例的格式

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| id | string | 服务 ID |
| name | string | 服务名称 |
| project | string | 应用名称 |
| created | datetime | 服务创建时间 |
| updated | datetime | 服务更新时间 |
| desired\_state | string | 期望状态 （如果当前状态是中间状态时，期望状态指明变迁终态） |
| current\_state | string | 当前状态 |
| definition | map | Compose 中服务定义  key/value |
| extensions | map | 容器服务 Compose 中服务扩  展 key/value |
| containers | map | 服务中所包含容器 key （容器  ID）/value（属性） |

## 示例

请求示例

GET /services/wordpress\_db HTTP/1.1

### 返回示例

|  |
| --- |
| HTTP/1.1 200 Ok  Content-Type:application/json;charset=UTF-8  {  "id": "wordpress\_db",  "name": "db",  "project": "wordpress",  "definition": {  "environment": [  "MYSQL\_ROOT\_PASSWORD=password"  ],  "image": "mysql:5.7",  "restart": "always"  },  "extensions": {  "scale": 1,  "logs": [  "/var/log/mysql"  ]  },  "created": "2016-04-21T13:36:32.440646459Z",  "updated": "2016-04-21T13:36:33.270308958Z",  "desired\_state": "running",  "current\_state": "running", "containers": {  "5616f05d27516b3502a391fd2ca9d312cabffa5ad431bf261ea81f4ceabd476e": { "name": "/wordpress\_db\_1",  "node": "10.246.2.3",  "ip": "10.0.0.2",  "running": true,  "status": "running",  "health": "success"  }  }  } |

# 启动服务实例

## 描述

说明：启动一个服务实例中所有容器。

## 请求信息

### 请求行 RequestLine

|  |  |  |  |
| --- | --- | --- | --- |
| POST /services/{service\_id}/start HTTP/1.1 | | | |
| 请求行参数 URI Param | |  |  |
| 名称 | 类型 | 是否必须 | 描述 |
| service\_id | string | 是 | 服务实例 ID, 格式为  {project\_name}\_{serv ice\_name} |
| 特有请求头 RequestHead    无    请求体 RequestBody    无    返回信息    返回行 ResponseLine | |  |  |
| HTTP/1.1 200 OK | |  |  |

特有返回头 ResponseHead

无

### 返回体 ResponseBody

无

## 示例

请求示例

POST /service/test\_web/start HTTP/1.1

返回示例

HTTP/1.1 200 OK

# 停止服务实例

描述

说明：停止一个服务实例中所有容器。

## 请求信息

请求行 RequestLine

POST /services/{service\_id}/stop?t={timeout} HTTP/1.1

### 请求行参数 URI Param

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 描述 |
| service\_id | string | 是 | 服务实例 ID, 格式为  {project\_name}\_{serv ice\_name} |
| timeout | int | 否 | 停止容器的超时时间  （以秒为单位），缺省为 10 |

特有请求头 RequestHead

无

### 请求体 RequestBody

无

## 返回信息

返回行 ResponseLine

HTTP/1.1 200 OK

特有返回头 ResponseHead

无

### 返回体 ResponseBody

无

## 示例

请求示例

POST /projects/test\_web/stop HTTP/1.1

返回示例

HTTP/1.1 200 OK

# 终止服务实例

描述

说明：向一个服务实例中所有容器发送信号（Signal），缺省信号会终止所有容器。

## 请求信息

请求行 RequestLine

POST /services/{service\_id}/kill?signal={signal} HTTP/1.1

### 请求行参数 URI Param

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 描述 |
| service\_id | string | 是 | 服务实例 ID, 格式为  {project\_name}\_{serv ice\_name} |
| signal | string | 否 | 缺省为 KILL |

特有请求头 RequestHead

无

### 请求体 RequestBody

无

## 返回信息

返回行 ResponseLine

HTTP/1.1 200 OK

特有返回头 ResponseHead

无

### 返回体 ResponseBody

无

## 示例

### 请求示例

|  |  |  |  |
| --- | --- | --- | --- |
| POST /projects/test\_web/kill HTTP/1.1 | |  |  |
| 返回示例 | |  |  |
| HTTP/1.1 200 OK | |  |  |
| 伸缩服务实例  描述    说明：伸缩一个服务实例中所包含容器到指定数量。      请求信息    请求行 RequestLine | |  |  |
| POST /services/{service\_id}/scale HTTP/1.1 | |  |  |
| 请求行参数 URI Param | |  |  |
| 名称 | 类型 | 是否必须 | 描述 |
| service\_id | string | 是 | 服务实例 ID, 格式为  {project\_name}\_{serv ice\_name} |
| 特有请求头 RequestHead    无    请求体 RequestBody    JSON object | |  |  |
| {  "scale\_to": "int"  } | |  |  |

## 返回信息

返回行 ResponseLine

HTTP/1.1 200 OK

特有返回头 ResponseHead

无

### 返回体 ResponseBody

无

## 示例

### 请求示例

将 test\_web 服务定义的容器实例数改变为 3 个。

POST /projects/test\_web/scale HTTP/1.1

Content-Type: application/json

{

"scale\_to": 3

}

返回示例

HTTP/1.1 200 OK

# 触发器

## 简介

触发器是容器服务提供的简单快捷地进行持续部署的 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 API。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。

## 创建触发器

### 操作流程

登录 容器服务管理控制台。

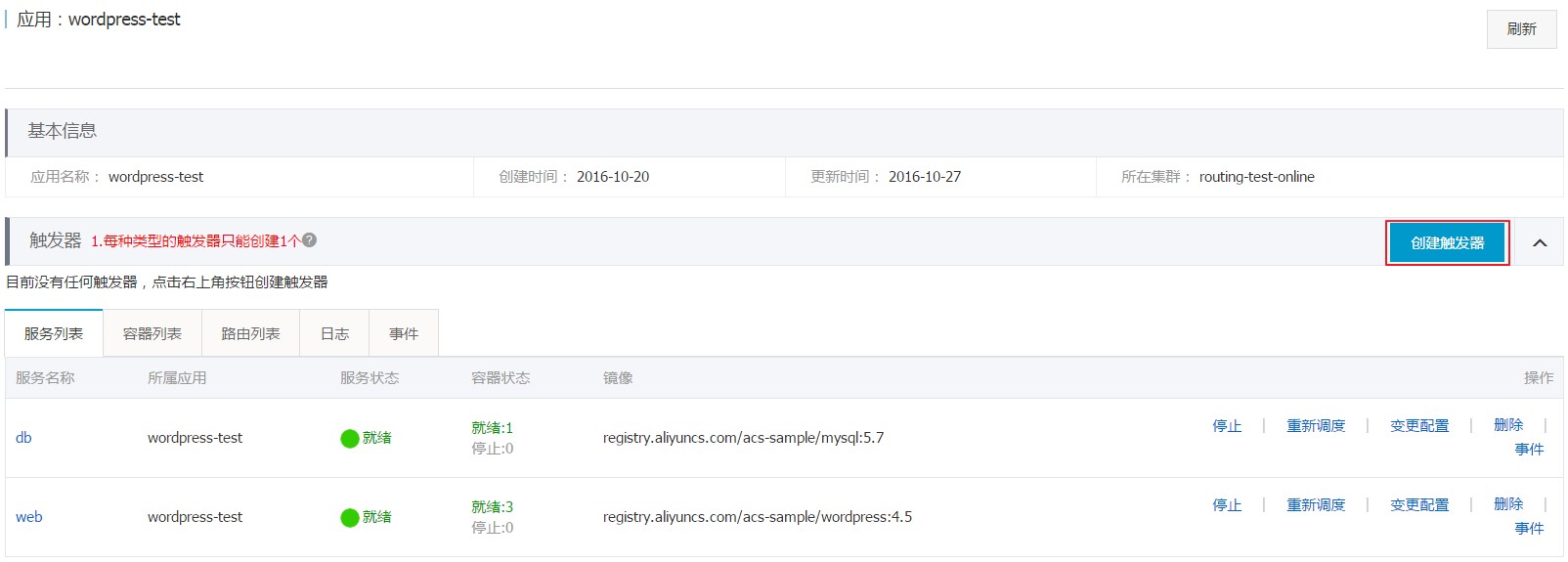
单击左侧导航栏中的 应用。

选择目标应用所在的集群。

选择目标应用并单击应用的名称。如下图所示。



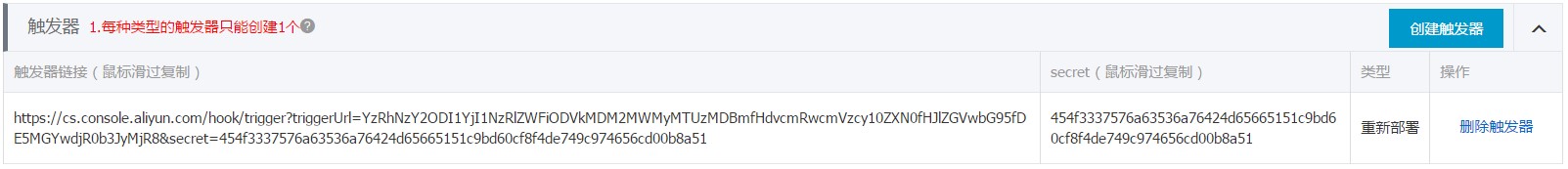
在应用详情页面，单击 创建触发器。如下图所示。



选择 重新部署 并单击 确定。



此时生成的触发器地址即为 API 的地址。



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

。

curl

https://cs.console.aliyun.com/hook/trigger?triggerUrl=YzI4YTk5NzFkZWZkYzQ2MTJiOWZkNTM1MzY2Z DU1M2NifGNvbGxlY3RkLWJlbmNobWFya3xyZWRlcGxveXwxOGlxbjc1Z25uMmVzfA==&secret=44586c6 b466352395143584c3970654ff5323d2509d546fdc1b33054b0928da8

# 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。

注意：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。

注意：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：

name=my-clusters + "\n" + 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:/uA9QF5CHrr1FK3siBA4xLMTWE0=

{"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（注意：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。

注意：第一行行尾有一个 \n 的换行符。

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  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))):

ZmVlMDNkNDA1ZTQyMWViYWY1MTRhZGVjODgxMDM4YzRiMzEzNTg0ZA==

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:/uA9QF5CHrr1FK3siBA4xLMTWE0=  {"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"} |

附录

错误代码表

错误代码表格

# 状态表

|  |  |
| --- | --- |
| 状态名 | 状态说明 |
| Stopped | 已停止 |
| Starting | 启动中 |
| Running | 运行中 |
| Stopping | 停止中 |

|  |  |
| --- | --- |
| Deleted | 已释放 |